

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:

Syafril 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 ijin tertulis dari penerbit

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

Imam Syafi'i

CONTRIBUTORS

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

CONTENTS IN BRIEF

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

Listings

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 Internship.

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 pengantar para programmer, analis, IT Operation dan Project Manajer. Dalam melakukan implementasi git pada diri dan organisasinya.

Rumusnya cuman sebagai contoh aja biar keren[?].

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

BAB 1

CHAPTER 1

1.1 1174070 - Arrizal Furqona Gifary

1.1.1 Teori

1. Sejarah dan Perkembangan

Kecerdasan Buatan atau dalam Bahasa Inggris sering disebut Artificial Intelligence yang sering disebut juga sebagai AI, pada 10 tahun lalu masyarakat belum terlalu mengetahui hal tersebut dan masih menjadi bahan candaan dikalangan masyarakat. Awal perkembangan AI dimulai pada tahun 1952-1969 yang dimulai dengan kesuksesan Newell dan temannya Simon menggunakan sebuah program yang disebut dengan General Problem Solver. Program ini dibangun untuk tujuan penyelesaian masalah secara manusiawi. Pada tahun 1966-1974 perkembangan kecerdasan buatan mulai melambat. Ada 3 faktor utama yang menyebabkan hal itu terjadi:

- Banyak subjek pada program AI yang bermunculan hanya mengandung sedikit atau bahkan sama sekali tidak mengandung sama sekali pengetahuan (knowledge).

- Kecerdasan buatan harus bisa menyelesaikan banyak masalah.
- Untuk menghasilkan perilkau inteliensia ada beberapa batasan pada struktur yang bisa digunakan.

Definisi kecerdasan buatan itu sendiri adalah suatu system teknologi yang didalamnya ditambahakan kecerdasan oleh manusia, kecerdasan buatan diatur dan dikembangkan dalam konteks ilmiah, dan bentukan dari kecerdasan entitas ilmiah yang ada.

2. Definisi

Supervised learning, klasifikasi, regresi, unsupervised learning, dataset, trainingset dan testingset.

- **Supervised Learning**

Supervised Learning merupakan sebuah tipe learning yang mempunyai variable input dan variable output, tipe ini juga menggunakan satu algoritma atau lebih dari satu algoritma yang digunakan untuk mempelajari fungsi pemetaan dari input ke output.

- **Klasifikasi**

Klasifikasi adalah pengelompokan data di mana data yang digunakan memiliki label atau kelas target. Sehingga algoritma untuk menyelesaikan masalah klasifikasi dikategorikan ke dalam pembelajaran terbimbing.

- **Regresi**

regressi metode analisis statistik yang digunakan untuk dapat melihat efek antara dua atau lebih variabel. Hubungan variabel dalam pertanyaan adalah fungsional yang diwujudkan dalam bentuk model matematika. Dalam analisis regresi, variabel dibagi menjadi dua jenis, yaitu variabel respons atau yang biasa disebut variabel dependen dan variabel independen atau dikenal sebagai variabel independen. Ada beberapa jenis analisis regresi, yaitu regresi sederhana yang mencakup linear sederhana dan regresi non-linear sederhana dan regresi berganda yang mencakup banyak linier atau non-linear berganda. Analisis regresi digunakan dalam pembelajaran mesin pembelajaran dengan metode pembelajaran terawasi.

- **Unsupervised learning**

unsupervised learning jenis pembelajaran di mana kita hanya memiliki data input (input data) tetapi tidak ada variabel output yang terkait. Tujuan dari pembelajaran tanpa pengawasan adalah untuk memodelkan struktur dasar atau distribusi data dengan tujuan mempelajari data lebih lanjut, dengan kata lain, itu adalah fungsi simpulan yang menggambarkan atau menjelaskan data.

- **Data set**

Data set objek yang merepresentasikan data dan relasinya di memory.

Strukturnya mirip dengan data di database. Dataset berisi koleksi dari datatable dan datarelation.

- Training Set

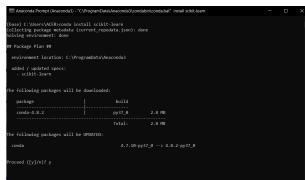
Training set adalah bagian dari dataset yang di latih untuk membuat prediksi atau menjalankan fungsi dari algoritma ML lain sesuai dengan masing-masing. Memberikan instruksi melalui algoritma sehingga mesin yang di praktikkan dapat menemukan korelasinya sendiri.

- Testing Set

testing set adalah bagian dari dataset yang kami uji untuk melihat akurasinya, atau dengan kata lain untuk melihat kinerjanya.

1.1.2 Praktek

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



Gambar 1.1 Instalasi Package Scikit Learn

Name	Type	Size	Value
X	float32	(10, 200)	[0.54888155 0.71551834 0.00270535 ... -0.4001078 0.64586484 0.5017755]
X_norm	float64	(10, 2071)	[1.0808871 1.3275481 -0.66959556 ... -0.52686495 0.45897815 ...]
x	float64	(1, 4)	[1. [2. 3. 4.]]
Uris	utils.Runch	8	Bunch object of sklearn.utils module
x	float64	(159, 4)	[5.1 5.5 1.4 8.2]
y	int32	(159,)	[0 0 0 ... 0]

Gambar 1.2 Isi Variabel Explorer

- ## 2. Mencoba loading an example dataset

```
1 from sklearn import datasets # Digunakan Untuk Memanggil  
2         class datasets dari library sklearn  
3 iris = datasets.load_iris() # Menggunakan contoh datasets  
4         iris  
5 x = iris.data             # Menyimpan nilai data sets iris  
6         pada variabel x  
7 y = iris.target            # Menyimpan nilai data label iris  
8         pada variabel y
```

- ### 3. Mencoba Learning dan predicting

```

1 from sklearn.neighbors import KNeighborsClassifier #Digunakan
   Untuk Memanggil fungsi KNeighborsClassifier
2                                         # pada
3 class sklearn dan library sklearn
4 import numpy as np # memanggil library numpy dan dibuat alias
   np
5 knn=KNeighborsClassifier(n_neighbors=1) #membuat variabel kkn
   , dan memanggil fungsi KNeighborsClassifier
6                                         #dan mendefinisikan k
   -nya adalah 1
7 knn.fit(x,y)                         #Perhitungan
   matematika library kkn
8 a=np.array([1.0,2.0,3.0,4.0])        #Membuat Array
9 a = a.reshape(1,-1)                  #Mengubah Bentuk
   Array jadi 1 dimensi
10 hasil = knn.predict(a)              #Memanggil fungsi
   prediksi
11 print(hasil)                      #menampilkan hasil
   prediksi

```

4. Mencoba Model Persistence

```

1 from sklearn import svm # Digunakan untuk memanggil class svm
   dari library sklearn
2 from sklearn import datasets # Diguangkan untuk class datasets
   dari library sklearn
3 clf = svm.SVC()           # membuat variabel clf , dan
   memanggil class svm dan fungsi SVC
4 X, y = datasets.load_iris(return_X_y=True) #Mengambil dataset
   iris dan mengembalikan nilainya.
5 clf.fit(X, y)             #Perhitungan nilai label
6
7 from joblib import dump, load #memanggil class dump dan load
   pada library joblib
8 dump(clf, '1174070.joblib') #Menyimpan model kedalam 1174027.
   joblib
9 hasil = load('1174070.joblib') #Memanggil model 1174027
10 print(hasil) # Menampilkan Model yang dipanggil sebelumnya

```

5. Mencoba Conventions

```

1 import numpy as np # memanggil library numpy dan dibuat alias
   np
2 from sklearn import random_projection #Memanggil class
   random_projection pada library sklean
3
4 rng = np.random.RandomState(0) #Membuat variabel rng , dan
   mendefinisikan np, fungsi random dan attr RandomState
   kedalam variabel
5 X = rng.rand(10, 2000) # membuat variabel X, dan menentukan
   nilai random dari 10 – 2000
6 X = np.array(X, dtype='float32') #menyimpan hasil nilai
   random sebelumnya, kedalam array , dan menentukan
   typedatanya sebagai float32

```

```

7 X.dtype # Mengubah data tipe menjadi float64
8
9 transformer = random_projection.GaussianRandomProjection() #
    membuat variabel transformer , dan mendefinisikan
    classrandom_projection dan memanggil fungsi
    GaussianRandomProjection
10 X_new = transformer.fit_transform(X) # membuat variabel baru
    dan melakukan perhitungan label pada variabel X
11 X_new.dtype # Mengubah data tipe menjadi float64
12 print(X_new) # Menampilkan isi variabel X_new

```

1.1.3 Penanganan Error

1. ScreenShoot Error

```

file "D:\Vullah\Semester 6\Tercerdason\Butan\lah\sklearn.py", line 8, in <module>
    from sklearn import datasets
ImportError: cannot import name 'datasets' from 'sklearn' (D:\Vullah\Semester
6\tercerdason\Butan\lah\sklearn.py)

```

Gambar 1.3 Import Error

```

file "C:\Users\svirul\Anaconda3\lib\site-packages\sklearn\utils\validation.py", line
556, in check_array
    "Expected 2D array, got %r instead"
ValueError: Expected 2D array, got 1D array instead
array[[ 1.23]
      [ 2.34]
      [ 3.45]
      [ 4.56]
      [ 5.67]
      [ 6.78]
      [ 7.89]
      [ 8.90]
      [ 9.01]
      [ 10.12]]
Reshape your data either using array.reshape(-1, 1) if your data has a single feature or
array.reshape(1, -1) if it contains a single sample.

```

Gambar 1.4 Value Error

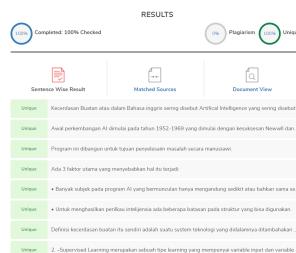
2. Tuliskan Kode Error dan Jenis Error

- Import Error
- Value Error

3. Cara Penangan Error

- Import Error
Dengan Menginstall Library Yang Tidak Ditemukan
- Value Error
Mengubah Bentuk Arraynya, Menjadi 1 Dimensi

1.1.4 Bukti Tidak Plagiat



Gambar 1.5 Bukti Tidak Melakukan Plagiat 1



Gambar 1.6 Bukti Tidak Melakukan Plagiat 2

1.2 Fanny Shafira Damayanti (1174069)

1.2.1 Teori

1. Definisi Kecerdasan buatan

Kecerdasan buatan atau Artificial intelligence merupakan kecerdasan yang ditambahkan kedalam suatu sistem yang diatur secara ilmiah. Kecerdasan buatan dibuat untuk menggantikan pekerjaan yang dilakukan oleh manusia menjadi dikerjakan oleh sistem.

2. Sejarah Kecerdasan Buatan

- Abad 17, Rene Descartes berkata bahwa tubuh hewan adalah sekumpulan mesin yang rumit.
- 1642, Blaise Pascal menciptakan mesin penghitung digital mekanis pertama.
- Abad 19, Charles Babbage dan Ada Lovelace bekerja di program penghitung mekanis.
- 1950, John McCarthy membuat istilah “Kecerdasan Buatan”.
- 1960-1970, Joel Moses membuat program yang pertama kali sukses dalam bidang matematika.
- 1980, jaringan saraf digunakan secara meluas dengan algoritme perambatan balik.

- 2004, DARPA membuat kendaraan yang bisa dijalankan sendiri tanpa manusia.

3. Perkembangan kecerdasan buatan

- Masa persiapan (1943-1946) Warren McCulloch dan Walter Pitt menge-mukakan tiga hal : pengetahuan fisiologi dasar dan fungsi sel syaraf dalam otak, analisa formal tentang logika proposisi, dan teori komputasi Turing.

Pada tahun 1950, Norbert Wiener membuat penelitian mengenai prinsip-prinsip teori feedback.

Pada tahun 1956, John McCarthy meyakinkan Minsky, Claude Shannon dan Nathaniel Rochester untuk membantunya melakukan penelitian dalam bidang Otomata, Jaringan Syaraf dan pembelajaran inteliijensia.

- Awal perkembangan (1952-1969) Pada tahun 1958, McCarthy di MIT AI Lab Memo No.1 mendefinisikan bahasa pemrograman tingkat tinggi yaitu LISP,

Pada tahun 1959, Nathaniel Rochester dari IBM dan mahasiswa-mahasiswanya mengeluarkan program kecerdasan buatan yaitu Geometry Theorem Prover.

Pada tahun 1963, program yang dibuat James Slagle mampu menyelesaikan masalah integral tertutup untuk mata kuliah Kalkulus. Pada tahun 1986, program analogi buatan Tom Evan menyelesaikan masalah analogi geometris yang ada pada tes IQ.

- Perkembangan Kecerdasan Buatan Melambat (1969-1979) Bruce Buchanan dan Joshua Lederberg yang membuat program untuk memecahkan masalah struktur molekul dari informasi yang didapatkan dari spectrometer massa.
- AI Menjadi sebuah industri Industrialisasi kecerdasan buatan diawali dengan ditemukannya sistem pakar yang dinamakan R1 yang mampu mengkonfigurasi system-sistem computer baru.
- Kembalinya Jaringan Syaraf Tiruan (1986-sekarang) Pada tahun 1985-an setidaknya empat kelompok riset menemukan kembali algoritma belajar propagasi balik (Back-Propagation Learning). Algoritma ini berhasil diimplementasikan ke dalam bidang ilmu computer dan psikologi.

4. Definisi Supervised Learning

Supervised Learning merupakan cabang dari Artificial Intelligence. supervised learning adalah suatu ilmu yang mempelajari perancangan dan pengembangan algoritma.

5. Klasifikasi Supervised Learning

- Logistic regression.

- K-nearest neighbors.
 - Support vector machine (SVM)
 - Naive Bayes.
 - Decision tree classification.
 - Random forest classification.

6. Regresi dan Unsupervised Learning

Regresi merupakan sebuah metode analisis statistic yang digunakan untuk mengetahui pengaruh antara dua variable atau lebih.

Untuk mempelajari Unsupervised learning kita tidak perlu data training untuk melakukan prediksi maupun klasifikasi.

7. Dataset

Dataset merupakan objek yang mempresentasikan data dan relasinya pada memori.

8. Training Set

Training Set merupakan bagian dari dataset untuk membuat prediksi atau menjalankan fungsi dari sebuah algoritma Machine Learning.

9. Testing Set

Testing set digunakan untuk mengukur apakah classifier berhasil melakukan klasifikasi dengan benar.

1.2.2 Instalasi

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

Gambar 1.7 Instalasi Package Scikit Learn

Name	Type	Size	Value
digits	utils.Bunch	5	Bunch object of sklearn.utils module
iris	utils.Bunch	6	Bunch object of sklearn.utils module

Gambar 1.8 Isi Variabel Explorer

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

```

1 #%% Mencoba loading an example dataset
2 from sklearn import datasets # Digunakan Untuk Memanggil
   class datasets dari library sklearn
3 iris = datasets.load_iris() # Menggunakan contoh datasets
   iris
4 x = iris.data           # Menyimpan nilai data sets iris
   pada variabel x
5 y = iris.target         # Menyimpan nilai data label iris
   pada variabel y

```

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

```

1 #%%Mencoba Learning dan predicting
2 from sklearn.neighbors import KNeighborsClassifier #Digunakan
   Untuk Memanggil fungsi KNeighborsClassifier
3                                     # pada
   class sklearn dan library sklearn
4 import numpy as np # memanggil library numpy dan dibuat alias
   np
5 knn=KNeighborsClassifier(n_neighbors=1) #membuat variabel kkn
   , dan memanggil fungsi KNeighborsClassifier
6                                     #dan mendefinisikan k
   -nya adalah 1
7 knn.fit(x,y)                      #Perhitungan
   matematika library kkn
8 a=np.array([1.0,2.0,3.0,4.0])      #Membuat Array
9 a = a.reshape(1,-1)                 #Mengubah Bentuk
   Array jadi 1 dimensi
10 hasil = knn.predict(a)            #Memanggil fungsi
   prediksi

```

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

```

1 #%% Model Persistense
2 from sklearn import svm # Digunakan untuk memanggil class svm
   dari library sklearn
3 from sklearn import datasets # Diguangkan untuk class datasets
   dari library sklearn
4 clf = svm.SVC()           # membuat variabel clf , dan
   memanggil class svm dan fungsi SVC
5 X, y = datasets.load_iris(return_X_y=True) #Mengambil dataset
   iris dan mengembalikan nilainya.
6 clf.fit(X, y)             #Perhitungan nilai label
7
8 from joblib import dump, load #memanggil class dump dan load
   pada library joblib

```

```

9 dump(clf, '1174069.joblib') #Menyimpan model kedalam 1174069.
   joblib
10 hasil = load('1174069.joblib') #Memanggil model 1174069

```

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

```

1 %% Conventions
2 import numpy as np # memanggil library numpy dan dibuat alias
   np
3 from sklearn import random_projection #Memanggil class
   random_projection pada library sklearn
4
5 rng = np.random.RandomState(0) #Membuat variabel rng , dan
   mendefinisikan np, fungsi random dan attr RandomState
   kedalam variabel
6 X = rng.rand(10, 2000) # membuat variabel X, dan menentukan
   nilai random dari 10 – 2000
7 X = np.array(X, dtype='float32') #menyimpan hasil nilai
   random sebelumnya, kedalam array , dan menentukan
   typedatanya sebagai float32
8 X.dtype # Mengubah data tipe menjadi float64
9
10 transformer = random_projection.GaussianRandomProjection() #
   membuat variabel transformer , dan mendefinisikan
   classrandom_projection dan memanggil fungsi
   GaussianRandomProjection
11 X_new = transformer.fit_transform(X) # membuat variabel baru
   dan melakukan perhitungan label pada variabel X
12 X_new.dtype # Mengubah data tipe menjadi float64

```

1.2.3 Penanganan Error

1. ScreenShoot Error

```

File "D:\Kuliah\Semester 5\Kecerdasan Buatan\Latihan\1\sklearn.py", line 8, in <module>
    from sklearn import datasets
ImportError: cannot import name 'datasets' from 'sklearn' (D:\Kuliah\Semester
5\Kecerdasan Buatan\Latihan\1\sklearn.py)

```

Gambar 1.9 Import Error

2. Tuliskan Kode Error dan Jenis Error

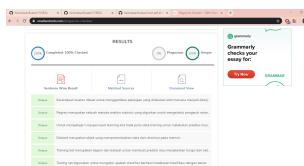
- Import Error

3. Cara Penangan Error

- Import Error

Dengan Menginstall Library Yang Tidak Ditemukan

1.2.4 Bukti Tidak Plagiat



Gambar 1.10 Bukti Tidak Melakukan Plagiat Chapter 1

1.2.5 Link Youtube

<https://youtu.be/Ra4Lu-C8OQY>

1.3 1174086 - Tia Nur Candida

1.3.1 Teori

1. Definisi Kecerdasan Buatan

Kecerdasan Buatan adalah suatu ilmu yang mempelajari bagaimana cara komputer melakukan sesuatu seperti yang dilakukan oleh manusia. Secara sederhana AI adalah teknik dan ilmu untuk membangun atau membuat suatu mesin menjadi cerdas, terutama pada program komputer. Kecerdasan yang dimaksud yaitu seperti yang dimiliki oleh manusia namun pada mesin akan dibuat cepat dan tepat atau akurat.

2. Sejarah Kecerdasan Buatan Sejarah kecerdasan buatan dimulai pada za-

man kuno. Benih kecerdasan buatan modern ditanamkan oleh filusif klasik dengan berusaha menggambarkan proses berpikir manusia. Karya tersebut memuncak pada penemuan komputer digital yang di program pada tahun 1940 an, dimana terdapat sebuah mesin yang didasarkan pada esensi abstrak penalaran matematika. Istilah kecerdasan buatan pertama kali dikemukaan pada tahun 1956 di Konferensi Dartmouth yang kemudian sejak saat itu kecerdasan buatan terus berkembang.

3. Perkembangan kecerdasan buatan

- Masa Persiapan AI (1943-1956) Pada tahun 1943, Warren McCulloch dan Walter Pitt mengemukakan tiga hal : pengetahuan fisiologi dasar dan fungsi sel syaraf dalam otak, analisa formal tentang logika proposisi, dan teori komputasi Turing. Mereka berhasil membuat suatu model sel syaraf tiruan dimana setiap sel syaraf digambarkan sebagai ‘on’ dan ‘off’. Mereka menunjukkan bahwa setiap fungsi dapat dihitung dengan suatu jaringan sel syaraf dan bahwa semua hubungan

logis dapat diimplementasikan dengan struktur jaringan yang sederhana. Pada tahun 1950, Nobert Wiener membuat penelitian mengenai prinsip-prinsip teori feedback. Contoh yang terkenal adalah thermostat. Penemuan ini juga merupakan awal dari perkembangan AI. Pada tahun 1956, John McCarthy meyakinkan Minsky, Claude Shannon dan Nathaniel Rochester untuk membantunya melakukan penelitian dalam bidang Otomata, Jaringan Syaraf dan pembelajaran intelijensia. Mereka mengerjakan proyek ini selama 2 bulan di Dartmouth. Hasilnya adalah program yang mampu berpikir non-numerik dan menyelesaikan masalah pemikiran, yang dinamakan Principia Mathematica. Hal ini menjadikan McCarthy disebut sebagai bapak kecerdasan buatan.

- Awal perkembangan AI (1952-1969) Kecerdasan buatan banyak mengalami kesuksesan pada tahun pertama. Pada tahun 1958, McCarthy di MIT AI Lab Memo No.1 mendefinisikan bahasa pemrograman tingkat tinggi yaitu LISP, yang sekarang mendominasi pembuatan program-program kecerdasan buatan. Kemudian, McCarthy membuat program yang dinamakan Programs with Common Sense. Di dalam program tersebut, dibuat rancangan untuk menggunakan pengetahuan dalam mencari solusi. Pada tahun 1959, Nathaniel Rochester dari IBM dan mahasiswa-mahasiswanya mengeluarkan program kecerdasan buatan yaitu Geometry Theorem Prover. Program ini dapat mengeluarkan suatu teorema menggunakan aksioma-aksioma yang ada. Pada tahun 1963, program yang dibuat James Slagle mampu menyelesaikan masalah integral tertutup untuk mata kuliah Kalkulus. Pada tahun 1986, program analogi buatan Tom Evan menyelesaikan masalah analogi geometris yang ada pada tes IQ.
- Perkembangan kecerdasan buatan melambat (1966-1974) Banyak masalah yang perlu di selesaikan oleh kecerdasan buatan dan baru sedikit program yang keluar menyebabkan melambat.
- Kecerdasan buatan menjadi sebuah industri (1980 - 1988) Industrialisasi kecerdasan buatan diawali dengan ditemukannya sistem pakar yang dinamakan R1 yang mampu mengkonfigurasi sistem-sistem computer baru. Program tersebut mulai dioperasikan di Digital Equipment Corporation (DEC), McDermott, pada tahun 1982. Pada tahun 1986, R1 telah berhasil menghemat US Dolar 40 juta per tahun. Pada tahun 1988, kelompok kecerdasan buatan di DEC menjalankan 40 sistem pakar. Hampir semua perusahaan besar di USA mempunyai divisi AI. Sehingga perusahaan yang sejak tahun 1982 hanya menghasilkan beberapa juta US dolar per tahun meningkat menjadi 2 miliar US dolar per tahun pada tahun 1988.
- Kembalinya Jaringan Syaraf Tiruan (1986 - Sekarang) Meskipun bidang ilmu computer menolak jaringan syaraf tiruan setelah diterbitkannya buku "Perceptrons" karangan Minsky dan Papert, tetapi

para ilmuwan masih mempelajari bidang ilmu tersebut dari sudut pandang yang lain yaitu fisika. Para ahli fisika seperti Hopfield (1982) menggunakan teknik-teknik mekanika statistika untuk menganalisa sifat-sifat pentimpanan dan optimasi pada jaringan syaraf. Para ahli psikologi, David Rumelhart dan Geoff Hinton, melanjutkan penelitian mengenai model jaringan syaraf tiruan pada memori. Pada tahun 1985-an setidaknya empat kelompok riset menemukan kembali algoritma belajar propagasi balik (Back-Propagation Learning). Algoritma ini berhasil diimplementasikan ke dalam bidang ilmu computer dan psikologi.

4. Definisi Supervised Learning

Merupakan tipe Machine Learning dimana model ini menyediakan training data berlabel. Supervised learning merupakan suatu pembelajaran yang terawasi dimana jika output yang diharapkan telah diketahui sebelumnya. Supervised Learning adalah tipe learning di mana kita mempunyai variable input dan variable output, dan menggunakan satu algoritma atau lebih untuk mempelajari fungsi pemetaan dari input ke output. Goal-nya adalah untuk memperkirakan fungsi pemetaannya, sehingga ketika kita mempunya input baru, kita dapat memprediksi output untuk input tersebut.

5. Klasifikasi

- Logistic regression.
- K-nearest neighbors.
- Support vector machine (SVM).
- Naive Bayes.
- Decision tree classification.
- Random forest classification.

6. Regresi

Regresi adalah suatu metode analisis statistik yang digunakan untuk melihat pengaruh antara dua atau lebih banyak variabel. Hubungan variabel tersebut bersifat fungsional yang diwujudkan dalam suatu model matematis.

7. Unsupervised Learning

Unsupervised Learning adalah tipe learning di mana kita hanya mempunyai data masukan (input data) tetapi tidak ada output variable yang berhubungan.

Goal dari unsupervised learning adalah untuk memodelkan struktur dasar atau distribusi dalam data dengan tujuan untuk mempelajari data lebih

jauh lagi, dengan kata lain, adalah menyimpulkan fungsi yang mendeskripsikan atau menjelaskan data.

8. Dataset

Dataset adalah objek yang merepresentasikan data dan relasinya di memory. Strukturnya mirip dengan data di database. Dataset berisi koleksi dari datatable dan datarelation.

9. Training Set

Training set adalah bagian dataset yang kita latih untuk membuat prediksi atau menjalankan fungsi dari sebuah algoritma ML lainnya sesuai tujuannya masing-masing. Kita memberikan petunjuk melalui algoritma agar mesin yang kita latih bisa mencari korelasinya sendiri.

10. Test Set

Test set adalah bagian dataset yang kita tes untuk melihat keakuratannya, atau dengan kata lain melihat performanya.

1.3.2 Praktek

- Instalasi Library scikit dari anaconda, mencoba kompilasi dan uji coba ambil contoh kode dan lihat variabel explorer



Gambar 1.11 Instalasi Package Scikit Learn

Name	Type	Size	Value
digits	util.Bunch	5	Bunch object of sklearn.utils module
Iris	util.Bunch	6	Bunch object of sklearn.utils module

Gambar 1.12 Isi Variabel Explorer

- Mencoba loading an example dataset

```

1 %% Mencoba loading an example dataset
2 from sklearn import datasets # Digunakan Untuk Memanggil
   class datasets dari library sklearn
3 iris = datasets.load_iris() # Menggunakan contoh datasets
   iris
4 x = iris.data           # Menyimpan nilai data sets iris
   pada variabel x
5 y = iris.target         # Menyimpan nilai data label iris
   pada variabel y

```

3. Mencoba Learning dan predicting

```

1 #%%Mencoba Learning dan predicting
2 from sklearn.neighbors import KNeighborsClassifier #Digunakan
   Untuk Memanggil fungsi KNeighborsClassifier
3                                     # pada
4 class sklearn dan library sklearn
5 import numpy as np # memanggil library numpy dan dibuat alias
   np
6 knn=KNeighborsClassifier(n_neighbors=1) #membuat variabel kkn
   , dan memanggil fungsi KNeighborsClassifier
7                                     #dan mendefinisikan k
   -nya adalah 1
8 knn.fit(x,y)                      #Perhitungan
   matematika library kkn
9 a=np.array([1.0,2.0,3.0,4.0])      #Membuat Array
10 a=a.reshape(1,-1)                 #Mengubah Bentuk
    Array jadi 1 dimensi
11 hasil = knn.predict(a)            #Memanggil fungsi
    prediksi

```

4. Mencoba Model Persistence

```

1 #%% Model Persistense
2 from sklearn import svm # Digunakan untuk memanggil class svm
   dari library sklearn
3 from sklearn import datasets # Diguankan untuk class datasets
   dari library sklearn
4 clf = svm.SVC()           # membuat variabel clf , dan
   memanggil class svm dan fungsi SVC
5 X, y = datasets.load_iris(return_X_y=True) #Mengambil dataset
   iris dan mengembalikan nilainya.
6 clf.fit(X, y)             #Perhitungan nilai label
7
8 from joblib import dump, load #memanggil class dump dan load
   pada library joblib
9 dump(clf, '1174086.joblib') #Menyimpan model kedalam 1174086.
   joblib
10 hasil = load('1174086.joblib') #Memanggil model 1174086

```

5. Mencoba Conventions

```

1 #%% Conventions
2 import numpy as np # memanggil library numpy dan dibuat alias
   np
3 from sklearn import random_projection #Memanggil class
   random_projection pada library skean
4
5 rng = np.random.RandomState(0) #Membuat variabel rng , dan
   mendefisikan np, fungsi random dan attr RandomState
   kedalam variabel
6 X = rng.rand(10, 2000) # membuat variabel X, dan menentukan
   nilai random dari 10 – 2000

```

```

7 X = np.array(X, dtype='float32') #menyimpan hasil nilai
    random sebelumnya, kedalam array, dan menentukan
    typedatanya sebagai float32
8 X.dtype # Mengubah data tipe menjadi float64
9
10 transformer = random_projection.GaussianRandomProjection() #
    membuat variabel transformer, dan mendefinisikan
    classrandom_projection dan memanggil fungsi
    GaussianRandomProjection
11 X_new = transformer.fit_transform(X) # membuat variabel baru
    dan melakukan perhitungan label pada variabel X
12 X_new.dtype # Mengubah data tipe menjadi float64

```

1.3.3 Penanganan Error

1. ScreenShoot Error

`ImportError: cannot import name 'datasets' from 'sklearn'`

Gambar 1.13 Import Error

`ValueError: Expected 2D array, got 1D array instead:
array[1, 2, 3].
Reshape your data either using array.reshape(-1, 1) if your data has a single feature or
array.reshape(1, -1) if it contains a single sample.`

Gambar 1.14 Value Error

2. Tuliskan Kode Error dan Jenis Error

- Import Error
- Value Error

3. Cara Penanganan Error

- Import Error
Dengan Menginstall Library Yang Tidak Ditemukan
- Value Error
Mengubah Bentuk Arraynya, Menjadi 1 Dimensi

1.3.4 Bukti Tidak Plagiat



Gambar 1.15 Bukti Tidak Melakukan Plagiat Chapter 1

1.4 1174054 — Aulyardha Anindita

1.4.1 Teori

1. Definisi, Sejarah dan Perkembangan Kecerdasan Buatan

▪ Definisi Kecerdasan Buatan

Kecerdasan buatan adalah suatu kecerdasan yang didalamnya berisi suatu sistem yang biasa diatur dalam sebuah konteks ilmiah. Kecerdasan buatan juga bisa didefinisikan sebagai sebuah kecerdasan yang diciptakan dan dimasukkan kedalam suatu mesin computer agar dapat melakukan pekerjaan seperti yang dapat dilakukan oleh manusia. Ada beberapa macam bidang atau ilmu yang menggunakan kecerdasan buatan diantaranya adalah sistem pakar, permainan computer (game), logika fuzzy, jaringan saraf tiruan dan robotika.

Penelitian dalam AI mencakup pembuatan mesin dan suatu program computer untuk mengotomatisasikan tugas-tugas yang membutuhkan perilaku cerdas, seperti : pengendalian, perencanaan dan penjadwalan serta kemampuan untuk menjawab diagnose dan pertanyaan pelanggan serta pengenalan tulisan tangan. Suara dan wajah

▪ Sejarah Kecerdasan Buatan

- Pada tahun 1940 dan 1950 Artificial Intelligence merupakan suatu inovasi baru dalam bidang ilmu pengetahuan dimana pada tahun ini computer modern sudah ada

- Pada tahun 1950 awal, studi tentang “mesin berfikir” mempunyai berbagai nama seperti cybernetics, teori automata, dan pemrosesan informasi

- Pada tahun 1956, para ilmuwan jenius seperti Alan Turing, Norbert Wiener, Claude Shannon dan Warren McCullough bekerja secara independen di bidang cybernetics, matematika, algoritma dan teori jaringan. John McCarthy merupakan orang yang menciptakan istilah tersebut dan mendirikan laboratorium kecerdasan buatan di MIT dan Stanford

- Pada tahun 1956, McCarthy mendirikan Konferensi Dartmouth di Hanover, New Hampshire. Dia merupakan peneliti terkemuka dalam teori kompleksitas, simulasi Bahasa, dan hubungan antara keacakan dan pemikiran kreatif, jaringan saraf diundang. Sehingga Konferensi Dartmouth 1956 dianggap sebagai kelahiran Kecerdasan Buatan.

- Sejak saat itu, Kecerdasan Buatan telah hidup melalui decade kemuliaan dan cemohan yang dikenal dengan luas sebagai musim panas dan musim dingin Ai.

▪ Perkembangan Kecerdasan Buatan

Saat ini, teknologi Artificial Intelligence sangat ramai diperbincangkan

oleh masyarakat. Sudah banyak pekerjaan yang hilang karena adanya AI, seperti pekerjaan kasir, penjaga pintu tol, parkir, dan sebagainya. Hal ini terjadi karena AI lebih unggul dalam hal kinerja, fitur dan lain sebagainya. Walaupun masih ada beberapa aspek yang memang pekerja manusia masih unggul dibandingkan AI itu sendiri.

Berdasarkan survei yang dilakukan oleh Microsoft, hasilnya adalah 39 responden masih mempertimbangkan untuk menggunakan mobil tanpa pengemudi dan sebanyak 36 responden lainnya setuju bahwa robot atau AI dengan menggunakan software untuk beroperasi mampu meningkatkan produktivitas. Dari survei tersebut, dapat ditarik kesimpulan bahwa pengguna AI harus lebih bijaksana dalam pengembangan dan penggunaan dari AI sehingga tidak memiliki efek samping terhadap profuktifitas kerja dan keseharian sebagai pengguna dalam kehidupan sehari-hari.

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

▪ Supervised Learning

Supervised learning adalah suatu tugas pengumpulan data yang berfungsi untuk menyimpulkan fungsi dari data pelatihan yang berlabel. Didalam Supervised Learning, setiap contoh merupakan pasangan yang terdiri dari objek input dan nilai output yang diinginkan. Algoritma pembelajaran yang diawasi berupa menganalisis data pelatihan dan menghasilkan fungsi yang disimpulkan yang digunakan untuk memetakan contoh baru.

Supervised Learning adalah suatu pendekatan dimana sudah terdapat data yang dilatih selain itu juga sudah memiliki variable yang ditargetkan sehingga tujuan dari pendekatan tersebut adalah mengelompokkan suatu data ke data yang sudah ada. Supervised learning sendiri menyediakan algoritma pembelajaran dengan jumlah yang diketahui untuk mendukung penilaian dimasa depan. Supervised learning sebagian besar memiliki kaitan dengan AI dengan menggunakan model pembelajaran generatif. Data pelatihan untuk pembelajaran yang diawasi mencakup beberapa contoh dengan subjek input yang berpasangan dan output yang diinginkan.

Modul supervised learning mempunyai beberapa keunggulan dibandingkan pendekatan tanpa pengawasan, tapi mereka juga memiliki keterbatasan. System lebih cenderung membuat penilaian bahwa manusia dapat berhubungan, misalnya manusia mempunyai dasar untuk keputusan. Tapi, dalam kasus tersebut yang menggunakan metode berbasis pengambilan, supervised learning mengalami kesulitan dalam menangani suatu informasi baru.

▪ Klasifikasi

Klasifikasi merupakan pembagian menurut kelas-kelas. Menurut ilmu

pengetahuan, klasifikasi adalah suatu proses pengelompokan benda berdasarkan ciri-ciri persamaan dan perbedaan. Dalam pembelajaran mesin dan statistic, klasifikasi merupakan suatu pendekatan pembelajaran yang diawasi dimana program computer tersebut belajar dari input data yang diberikan kepadanya lalu menggunakan pembelajaran tersebut untuk mengklasifikasikan pengamatan baru. Kumpulan data tersebut mungkin hanya bersifat dua kelas atau mungkin juga multi-kelas.

- **Regresi**

Regresi adalah suatu metode analisis statistik yang digunakan untuk melihat pengaruh antara dua atau lebih variable. Regresi sendiri membahas masalah ketika variable output yaitu nilai ril atau berkelanjutan seperti gaji atau berat. Banyak model yang dapat digunakan, yang paling sederhana adalah regresi linear.

- **Unsupervised Learning**

Unsupervised learning berbeda dengan supervised learning, perbedaannya yaitu unsupervised learning tidak memiliki data pelatihan, sehingga data dapat dikelompokkan menjadi dua atau 3 begitupun seterusnya. Unsupervised learning adalah suatu pelatihan algoritma kecerdasan buatan (AI) menggunakan beberapa informasi yang tidak diklasifikasikan atau diberi label dan memungkinkan algoritma untuk bertindak atas informasi tersebut. System AI disini dapat dikelompokkan berdasarkan informasi yang tidak disortir berdasarkan persamaan dan perbedaan meskipun tidak ada kategori yang disediakan. System AI disajikan dengan data yang tidak berlabel, tidak terkategori dan algoritma system bekerja pada data tanpa pelatihan sebelumnya sehingga outputnya tergantung pada algoritma kode.

- **Data Set**

Data set adalah suatu objek yang merepresentasikan data dan memiliki relasi yang ada di dalam memory. Struktur data set mirip dengan data yang ada didatabase, namun bedanya data set berisi koleksi dari data table dan data relation. Untuk mendapatkan data yang tepat, berarti mengumpulkan atau mengidentifikasi data yang berkorelasi dengan hasil yang ingin anda prediksi.

- **Training Set**

Training set adalah salah satu set yang biasa digunakan oleh algoritma klasifikasi. Seperti decision tree, bayesian, neural network, dll. Mereka dapat digunakan untuk membentuk model classifier, dalam menjalankan pelatihan yang diatur melalui jaringan saraf yang mengajarkan pada net dengan cara menimbang berbagai fitur, menyeuaikan koefisien berdasarkan kemungkinan mereka meminimalkan kesalahan. Kofiesen tersebut juga dikenal sebagai parameter.

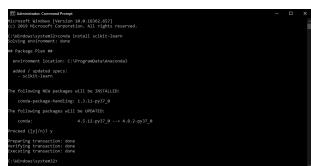
- **Testing Set**

Testing set adalah salah satu set yang digunakan untuk mengukur

sejauh mana classifier berhasil melakukan klasifikasi dengan benar. Hal ini berfungsi sebagai materai persetujuan tapi tak digunakan sampai akhir. Setelah melatih dan mengoptimalkan data, kita dapat melakukan pengujian sarat terhadap pengambilan sampel aca. Dan hasilnya harus memvalidasi bahwa jaring data tersebut secara akurat mengenali gambar atau mengenali setidaknya (x) dari jumlah tersebut.

1.4.2 Praktek

1. Instalasi library scikit dari anaconda



Gambar 1.16 Instalasi Package Scikit Learn

Name	Type	Size	Value
digits	util.Bunch	5	Bunch object of sklearn.utils module
iris	util.Bunch	6	Bunch object of sklearn.utils module

Gambar 1.17 Isi Variabel Explorer

2. Mencoba Loading an example dataset

```

1 #%% Mencoba loading an example dataset
2 from sklearn import datasets # Digunakan Untuk Memanggil
   class datasets dari library sklearn
3 iris = datasets.load_iris() # Menggunakan contoh datasets
   iris
4 x = iris.data           # Menyimpan nilai data sets iris
   pada variabel x
5 y = iris.target         # Menyimpan nilai data label iris
   pada variabel y

```

3. Mencoba Learning and predicting

```

1 #%%Mencoba Learning dan predicting
2 from sklearn.neighbors import KNeighborsClassifier #Digunakan
   Untuk Memanggil fungsi KNeighborsClassifier
3                                     # pada
   class sklearn dan library sklearn
4 import numpy as np # memanggil library numpy dan dibuat alias
   np

```

```

5 knn=KNeighborsClassifier(n_neighbors=1) #membuat variabel kkn
   , dan memanggil fungsi KNeighborsClassifier
6                                         #dan mendefinisikan k
7 knn.fit(x,y)                         #Perhitungan
8 matematika library kkn
9 a=np.array([1.0,2.0,3.0,4.0])
10 a = a.reshape(1,-1)
    Array jadi 1 dimensi
11 hasil = knn.predict(a)                #Memanggil fungsi
    prediksi
12 print(hasil)                        #menampilkan hasil
    prediksi

```

4. Mencoba Model persistence

```

1 #%% Model Persistense
2 from sklearn import svm # Digunakan untuk memanggil class svm
   dari library sklearn
3 from sklearn import datasets # Diguangkan untuk class datasets
   dari library sklearn
4 clf = svm.SVC()           # membuat variabel clf , dan
   memanggil class svm dan fungsi SVC
5 X, y = datasets.load_iris(return_X_y=True) #Mengambil dataset
   iris dan mengembalikan nilainya.
6 clf.fit(X, y)             #Perhitungan nilai label
7
8 from joblib import dump, load #memanggil class dump dan load
   pada library joblib
9 dump(clf, '1174054.joblib') #Menyimpan model kedalam 1174054.
   joblib
10 hasil = load('1174054.joblib') #Memanggil model 1174054
11 print(hasil) # Menampilkan Model yang dipanggil sebelumnya

```

5. Mencoba Conventions

```

1 #%% Conventions
2 import numpy as np # memanggil library numpy dan dibuat alias
   np
3 from sklearn import random_projection #Memanggil class
   random_projection pada library sklean
4
5 rng = np.random.RandomState(0) #Membuat variabel rng , dan
   mendefinisikan np, fungsi random dan attr RandomState
   kedalam variabel
6 X = rng.rand(10, 2000) # membuat variabel X, dan menentukan
   nilai random dari 10 – 2000
7 X = np.array(X, dtype='float32') #menyimpan hasil nilai
   random sebelumnya, kedalam array , dan menentukan
   typedatanya sebagai float32
8 X.dtype # Mengubah data tipe menjadi float64
9

```

```

10 transformer = random_projection.GaussianRandomProjection() # membuat variabel transformer, dan mendefinisikan class random_projection dan memanggil fungsi GaussianRandomProjection
11 X_new = transformer.fit_transform(X) # membuat variabel baru dan melakukan perhitungan label pada variabel X
12 X_new.dtype # Mengubah data tipe menjadi float64
13 print(X_new) # Menampilkan isi variabel X_new

```

1.4.3 Penanganan Error

1. ScreenShoot Error

```

File "D:/Mata Kuliah/Tingkat 3/Semester 6/Kecerdasan Buatan/Chapter 1/1174054.py", line 33, in <module>
    from joblib import dump, load #mengambil class dump dan load
    ModuleNotFoundError: No module named 'joblib'

```

Gambar 1.18 Module Not Found Error

```

File "D:/Mata Kuliah/Semester 6/Kecerdasan Buatan/DAIkecilLearn.py", line 8, in <module>
    from sklearn import datasets
ImportError: cannot import name 'datasets' from 'sklearn' (D:\Kuliah\Semester 6\kecerdasan buatan\DAIkecilLearn.py)

```

Gambar 1.19 Import Error

2. Tuliskan Kode Error dan Jenis Error

- Module Not Found Error
- Import Error

3. Cara Penanganan Error

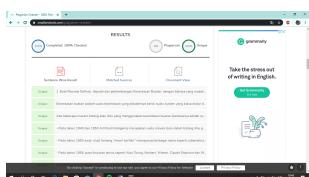
- Module Not Found Error

Dengan memperbaiki penulisan atau kesalahan dalam penulisan kode atau melakukan install package atau modul yang belum terinstal

- Import Error

Dengan Menginstall Library Yang Tidak Ditemukan

1.4.4 Bukti Tidak Plagiat



Gambar 1.20 Bukti Plagiarisme

1.4.5 Link Youtube

<https://youtu.be/gI9Q60DzEfI>

1.5 Ainul Filiani 1174073

1.5.1 Pengertian Kecerdasan Buatan

Kecerdasan buatan adalah salah satu cabang ilmu pengetahuan yang berhubungan dengan mesin untuk memecahkan persoalan yang rumit dengan cara yang lebih manusiawi. Hal ini biasanya dilakukan dengan mengikuti karakteristik dan analogi berpikir dari kecerdasan atau inteligance manusia, dan menerapkan sebagai algoritma yang dikenal oleh komputer. Dengan suatu pendekatan yang kurang lebih fleksibilitas dan efisien dapat diambil tergantung keperluan yang mempengaruhi bagaimana wujud dari prilaku kecerdasan buatan. AI biasanya dihubungkan dengan ilmu komputer, akan tetapi juga terkait erat dengan bidang lain seperti matematika, Psikologi, Pengamatan, Biolog, filosofi, dan lainnya

1.5.2 Sejarah Kecerdasan Buatan

Kecerdasan buatan merupakan bidang ilmu komputer yang sangat penting di era kini dan masa yang akan datang untuk mewujudkan sistem komputer yang cerdas. Bidang ini telah berkembang sangat pesat di 20 tahun terakhir seiring dengan kebutuhan perangkat cerdas pada industry dan rumah tangga. Kata Intelligence berasal dari bahasa latin "intelligo" yang berarti "saya paham". Berarti dasar dari intelligence adalah kemampuan untuk memahami dan melakukan aksi. nyatanya, bidang Kecerdasan Buatan atau disingkat dengan AI, berawal dari kemunculan komputer sekitar tahun 1940-an, sedangkan perkembangan sejarah dapat ditelusuri sejak zaman Mesir kuno. Pada saat ini, perhatian mendesak diberikan pada kemampuan komputer untuk melakukan hal-hal yang dapat dilakukan manusia. Dalam hal ini, komputer ini dapat meningkatkan kemampuan kecerdasan dan kecerdasan manusia. Pada awal abad ke-17, René berbicara tentang tubuh binatang yang tidak meminta apa pun selain mesin yang rumit. Blaise Pascal membuat mesin hitung digital mekanis pertama pada tahun 1642. Pada 19, Charles Babbage dan Ada Lovelace bekerja pada mesin hitung mekanis yang dapat diprogram. Bertrand Russell dan Alfred Whitehead North menerbitkan Principia Mathematica, yang merombak logistik formal. Warren McCulloch dan Walter Pitts menerbitkan "Kalkulus Logika Gagasan yang Menjaga Aktivitas" pada tahun 1943 yang membentuk dasar bagi jaringan saraf. 1950-an adalah periode upaya aktif dalam AI. program permainan catur yang ditulis oleh Dietrich Prinz. John McCarthy menciptakan istilah "kecerdasan buatan" pada konferensi pertama yang menjadi dasar perjanjian itu, pada tahun 1956. Dia

juga menemukan bahasa pemrograman Lisp. Alan Turing memperkenalkan "tes Turing" sebagai cara untuk mengoperasionalkan tes kecerdasan cerdas.

1.5.3 Perkembangan dan Penggunaan Kecerdasan

Menurut studi Harvard Business Review dan ICM Unlimited pada tahun 2016, perusahaan besar memberikan kompensasi 10 persen lebih tinggi untuk setiap karyawan, Terrelong melanjutkan pengembangan Artificial Intelligence (AI) tidak hanya untuk membuat gambar atau video palsu lebih mudah, tetapi juga membuatnya sulit untuk membuktikannya. Meskipun pada saat ini, upaya untuk membuat dan mendistribusikan konten hoax, alias hoaks, masih dapat diatasi, tetapi berhasil, tantangan yang dihadapi semakin sulit. Selain itu, AI memungkinkan pembuatan gambar, video, atau audio palsu dari bahan yang relatif minim. Moody's, yang harus disetujui, membuktikan upaya itu akan semakin menantang dan membutuhkan teknik forensik yang lebih canggih. Pada Mei 2019, para peneliti di Samsung AI Center dan Institut Sains dan Teknologi Skolkovo di Moskow, Rusia menunjukkan bahwa mereka dapat membuat tayangan video yang menampilkan masing-masing individu. Video ini sangat realistik tetapi sebenarnya palsu, dibuat menggunakan model pembelajaran tertentu yang disebut Generative Adversarial Network (GAN). Hasil dari proses GAN disebut deepfakes karena mereka menggunakan teknik pembelajaran yang mendalam untuk membuat konten palsu. Untuk jangka pendek, perusahaan diharapkan untuk terus memainkan media sosial dan situs untuk melihat pentingnya disinformasi dan meminta mereka yang bertanggung jawab untuk media sosial dan situs terkait untuk mengunduh konten. Terrelong menambahkan langkah lain yang bisa diam-bil untuk merilis materi resmi untuk melawan konten palsu."Perlawanan terhadap konten palsu membutuhkan kombinasi teknologi dan pendidikan,".

1.6 resume mengenai definisi supervised learning, klarifikasi, regresi, dan un-supervised learning. Data Set, training set dan testing set

1.6.1 Supervised Learning

Supervised Learning adalah tugas mengumpulkan data untuk melengkapi fungsi data pelatihan yang diberi label. Data pelatihan terdiri dari contoh pelatihan. Dalam pembelajaran terawasi, setiap contoh adalah pasangan yang terdiri dari objek input (biasanya vektor) dan nilai output dingin (juga disebut sinyal pengawasan super). Algoritma pembelajaran yang diawasi menganalisis data pelatihan dan menghasilkan fungsi yang lengkap, yang dapat digunakan untuk memetakan contoh-contoh baru. Skenario akan memungkinkan algoritma menentukan label kelas dengan benar untuk instance yang tidak terlihat. Ini membutuhkan algoritma pembelajaran untuk menggeneralisasi

data pelatihan sehingga tidak muncul dengan cara yang "masuk akal". Pembelajaran terawasi semakin dekat di mana ada pelatihan praktis selain dapat bervariasi yang berarti tujuannya adalah di mana mengelompokkan data ke dalam database yang ada. Pembelajaran terawasi menyediakan jumlah pembelajaran yang direkomendasikan untuk mendukung penilaian di masa depan. Obrolan, program mengemudi mandiri, pengenalan wajah, tatap muka dan robot adalah beberapa sistem yang dapat menggunakan pembelajaran yang diawasi atau tidak diawasi. Pembelajaran terbimbing sebagian besar terkait dengan AI berdasarkan pengambilan mereka juga mungkin diperlukan menggunakan model pembelajaran generatif. Pelatihan data untuk pembelajaran dimulai dengan mendiskusikan contoh-contoh dengan subjek input berpasangan dan output yang diinginkan (juga disebut sebagai sinyal pengawasan). Dalam pembelajaran yang diawasi untuk pemrosesan gambar, misalnya sistem AI dapat lengkap dengan gambar mengemudi yang berlabel dalam kategori mobil dan truk. Setelah jumlah yang memadai, sistem harus dapat membedakan antara dan mengklasifikasikan gambar yang tidak berlabel, di mana waktu pelatihan dapat diselesaikan secara penuh. Model Pembelajaran Terpandu memiliki beberapa keunggulan dibandingkan pengawasan, tetapi mereka juga memiliki keterbatasan. Sistem lebih cenderung membuat penilaian bahwa hak asasi manusia dapat dihubungkan, misalnya karena manusia telah memberikan dasar untuk pengambilan keputusan. Namun, dalam hal metode berbasis pengambilan, Supervised Learning menghilangkan kesulitan dalam menangani informasi baru. Jika sistem dikategorikan untuk mobil dan truk, maka sepeda disediakan, misalnya, harus dikelompokkan dalam satu kategori atau yang lain. Namun. Jika sistem AI generatif, mungkin tidak tahu apa itu sepeda tetapi akan dapat mengenalinya sebagai milik kategori yang terpisah.

1.6.2 Klasifikasi

Klasifikasi adalah pembagian hal sesuai dengan kelas (kelas). Menurut Science, klasifikasi adalah proses pengelompokan materi berdasarkan karakteristik dan perbedaan yang sama. Dalam masalah klasifikasi, kami mencoba memprediksi sejumlah nilai yang terpisah. Label (y) Umumnya datang dalam bentuk kategorikal dan mewakili sejumlah kelas. Dalam pembelajaran statistik dan pembelajaran mesin statistik, klasifikasi adalah pembelajaran yang dimulai ketika sebuah program komputer belajar dari input data yang disediakan untuk mendukung dan kemudian menggunakan pembelajaran ini untuk mengklasifikasikan pembelajaran baru. Pengumpulan data ini mungkin hanya dua kelas (seperti mengidentifikasi apakah orang ini laki-laki atau perempuan atau orang itu adalah spam atau bukan-spam) atau mungkin juga multi-kelas. Beberapa contoh masalah klasifikasi adalah: pengenalan ucapan, pengenalan tulisan tangan, metrik identifikasi, klasifikasi dokumen dll.

1.6.3 Regresi

Regresi adalah metode analisis statistik yang digunakan untuk melihat perbedaan antara dua atau lebih variabel. Regresi sedang membahas masalah kompilasi, variabel output adalah nilai nyata atau dipertahankan, seperti "gaji" atau "berat". Banyak model yang berbeda dapat digunakan untuk makan, cara paling sederhana adalah linearitas linear. Itu mencoba untuk mencocokkan data dengan pesawat-hyper terbaik yang melewati titik.

1.6.4 unsupervised learning

Belajar tanpa pengawasan berbeda dari Belajar dengan Supervisi. Perbedaannya adalah bahwa pembelajaran tanpa pengawasan tidak memiliki data pelatihan, jadi dari data yang tersedia kami mengelompokkan data menjadi 2 atau 3 bagian dan seterusnya. Unsupervised Learning adalah pelatihan dalam algoritma kecerdasan buatan (AI) menggunakan informasi yang tidak diklasifikasi atau diberi label dan menyediakan algoritma untuk memperbaiki informasi yang diberikan tanpa bimbingan. Dalam Unattended Learning, sistem AI dapat mengklasifikasikan informasi yang tidak diurutkan berdasarkan ekuitas dan perbedaan dalam kategori mendadak yang disediakan. Dalam Supervised Learning Learning, sistem AI disajikan dengan sistem wajib yang tidak diberi label, tidak dikategorikan dan algoritma bekerja pada data tanpa pelatihan sebelumnya. Outputnya tergantung pada algoritma kode. Menyerahkan sistem untuk Belajar Tanpa Pengawasan adalah salah satu cara untuk menerima AI. Algoritma Pembelajaran tanpa pengawasan dapat melakukan tugas yang lebih kompleks daripada sistem pembelajaran yang diawasi. Namun, pembelajaran tanpa pengawasan dapat lebih tidak konsisten dengan model alternatif. Sementara Supervised Learning Mungkin, misalnya, mencari sendiri dengan memilih kucing dari anjing, ia juga dapat menambahkan kategori yang tidak diinginkan dari yang tidak diinginkan untuk ditingkatkan menjadi ras yang tidak biasa, membuat pesanan diperlukan.

1.6.5 Data Set

Dataset adalah objek yang mewakili data dan hubungan dalam memori. Strukturnya mirip dengan basis data basis data, tetapi hanya kumpulan data yang dikumpulkan dari catatan dan latar belakang yang diaktifkan. dapatkan persetujuan yang tepat untuk mengumpulkan atau mengidentifikasi data yang berkorelasi dengan hasil yang ingin Anda hasilkan; yaitu data yang berisi sinyal tentang acara yang Anda sukai. Data harus disinkronkan dengan masalah yang Anda coba selesaikan. Gambar kucing bukan kompilasi yang sangat berguna. Anda sedang membangun sistem identifikasi wajah. Memodifikasi data yang selaras dengan masalah yang ingin Anda selesaikan harus dilakukan oleh para ahli data. Jika Anda tidak memiliki data yang benar, maka upaya Anda untuk membuat solusi AI harus kembali ke in-

stalasi data. Format ujung kanan untuk belajar secara umum adalah array tensor, atau multi-dimensional. Jadi pipa data yang dibangun untuk pembelajaran dibangun secara umum untuk mengubah semua gambar, video, suara, suara, teks atau deret waktu menjadi vektor dan tensor yang dapat digunakan operasi aljabar linier. Data yang diperlukan perlu dinormalisasi, distandarisasi dan dikembalikan untuk meningkatkan kegunaannya, dan semua ini adalah langkah-langkah dalam pembelajaran mesin ETC. Deeplearning4j menawarkan alat ETV Data Vec untuk melakukan tugas memfasilitasi data. Pembelajaran yang mendalam, dan pembelajaran mesin yang lebih umum, membutuhkan pelatihan yang baik agar dapat bekerja dengan baik. Mengumpulkan dan membangun satu set badan pelatihan yang cukup besar dari data yang diketahui membutuhkan waktu dan pengetahuan khusus tentang pengetahuan dan cara untuk mengumpulkan informasi yang relevan. Perangkat pelatihan bertindak sebagai patokan terhadap mana jaring pembelajaran dalam pengeboran. Itulah yang mereka perbarui untuk direkonstruksi sebelum mereka merilis data yang belum pernah dilihat sebelumnya. Pada saat ini, manusia memiliki pengetahuan luas tentang mengidentifikasi instrumen yang tepat dan mengubahnya menjadi representasi numerik yang dapat dipahami oleh algoritma pembelajaran dalam, tensor. Membangun set pelatihan, dalam arti tertentu, pra-pelatihan. Kumpulan pelatihan yang membutuhkan banyak waktu atau keahlian yang dapat membantu dalam dunia data dan pemecahan masalah. Sifat keahlian terbesar Anda dalam memberi tahu algoritma Anda apa yang penting bagi Anda adalah memilih apa yang Anda masukkan dalam kursus pelatihan Anda. Ini melibatkan menceritakan kisah melalui data awal yang Anda pilih untuk memandu proses pembelajaran mendalam Anda dengan mengekstraksi fitur-fitur penting, baik dalam pengaturan pelatihan dan data yang ingin Anda buat untuk dipelajari. Agar pelatihan ini bermanfaat, Anda harus memecahkan masalah yang Anda selesaikan; yaitu, apa yang Anda inginkan agar sesuai dengan pembelajaran Anda, di mana hasil yang ingin Anda prediksi.

1.6.6 Training Set

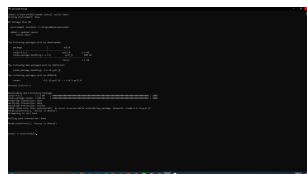
Set Pelatihan adalah set yang digunakan oleh algoritma klasifikasi. Dapat dicontohkan oleh: decisiontree, bayesian, neural network dll. Semuanya dapat digunakan untuk membuat model kelas. Terkait dengan pelatihan yang mengatur melalui jaringan saraf di internet bagaimana menimbang berbagai fitur, sesuaikan koefisien sesuai dengan apa yang mereka tingkatkan dalam hasil Anda. Koefisien, juga dikenal sebagai parameter, akan terkandung dalam sensor dan bersama-sama mereka disebut model, model data karena mereka menyandikan latihan yang mereka praktekkan.

1.6.7 testing Set

Tes ini digunakan untuk mengukur sejauh mana classifil berhasil mengklasifikasi dengan benar. Ini digunakan sebagai meterai persetujuan, dan Anda tidak dapat digunakan sampai akhir. Setelah Anda melatih dan mengoptimalkan data Anda, Anda menguji jaringan saraf Anda untuk mengambil sampel acak akhir ini. Hasilnya harus memvalidasi gambar bersih Anda, atau gambar mengenali [x] dari nomor itu. Jika Anda tidak mendapatkan prediksi yang akurat, kembalilah ke set pelatihan Anda, lihat mitra Anda yang Anda gunakan untuk mengelola jaringan Anda, dan kualitas data Anda dan lihat teknik pra-pemanfaatan yang dapat Anda gunakan.

1.6.8 Instalasi

- Instalasi Library scikit dari anaconda, mencoba kompilasi dan uji coba ambil contoh kode dan lihat variabel explorer



Gambar 1.21 Instalasi Package Scikit Learn

Name	Type	Size	Value
digits	util.Bunch	5	Bunch object of sklearn.utils module
iris	util.Bunch	6	Bunch object of sklearn.utils module

Gambar 1.22 Isi Variabel Explorer

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

```

1 #%% Mencoba loading an example dataset
2 from sklearn import datasets # Digunakan Untuk Memanggil
   class datasets dari library sklearn
3 iris = datasets.load_iris() # Menggunakan contoh datasets
   iris
4 x = iris.data           # Menyimpan nilai data sets iris
   pada variabel x
5 y = iris.target          # Menyimpan nilai data label iris
   pada variabel y

```

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

```

1 %%Mencoba Learning dan predicting
2 from sklearn.neighbors import KNeighborsClassifier #Digunakan Untuk Memanggil fungsi KNeighborsClassifier
3                                         # pada
4 class sklearn dan library sklearn
5 import numpy as np # memanggil library numpy dan dibuat alias np
6 knn=KNeighborsClassifier(n_neighbors=1) #membuat variabel kkn , dan memanggil fungsi KNeighborsClassifier
7                                         #dan mendefinisikan k -nya adalah 1
8 knn.fit(x,y)                         #Perhitungan matematika library kkn
9 a=np.array([1.0,2.0,3.0,4.0])         #Membuat Array
10 a = a.reshape(1,-1)                  #Mengubah Bentuk
11     Array jadi 1 dimensi
12 hasil = knn.predict(a)               #Memanggil fungsi prediksi

```

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

```

1 %% Model Persistense
2 from sklearn import svm # Digunakan untuk memanggil class svm dari library sklearn
3 from sklearn import datasets # Digunakan untuk class datasets dari library sklearn
4 clf = svm.SVC()           # membuat variabel clf , dan memanggil class svm dan fungsi SVC
5 X, y = datasets.load_iris(return_X_y=True) #Mengambil dataset iris dan mengembalikan nilainya .
6 clf.fit(X, y)             #Perhitungan nilai label
7
8 from joblib import dump, load #memanggil class dump dan load pada library joblib
9 dump(clf, '1174073.joblib') #Menyimpan model kedalam 1174069. joblib
10 hasil = load('1174073.joblib') #Memanggil model 1174069

```

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

```

1 %% Conventions
2 import numpy as np # memanggil library numpy dan dibuat alias np
3 from sklearn import random_projection #Memanggil class random_projection pada library sklearn
4
5 rng = np.random.RandomState(0) #Membuat variabel rng , dan mendefinisikan np, fungsi random dan attr RandomState kedalam variabel
6 X = rng.rand(10, 2000) # membuat variabel X, dan menentukan nilai random dari 10 – 2000

```

```

7 X = np.array(X, dtype='float32') #menyimpan hasil nilai
     random sebelumnya, kedalam array, dan menentukan
     typedatanya sebagai float32
8 X.dtype # Mengubah data tipe menjadi float64
9
10 transformer = random_projection.GaussianRandomProjection() #
      membuat variabel transformer, dan mendefinisikan
      classrandom_projection dan memanggil fungsi
      GaussianRandomProjection
11 X_new = transformer.fit_transform(X) # membuat variabel baru
      dan melakukan perhitungan label pada variabel X
12 X_new.dtype # Mengubah data tipe menjadi float64

```

1.6.9 Penanganan Error

1. ScreenShoot Error

```

file "D:\Kuliah\Semester 6\Kecerdasan Buatan\Dataset\sklearn.py", line 8, in <module>
    from sklearn import datasets
ImportError: cannot import name 'datasets' from 'sklearn' (D:\Kuliah\Semester
6\Kecerdasan Buatan\Dataset\sklearn.py)

```

Gambar 1.23 Import Error

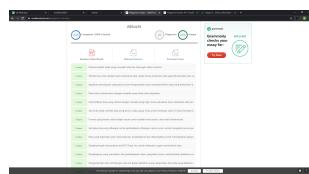
2. Tuliskan Kode Error dan Jenis Error

- Import Error

3. Cara Penangan Error

- Import Error
Dengan Menginstall Library Yang Tidak Ditemukan

1.6.10 Bukti Tidak Plagiat



Gambar 1.24 Bukti Tidak Melakukan Plagiat Chapter 1

1.6.11 Link Youtube

1.7 Chandra Kirana Poetra (1174079)

1.7.1 Teori

1. Definisi Kecerdasan buatan

Dalam bidang komputer, Artificial Intelligence (AI), atau biasa disebut juga sebagai Machine Intelligence merupakan bentuk dari representasi kecerdasan yang dilakukan oleh mesin, hampir mirip seperti bagaimana manusia melakukan kecerdasan. Beberapa sumber mendefinisikan bahwa bidang yang mempelajari suatu agen kecerdasan merupakan suatu alat yang mengenali lingkungan sekitarnya dan mencoba untuk membuat kesimpulan untuk memaksimalkan kemungkinan tingkat keberhasilan dari pencapaian yang ingin dituju.

2. Sejarah dan Perkembangan Kecerdasan Buatan

- Pada tahun 1943, pekerjaan pertama yang dikenal sebagai AI telah dilakukan oleh Warren McCulloch dan juga Walter Pitts yang dinamakan sebagai artificial neurons
- Pada tahun 1955, Allen Newell dan Herbert A. Simon membuat program kecerdasan buatan pertama yang dinamakan Logic Theorist
- Pada tahun 1972, robot pertama dibuat di Jepang dengan nama Wabot-1 dengan kecerdasan buatan
- Pada tahun 1980, muncul bidang baru dari kecerdasan buatan yaitu Expert System yang membantu dalam pemberian keputusan
- Tahun 1997, IBM Deep Blue mengalahkan juara catur dunia Gary Kasparov dan menjadi komputer pertama yang mengalahkannya
- Tahun 2006, perusahaan sudah mulai menerapkan kecerdasan buatan pada produknya seperti Netflix dan Twitter.
- Tahun 2018, Project Debater dari IBM melakukn debat tentang topik yang kompleks dan berakhir dengan hasil memuaskan

3. Definisi Supervised Learning

Supervised Learning adalah proses untuk melatih mesin secara input dan output melalui contoh nyata secara langsung

4. Klasifikasi Supervised Learning

- Support Vector Machines
- linear regression
- logistic regression
- naive Bayes

- linear discriminant analysis
- decision trees
- k-nearest neighbor algorithm
- Neural Networks (Multilayer perceptron)
- Similarity learning

5. Regresi dan Unsupervised Learning

Regresi adalah suatu proses statistikal yang mengestimasi hubungan antara variable satu dengan variable yang lainnya.

Unsupervised Learning adalah bentuk dari machine learning yang mencari bentuk atau hubungan dari data set yang tidak mempunyai label dengan bantuan yang minimal dari manusia.

6. Dataset

Dataset adalah koleksi suatu data

7. Training Set

Training Set merupakan data yang digunakan untuk keperluan pembelajaran yang biasanya digunakan oleh machine learning

8. Testing Set

Testing set adalah data yang real yang digunakan untuk melatih machine learning

1.7.2 Instalasi

1. Instalasi Library scikit dari a naconda, mencoba kompilasi dan uji coba ambil contoh kode dan lihat variabel explorer

```

$ conda install -c anaconda scikit-learn
Collecting package metadata (current_repodata.json): done
Solving environment: done
++ Package Plan ++
  environment location: C:\Users\user\Anaconda3
  added / updated specs:
    - scikit-learn

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

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

Proceed ([y]/n)? y

downloading and extracting Packages:
  conda-4.8.2           2.8 MB |████████████████████████████████| 100%
  verifying transaction: done
  writing transaction file: done
  installing transaction file: done
[2020-01-10 11:14:11] CONDA-GLOBAL-MODIFY=1

```

Gambar 1.25 Instalasi Package Scikit Learn

Name	Type	Size	Value
digits	utils.Bunch	5	Bunch object of sklearn.utils module
iris	utils.Bunch	6	Bunch object of sklearn.utils module

Gambar 1.26 Isi Variabel Explorer

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

```

1 %%Loading an example dataset
2 from sklearn import datasets # Load library dataset
3 iris = datasets.load_iris()
4 # variable iris diisi dengan contoh data
5 a = iris.data # Menyimpan value data ke variable A
6 b = iris.target # Menyimpan value data ke variable B

```

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

```

1 %% Learning dan predicting
2 from sklearn.neighbors import KNeighborsClassifier
3 #Load library
4 import numpy as np
5 #load library
6
7 knn = KNeighborsClassifier(n_neighbors=1)
8 #mendefinisikan variabel bernama kkn, dan memanggil fungsi
KNeighborsClassifier
9 # dan memberikan value 1
10 knn.fit(a,b) # perhitungan library knn
11
12 x = np.array([1.0 ,2.0 ,3.0 ,4.0])
13 # membuat array
14 x = x.reshape(1,-1)
15 #Convert array menjadi 1 dimensi
16
17 hasil = knn.predict(x)
18 #Memanggil fungsi predict dari KNN
19 print(hasil)
20 #menampilkan value dari variable hasil

```

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

```

1 %% Model Persistense
2 from sklearn import svm
3 # Load library
4 from sklearn import datasets
5 # Load Library
6 clf = svm.SVC()
7 # mendefinisikan variabel clf, dan memanggil fungsi SVC dari
class svm
8 a, b = datasets.load_iris(return_X_y=True)
9 #Variable a dan b diisi dengan dataset iris dan mengembalikan
nilainya.
10 clf.fit(a, b)
11 #memanggil fungsi fit dari clf
12

```

```

13 from joblib import dump, load
14 #Load library
15 dump(clf, '1174079.joblib')
16 #Menyimpan model kedalam 1174079.joblib
17 hasil = load('1174079.joblib')
18 #memuat model 1174079
19 print(hasil) # Menampilkan Hasil

```

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

```

1 %% Conventions
2 import numpy as np
3 # Load Library
4 from sklearn import random_projection
5 #Load class random_projection dari library sklearn
6
7 rng = np.random.RandomState(0)
8 #Membuat variabel rng, dan mendefinisikan np, fungsi random dan
     attr RandomState kedalam variabel
9 X = rng.rand(10, 2000)
10 # membuat variabel X, dan menentukan nilai random dari 10 –
      2000
11 X = np.array(X, dtype='float32')
12 #menyimpan hasil nilai random sebelumnya, kedalam array , dan
      menentukan typedatanya sebagai float32
13 X.dtype
14 # Mengubah data tipe menjadi float64
15
16 transformer = random_projection.GaussianRandomProjection()
17 #membuat variabel transformer, dan mendefinisikan
      classrandom_projection dan memanggil fungsi
      GaussianRandomProjection
18 X_new = transformer.fit_transform(X)
19 # membuat variabel baru dan melakukan perhitungan label pada
      variabel X
20 X_new.dtype
21 # Mengubah data tipe menjadi float64
22 print(X_new)
23 # Menampilkan isi variabel X_new

```

1.7.3 Penanganan Error

1. ScreenShoot Error

```

-----  

exec(compile(f.read(), filename,  

'exec'), namespace)  

File "F:/Poltekpos/D4 TI 3C/Semester 6/  

Kecerdasan Buatan/Github/Upload 1 Naret  

2020/src/src/1174079/1174079.py", line 51, in  

<module>  

    import numpy as np  

ModuleNotFoundError: No module named  

'numpy'  

To run this file in Python 3, you must type:  


```

Gambar 1.27 No Module Named Numpya

2. Tuliskan Kode Error dan Jenis Error

- ModuleNotFoundError

3. Cara Penangan Error

- ModuleNotFoundError

Mengecek Typo dan menulis kembali library yang akan diimport

1.7.4 Bukti Tidak Plagiat



Gambar 1.28 Bukti Tidak Melakukan Plagiat Chapter 1

1.7.5 Link Youtube

<https://youtu.be/nPua0lRXjO8>

1.8 D. Irga B. Naufal Fakhri

1.8.1 Teori

1.8.1.1 Definisi Kecerdasan Buatan Kecerdasan Buatan atau yang sering disebut AI (Artificial Intelligence) merupakan suatu cabang didalam bisnis sains dan komputer sains yang didalamnya membahas tentang bagaimana caranya untuk membuat sebuah komputer dengan kemampuan atau ke pintaran layaknya atau mirip dengan yang dimiliki manusia. Contohnya, bagaimana komputer bisa berkomunikasi dengan pengguna baik menggunakan kata, suara ataupun yang lainnya. Dengan kemampuan ini, diharapkan komputer dapat

mengambil keputusan dengan sendirinya untuk memecahkan berbagai kasus yang ditemuinya kemudian itulah yang disebut dengan kecerdasan buatan. Kecerdasan buatan adalah kemampuan komputer digital atau robot yang dikendalikan komputer untuk melakukan tugas yang umumnya dikaitkan dengan sesuatu yang cerdas. Istilah ini sering diterapkan pada proyek pengembangan sistem yang diberkahi dengan karakteristik proses intelektual manusia, seperti kemampuan untuk berpikir, menemukan makna, menggeneralisasi, atau belajar dari pengalaman masa lalu.

Kecerdasan Buatan (AI) merupakan salah satu bidang yang sangat berhubungan dengan memanfaatkan mesin (komputer) untuk memecahkan suatu masalah atau persoalan yang rumit dengan cara yang lebih mudah dimengerti oleh manusia. Kecerdasan Buatan (AI) yang semakin canggih yang mampu menambahkan pengetahuannya dengan cara melakukan banyak testing dan perkembangan dari target yang di analisa. Contoh dari kecerdasan buatan yang paling terkenal saat ini adalah Google Assistant, Alexa dan Siri. Google Assistant, Alexa dan Siri sangat dikenal karena penggunaannya yang mudah oleh user untuk menemukan berbagai hal atau membuatnya untuk melakukan sesuatu terhadap smartphone atau smarthome anda dan contohnya masih banyak lagi.

1.8.1.2 Sejarah Kecerdasan Buatan Kecerdasan Buatan (Artificial intelligence) mulai dibentuk sejak adanya komputer modern yang diperkirakan terjadi pada tahun 1940 dan 1950. Ilmu pengetahuan kecerdasan buatan ini dikhawasukan dalam perancangan otomatisasi tingkah laku cerdas dalam sistem kecerdasan komputer.

Pada awal 50-an, studi tentang “mesin berpikir” memiliki berbagai nama seperti cybernetics, teori automata, dan pemrosesan informasi. Pada tahun 1956, para ilmuan jenius seperti Alan Turing, Norbert, Wiener, Claude Shannon dan Warren McCullough telah bekerja secara independen dibidang cybernetics, matematika, algoritma dan teori jaringan. Namun, seprang ilmuan komputer dan kognitif John McCarthy adalah orang yang dating dengan ide untuk bergabung dengan upaya penelitian terpisah ini kedalam satu bidang yang akan mempelajari topic baru untuk imajinasi manusia yaitu kecerdasan buatan. Dia adalah orang yang menciptakan istilah tersebut dan kemudian mendirikan laboratorium Kecerdasan Buatan di MIT dan Stanford. Pada tahun 1956, McCarthy yang sama mendirikan Konferensi Dartmouth di Hanover, New Hampshire. Peneliti terkemuka dalam teori kompleksitas, simulasi bahasa, hubungan antara keacakan dan pemikiran kreatif, jaringan saraf diundang. Tujuan dari bidang penelitian yang baru dibuat adalah untuk mengembangkan mesin yang dapat mensimulasikan setiap aspek kecerdasan. Itulah sebabnya Konferensi Dartmouth 1956 dianggap sebagai kelahiran Kecerdasan Buatan. Sejak saat itu, Kecerdasan Buatan telah hidup melalui dekade kemuliaan dan cemoohan, yang dikenal luas sebagai musim panas dan musim dingin AI. Musim panasnya ditandai dengan optimisme dan dana be-

sar, sedangkan musim dinginnya dihadapkan dengan pemotongan dana, ketidakpercayaan dan pesimisme.

1.8.1.3 Perkembangan Kecerdasan Buatan Teknologi Artificial Intelligence semakin ramai dibahas dalam berbagai diskusi teknologi di seluruh dunia. Menurut kebanyakan orang, pekerjaan seperti kasir, operator telepon, pengendara truk, dan lainnya sangat berpeluang besar untuk tergantikan oleh Artificial Intelligence. Mengapa terjadi hal demikian? dikarenakan memang bahwa AI lebih unggul dalam hal kinerja, fitur dan lain sebagainya. Namun, dalam beberapa aspek memang pekerja manusia masih unggul dibandingkan AI itu sendiri. Para generasi muda yang ada di dunia terutama di daerah Asia terlihat sudah memahami fungsi dan efek dari AI dalam kehidupan kita sehari-hari. Berdasarkan survei yang dilakukan oleh Microsoft, terdapat 39 persen responden yang mempertimbangkan untuk menggunakan mobil tanpa pengeudi dan 36 persen lainnya setuju bahwa robot masa depan dengan software untuk beroperasi mampu meningkatkan produktivitas. Dari survey tersebut kita sebagai pengguna AI harus lebih bijaksana dalam pengembangan dan penggunaan dari AI sehingga tanpa memberikan efek samping terhadap etos kerja dan keseharian kita sebagai pengguna dalam kehidupan sehari-hari.

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

1.8.1.4 Supervised Learning Supervised Learning adalah tugas pengumpulan data untuk menyimpulkan fungsi dari data pelatihan berlabel. Data pelatihan terdiri dari serangkaian contoh pelatihan. Dalam supervised learning, setiap contoh adalah pasangan yang terdiri dari objek input (biasanya

vektor) dan nilai output yang diinginkan(juga disebut sinyal pengawasan super). Algoritma pembelajaran yang diawasi menganalisis data pelatihan dan menghasilkan fungsi yang disimpulkan, yang dapat digunakan untuk memetakan contoh-contoh baru. Skenario optimal akan memungkinkan algoritma menentukan label kelas dengan benar untuk instance yang tidak terlihat. Ini membutuhkan algoritma pembelajaran untuk menggeneralisasi dari data pelatihan untuk situasi yang tidak terlihat dengan cara yang "masuk akal". Supervised Learning adalah pendekatan dimana sudah terdapat data yang dilatih selain itu juga terdapat variable yang ditargetkan sehingga tujuan dari pendekatan ini yaitu mengelompokkan suatu data ke dta yang sudah ada. Supervised Learning menyediakan algoritma pembelajaran dengan jumlah yang diketahui untuk mendukung penilaian dimasa depan. Chatbots, mobil self-driving, program pengenalan wajah, sistem pakar dan robot adalah beberapa sistem yang dapat menggunakan pembelajaran yang diawasi atau tidak diawasi. Supervised Learning sebagian besar terkait dengan AI berbasis pengambilan tetapi mereka juga mungkin mampu menggunakan model pembelajaran generatif. Data pelatihan untuk pembelajaran yang diawasi mencakup serangkaian contoh dengan subjek input berpasangan dan output yang diinginkan (yang juga disebut sebagai sinyal pengawasan).

Dalam pembelajaran yang diawasi untuk pemrosesan gambar, misalnya sistem AI mungkin dilengkapi dengan gambar berlabel kendaraan dalam kategori seperti mobil dan truk. Setelah jumlah pengamatan yang cukup, sistem harus dapat membedakan antara dan mengkategorikan gambar yang tidak berlabel, dimana waktu pelatihan dapat dikatakan lengkap. Model Supervised Learning memiliki beberapa keunggulan dibandingkan pendekatan tanpa pengawasan, tetapi mereka juga memiliki keterbatasan. Sistem lebih cenderung membuat penilaian bahwa manusia dapat berhubungan, misalnya karena manusia telah memberikan dasar untuk keputusan. Namun, dalam kasus metode berbasis pengambilan, Supervised Learning mengalami kesulitan dalam menangani informasi baru. Jika suatu sistem dengan kategori untuk mobil dan truk disajikan dengan sepeda, misalnya ia harus salah dikelompokkan dalam satu kategori ata yang lain. Namun, jika sistem AI bersifat generatif, ia mungkin tidak tahu apa sepeda itu tetapi akan dapat mengenalinya sebagai milik kategori yang terpisah.

1.8.1.5 Klasifikasi Klasifikasi adalah pembagian sesuatu menurut kelas-kelas (class). Menurut Ilmu Pengetahuan, Klasifikasi merupakan proses pengelompokan benda berdasarkan ciri-ciri persamaan dan juga perbedaan. Dalam masalah klasifikasi, kami mencoba memprediksi sejumlah nilai terpisah. Label (y) umumnya datang dalam bentuk kategorikal dan mewakili sejumlah kelas. Dalam pembelajaran mesin dan statistik, klasifikasi adalah pendekatan pembelajaran yang diawasi di mana program komputer belajar dari input data yang diberikan kepadanya dan kemudian menggunakan pembelajaran ini untuk mengklasifikasikan pengamatan baru. Kumpulan data ini mungkin hanya bersifat dua kelas (seperti mengidentifikasi apakah orang tersebut berjenis ke-

lamin laki-laki atau perempuan atau bahwa surat itu spam atau bukan-spam) atau mungkin juga multi-kelas. Beberapa contoh masalah klasifikasi adalah: pengenalan ucapan, pengenalan tulisan tangan, identifikasi metrik, klasifikasi dokumen dll.

1.8.1.6 Regresi Regresi adalah metode analisis statistik yang digunakan untuk melihat pengaruh antara dua ataupun lebih variabel. Regresi adalah membahas masalah ketika variabel output adalah nilai riil atau berkelanjutan, seperti "gaji" atau "berat". Banyak model yang berbeda dapat digunakan makan, yang paling sederhana adalah regresi linier. Ia mencoba untuk menyesuaikan data dengan hyper-plane terbaik yang melewati poin.

1.8.1.7 Unsupervised Learning Unsupervised Learning berbeda dengan Supervised Lerning. Perbedaannya ialah unsupervised learning tidak memiliki data latih, sehingga dari data yang ada kita mengelompokan data tersebut menjadi 2 ataupun 3 bagian dan seterusnya. Unsupervised Learning adalah pelatihan algoritma kecerdasan buatan (AI) menggunakan informasi yang tidak diklasifikasikan atau diberi label dan memungkinkan algoritma untuk bertindak atas informasi tersebut tanpa bimbingan. Dalam Unsupervised Learning, sistem AI dapat mengelompokkan informasi yang tidak disortir berdasarkan persamaan dan perbedaan meskipun tidak ada kategori yang disediakan.

Dalam Unsupervised Learning, sistem AI disajikan dengan data yang tidak berlabel, tidak terkategorisasi dan algoritma sistem bekerja pada data tanpa pelatihan sebelumnya. Outputnya tergantung pada algoritma kode. Menundukkan suatu sistem pada Unsupervised Learning adalah salah satu cara untuk menguji AI. Algoritma Unsupervised Learning dapat melakukan tugas pemrosesan yang lebih kompleks daripada sistem pembelajaran yang diawasi. Namun, pembelajaran tanpa pengawasan bisa lebih tidak terduga dari pada model alternatif. Sementara Unsupervised Learning mungkin, misalnya, mencari tahu sendiri cara memilah kucing dari anjing, mungkin juga menambahkan kategori yang tidak terduga dan tidak diinginkan untuk menangani breed yang tidak biasa, membuat kekacauan bukannya keteraturan

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

tensor, atau array multi-dimensi. Jadi jalur pipa data yang dibangun untuk pembelajaran mendalam umumnya akan mengkonversi semua data - baik itu gambar, video, suara, suara, teks atau deret waktu menjadi vektor dan tensor yang dapat diterapkan operasi aljabar linier. Data itu seringkali perlu dinormalisasi, distandarisasi dan dibersihkan untuk meningkatkan kegunaannya, dan itu semua adalah langkah dalam ETL pembelajaran mesin. DeepLearning4j menawarkan alat ETV DataVec untuk melakukan tugas-tugas pemrosesan data tersebut.

Pembelajaran yang dalam, dan pembelajaran mesin yang lebih umum, membutuhkan pelatihan yang baik agar bekerja dengan baik. Mengumpulkan dan membangun set pelatihan badan yang cukup besar dari data yang diketahui membutuhkan waktu dan pengetahuan khusus domain tentang di mana dan bagaimana mengumpulkan informasi yang relevan. Perangkat pelatihan bertindak sebagai tolok ukur terhadap mana jaring pembelajaran dalam dilatih. Itulah yang mereka pelajari untuk direkonstruksi sebelum mereka melepaskan data yang belum pernah mereka lihat sebelumnya. Pada tahap ini, manusia yang berpengetahuan luas perlu menemukan data mentah yang tepat dan mengubahnya menjadi representasi numerik yang dapat dipahami oleh algoritma pembelajaran mendalam, tensor. Membangun set pelatihan, dalam arti tertentu, pra-pra pelatihan. Set pelatihan yang membutuhkan banyak waktu atau keahlian dapat berfungsi sebagai keunggulan dalam dunia ilmu data dan pemecahan masalah. Sifat keahlian sebagian besar dalam memberi tahu algoritma Anda apa yang penting bagi Anda dengan memilih apa yang masuk ke dalam set pelatihan. Ini melibatkan menceritakan sebuah kisah melalui data awal yang Anda pilih yang akan memandu jaring pembelajaran mendalam Anda saat mereka mengekstraksi fitur-fitur penting, baik di set pelatihan maupun dalam data mentah yang telah mereka ciptakan untuk dipelajari. Untuk membuat set pelatihan yang bermanfaat, Anda harus memahami masalah yang Anda selesaikan; yaitu apa yang Anda inginkan agar jaring pembelajaran mendalam Anda memperhatikan, di mana hasil yang ingin Anda prediksi.

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

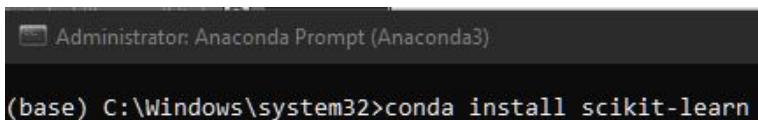
1.8.1.10 Testing Set Testing Set adalah set yang digunakan untuk mengukur sejauh mana classifier berhasil melakukan klasifikasi dengan benar. Ini berfungsi sebagai meterai persetujuan, dan Anda tidak menggunakaninya

sampai akhir. Setelah Anda melatih dan mengoptimalkan data Anda, Anda menguji jaringan saraf Anda terhadap pengambilan sampel acak akhir ini. Hasil yang dihasilkannya harus memvalidasi bahwa jaring Anda secara akurat mengenali gambar, atau mengenalinya setidaknya [x] dari jumlah tersebut. Jika Anda tidak mendapatkan prediksi yang akurat, kembalilah ke set pelatihan, lihat hyperparameter yang Anda gunakan untuk menyetel jaringan, serta kualitas data Anda dan lihat teknik pra-pemrosesan Anda.

1.8.2 Praktek

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

1. Pastikan anda telah menginstall anaconda lalu buka aplikasi Anaconda Prompt
2. Lalu pastikan anda telah menginstall python
3. Pada Anaconda Prompt install scikit dengan cara conda install scikit-learn

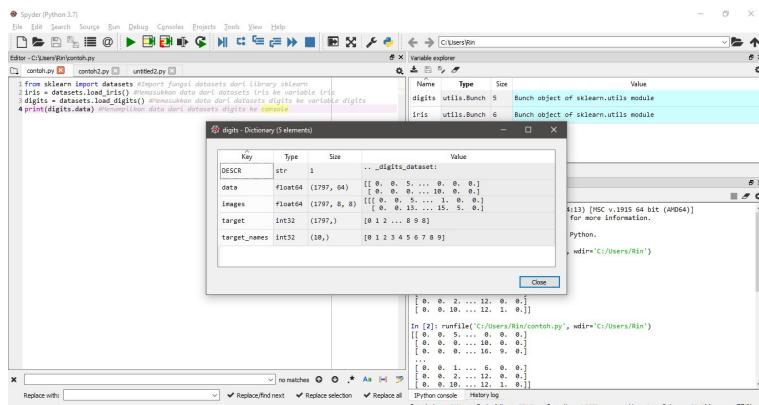


```
(base) C:\Windows\system32>conda install scikit-learn
```

Gambar 1.29 Instalasi Scikit Dari Anaconda Prompt

4. Lalu tulis kode yang ada dibawah ini dan run menggunakan spyder

```
1 from sklearn import datasets #Import fungsi datasets dari
   library sklearn
2 iris = datasets.load_iris() #Memasukkan data dari datasets
   iris ke variable iris
3 digits = datasets.load_digits() #Memasukkan data dari
   datasets digits ke variable digits
4 print(digits.data) #Menampilkan data dari datasets digits ke
   console
```



Gambar 1.30 Running Kode dari Spyder dan Hasil Variable Explorer

```
In [2]: runfile('C:/Users/Rin/contoh.py', wdir='C:/Users/Rin')
[[ 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 1.31 Running Loading an example dataset dari Spyder

1.8.2.2 Loading an example dataset

- Import fungsi datasets dari library sklearn

```
1 from sklearn import datasets #Import fungsi datasets dari  
library sklearn
```

- Memasukkan data dari datasets iris ke variable iris

```
1 iris = datasets.load_iris() #Memasukkan data dari datasets  
    iris ke variable iris
```

- Memasukkan data dari datasets digits ke variable digits

```
1 digits = datasets.load_digits() #Memasukkan data dari  
    datasets digits ke variable digits
```

- Menampilkan data dari datasets digits ke console

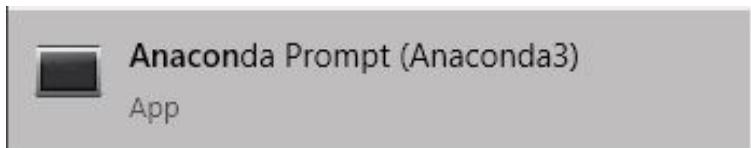
```
1 print(digits.data) #Menampilkan data dari datasets digits ke console
```

```
>>> from sklearn import datasets
```

Gambar 1.32 Hasil Running kode loading an example dataset

1.8.2.3 Learning and predicting

- Buka Anaconda Prompt



Gambar 1.33 Anaconda Prompt

- Lalu kita import datasets dari sklearn seperti dibawah ini

```
>>> from sklearn import datasets
```

Gambar 1.34 Menggunakan datasets

- lalu kita mendefinisikan iris dan digits menjadi variable

```
>>> iris = datasets.load_iris()
```

Gambar 1.35 mendefinisikan iris

```
>>> digits = datasets.load_digits()
```

Gambar 1.36 mendefinisikan digits

- Lalu kita import svm dari sklearn yang nantinya digunakan untuk menjadi estimasi angka kita

```
>>> from sklearn import svm
```

Gambar 1.37 Menggunakan `svm`

- Lalu, kita definisikan `clf` sebagai classifier, disini gamma didefinisikan secara manual

```
>>> clf = svm.SVC(gamma=0.001, C=100.)
```

Gambar 1.38 Mendefinisikan Classifier

- Estimator `clf` (for classifier) pertama kali dipasang pada model. Ini dilakukan dengan melewati training set ke metode `fit`. Untuk training set, akan menggunakan semua gambar dari set data yang ada, kecuali untuk gambar terakhir, yang dicadangkan untuk prediksi. Pada skrip dibawah memilih training set dengan sintaks Python `[:-1]`, yang menghasilkan array baru yang berisi semua kecuali item terakhir dari digits.

```
>>> clf.fit(digits.data[:-1], digits.target[:-1])
SVC(C=100.0, 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 1.39 Memanggil Classifier

- Menunjukkan prediksi angka baru

```
>>> clf.predict(digits.data[-1:])
array([8])
```

Gambar 1.40 Prediksi nilai baru

```
1 lastlinelastline
2 from sklearn import datasets #Import fungsi datasets dari library
   sklearn
3 iris = datasets.load_iris() #Memasukkan data dari datasets iris
   ke variable iris
4 digits = datasets.load_digits() #Memasukkan data dari datasets
   digits ke variable digits
```

```

5 from sklearn import svm #Mengimport sebuah Support Vector Machine
6 # (SVM) yang merupakan algoritma classification yang akan
7 # diambil dari Scikit-Learn.
8 clf = svm.SVC(gamma=0.001, C=100.) #Mendeklarasikan suatu value
9 # yang bernama clf yang berisi gamma.
10 clf.fit(digits.data[:-1], digits.target[:-1]) #Estimator clf (for
11 classifier)
12 hasil = clf.predict(digits.data[-1:]) #Menunjukkan prediksi
13 angka baru
14 print(hasil)

```

1.8.2.4 Model Persistance Model Persistance

```

1 lastlinelastline
2 %%Cara Dump Pertama
3 from sklearn import svm #Mengimport sebuah Support Vector
4 #Machine(SVM) yang merupakan algoritma classification yang
5 #akan diambil dari Scikit-Learn.
6 from sklearn import datasets #Import fungsi datasets dari library
7 #sklearn
8 clf = svm.SVC() #Mendefinisikan clf dengan fungsi svc dari
9 #library svm
10 X, y = datasets.load_iris(return_X_y=True) #Mengisi variable x
11 #dan y dengan data dari datasets
12 clf.fit(X, y) #Estimator clf (for classifier)
13
14
15 %%Cara Dump Kedua
16 from sklearn import svm # Digunakan untuk memanggil class svm
17 # dari library sklearn
18 from sklearn import datasets # Diguangkan untuk class datasets
19 # dari library sklearn
20 clf = svm.SVC() # membuat variabel clf , dan
21 # memanggil class svm dan fungsi SVC
22 X, y = datasets.load_iris(return_X_y=True) #Mengambil dataset
23 # iris dan mengembalikan nilainya.
24 clf.fit(X, y) #Perhitungan nilai label
25
26 from joblib import dump, load #memanggil class dump dan load pada
27 #library joblib
28 dump(clf, '1174066.joblib') #Menyimpan model kedalam 1174066.
29 #joblib
30 hasil = load('1174066.joblib') #Memanggil model 1174066
31 hasil.predict(X[0:1])
32 print(y[0]) # Menampilkan Model yang dipanggil sebelumnya

```

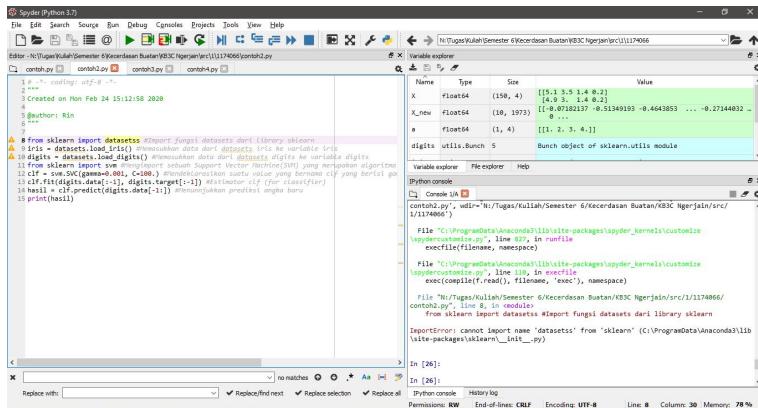
1.8.2.5 Conversion Conversion

```
1 lastlinelastline
2 import numpy as np # memanggil library numpy dan dibuat alias np
3 from sklearn import random_projection #Memanggil class
4         random_projection pada library sklearn
5
6 rng = np.random.RandomState(0) #Membuat variabel rng , dan
7         mendefinisikan np, fungsi random dan attr RandomState kedalam
8         variabel
9 X = rng.rand(10, 2000) # membuat variabel X, dan menentukan nilai
10        random dari 10 – 2000
11 X = np.array(X, dtype='float32') #menyimpan hasil nilai random
12        sebelumnya, kedalam array, dan menentukan typedatanya sebagai
13        float32
14 X.dtype # Mengubah data tipe menjadi float64
15
16 transformer = random_projection.GaussianRandomProjection() #
17         membuat variabel transformer , dan mendefinisikan
18         classrandom_projection dan memanggil fungsi
19         GaussianRandomProjection
20 X_new = transformer.fit_transform(X) # membuat variabel baru dan
21         melakukan perhitungan label pada variabel X
22 X_new.dtype # Mengubah data tipe menjadi float64
23 print(X_new) # Menampilkan isi variabel X_new
```

1.8.3 Penanganan Error

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

1. Screenshot Error



Gambar 1.41 ImportError: cannot import name 'datasetss' from 'sklearn'

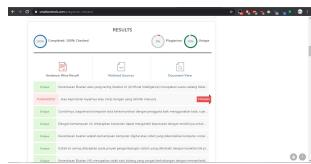
2. Tuliskan kode eror dan jenis errornya [hari ke 2](10)

- #### ■ ImportError

3. Solusi pemecahan masalah error tersebut[hari ke 2](10)

- ImportError
Cek kembali jika ada yang typo

1.8.4 Bukti Tidak Plagiat



Gambar 1.42 Bukti Tidak Melakukan Plagiat Chapter 1

1.8.5 Link Youtube

https://youtu.be/S8Sj_vZluUs

1.9 1174087 - Ilham Muhammad Ariq

1.9.1 Teori

1. Definisi Kecerdasan Buatan

Kecerdasan Buatan adalah suatu ilmu yang mempelajari bagaimana cara komputer melakukan sesuatu seperti yang dilakukan oleh manusia. Secara sederhana AI adalah teknik dan ilmu untuk membangun atau membuat suatu mesin menjadi cerdas, terutama pada program komputer. Kecerdasan yang dimaksud yaitu seperti yang dimiliki oleh manusia namun pada mesin akan dibuat cepat dan tepat atau akurat.

2. Sejarah Kecerdasan Buatan

Artificial intelligence merupakan inovasi baru di bidang ilmu pengetahuan. Mulai terbentuk sejak adanya komputer modern dan kira-kira terjadi sekitaran tahun 1940 dan 1950. Ilmu pengetahuan komputer ini khusus ditujukan dalam perancangan otomatisasi tingkah laku cerdas dalam sistem kecerdasan komputer. Pada awal 50-an, studi tentang "mesin berpikir" memiliki berbagai nama seperti cybernetics, teori automata, dan pemrosesan informasi. Pada tahun 1956, para ilmuan jenius seperti Alan Turing, Norbert Wiener, Claude Shannon dan Warren McCullough telah bekerja secara independen dibidang cybernetics, matematika, algoritma dan teori jaringan. Namun, seorang ilmuan komputer dan kognitif John McCarthy adalah orang yang dating dengan ide untuk bergabung dengan

upaya penelitian terpisah ini kedalam satu bidang yang akan mempelajari topic baru untuk imajinasi manusia yaitu kecerdasan buatan. Dia adalah orang yang menciptakan istilah tersebut dan kemudian mendirikan laboratorium Kecerdasan Buatan di MIT dan Stan ford.

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

3. Perkembangan Kecerdasan Buatan

Teknologi Artificial Intelligence semakin ramai dibahas dalam berbagai diskusi teknologi di seluruh dunia. Menurut kebanyakan orang, pekerjaan seperti kasir, operator telepon, pengendara truk, dan lainnya sangat berpeluang besar untuk tergantikan oleh Artificial Intelligence. Mengapa terjadi hal demikian? dikarenakan memang bahwa AI lebih unggul dalam hal kinerja, fitur dan lain sebagainya. Namun, dalam beberapa aspek memang pekerja manusia masih unggul dibandingkan AI itu sendiri. Para generasi muda yang ada di dunia terutama di daerah Asia terlihat sudah memahami fungsi dan efek dari AI dalam kehidupan kita sehari-hari. Berdasarkan survei yang dilakukan oleh Microsoft, terdapat 39 persen responden yang mempertimbangkan untuk menggunakan mobil tanpa pengemudi dan 36 persen lainnya setuju bahwa robot masa depan dengan software untuk beroperasi mampu meningkatkan produktivitas. Dari survei tersebut kita sebagai pengguna AI harus lebih bijaksana dalam pengembangan dan penggunaan dari AI sehingga tanpa memberikan efek samping terhadap etos kerja dan keseharian kita sebagai pengguna dalam kehidupan sehari-hari.

AI Summer 1 (1956-1973) KOnferensi Dartmounth diikuti oleh 17 tahun kemajuan luar biasa. Proyek penelitian yang dilakukan di MIT, universitas di Edinburgh, Stanford dan Carnegie Mellon menerima dana besar-besaran, yang akhirnya membawa hasil. Selama tahun-tahun itulah komputer pemrograman mulai melakukan masalah aljabar, membuktikan teorema geometris, memahami dan menggunakan sintaks dan tata bahasa Inggris. Terlepas dari ditinggalkannya koneksionisme dan terjemahan mesin yang gagal, yang menunda penelitian Natural Language Processing (NLP) selama bertahun-tahun, banyak prestasi dari masa lalu

yang membuat sejarah. Berikut ini beberapa diantaranya : Pelopor pembelajaran mesin, Ray Solomonoff meletakkan dasar-dasar teori metematika AI, memperkenalkan metode Bayesian universal untuk inferensi dan prediksi induktif Thomas Evans menciptakan program ANALOGI heuristik, yang memungkinkan komputer memecahkan masalah geometri-analogi Unimation, perusahaan robotika pertama didunia, menciptakan robot industri Unimate, yang bekerja pada jalur perakitan modil General Motors. Joseph Weizenbaum membangun ELIZA-program interaktif yang dapat membawa percakapan dalam bahasan Inggris tentang topik apapun. Ross Quillian menunjukkan jaring semantik, sedangkan Jaime Carbonell (Sr.) mengembangkan Cendikia-program interaktif untuk instruksi yang dibantu komputer berdasarkan jaring semantik. Edward Feigenbaum dan Julian Feldman menerbitkan Computeks and Thought, kumpulan artikel pertama tentang AI.

4. Definisi supervised learning, klasifikasi, regresi, unsupervised learning, dataset, training set dan testing set.

- **Supervised Learning**

Supervised Learning merupakan sebuah tipe learning yang mempunyai variable input dan variable output, tipe ini juga menggunakan satu algoritma atau lebih dari satu algoritma yang digunakan untuk mempelajari fungsi pemetaan dari input ke output.

- **Klasifikasi**

Klasifikasi adalah pengelompokan data di mana data yang digunakan memiliki label atau kelas target. Sehingga algoritma untuk menyelesaikan masalah klasifikasi dikategorikan ke dalam pembelajaran terbimbing.

- **Regresi**

Regresi metode analisis statistik yang digunakan untuk dapat melihat efek antara dua atau lebih variabel. Hubungan variabel dalam pertanyaan adalah fungsional yang diwujudkan dalam bentuk model matematika. Dalam analisis regresi, variabel dibagi menjadi dua jenis, yaitu variabel respons atau yang biasa disebut variabel dependen dan variabel independen atau dikenal sebagai variabel independen. Ada beberapa jenis analisis regresi, yaitu regresi sederhana yang mencakup linear sederhana dan regresi non-linear sederhana dan regresi berganda yang mencakup banyak linier atau non-linear berganda. Analisis regresi digunakan dalam pembelajaran mesin pembelajaran dengan metode pembelajaran terawasi.

- **Unsupervised learning**

Unsupervised learning jenis pembelajaran di mana kita hanya memiliki data input (input data) tetapi tidak ada variabel output yang terkait. Tujuan dari pembelajaran tanpa pengawasan adalah untuk

memodelkan struktur dasar atau distribusi data dengan tujuan mempelajari data lebih lanjut, dengan kata lain, itu adalah fungsi simpulan yang menggambarkan atau menjelaskan data.

- Data set

Data set objek yang merepresentasikan data dan relasinya di memory. Strukturnya mirip dengan data di database. Dataset berisi koleksi dari datatable dan datarelation.

- Training Set

Training set adalah bagian dari dataset yang di latih untuk membuat prediksi atau menjalankan fungsi dari algoritma ML lain sesuai dengan masing-masing. Memberikan instruksi melalui algoritma sehingga mesin yang di praktikkan dapat menemukan korelasinya sendiri.

- Testing Set

testing set adalah bagian dari dataset yang kami uji untuk melihat akurasinya, atau dengan kata lain untuk melihat kinerjanya.

1.9.2 Praktek

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

Gambar 1.43 Instalasi Package Scikit Learn

Name	Type	Size	Value
iris	utils.Bunch	6	Bunch object of sklearn.utils module
x	float64	(150, 4)	[150. 3.5 1.4 0.2] [149. 3. 1.4 0.2]
y	int32	(150,)	[0 0 0 ... 2 2 2]

Gambar 1.44 Isi Variabel Explorer

- ## 2. Mencoba loading an example dataset

```
1 #%% Mencoba loading an example dataset
2 from sklearn import datasets # Digunakan Untuk Memanggil
   class datasets dari library sklearn
3 iris = datasets.load_iris() # Menggunakan contoh datasets
   iris
```

```

4 x = iris.data           # Menyimpan nilai data sets iris
  pada variabel x
5 y = iris.target         # Menyimpan nilai data label iris
  pada variabel y

```

3. Mencoba Learning dan predicting

```

1 %%Mencoba Learning dan predicting
2 from sklearn.neighbors import KNeighborsClassifier #Digunakan
  Untuk Memanggil fungsi KNeighborsClassifier
3                                     # pada
4 class sklearn dan library sklearn
5 import numpy as np # memanggil library numpy dan dibuat alias
  np
6 knn=KNeighborsClassifier(n_neighbors=1) #membuat variabel kkn
  , dan memanggil fungsi KNeighborsClassifier
7                                     #dan mendefinisikan k
  -nya adalah 1
8 knn.fit(x,y)                      #Perhitungan
  matematika library kkn
9 a=np.array([1.0,2.0,3.0,4.0])      #Membuat Array
10 a = a.reshape(1,-1)                #Mengubah Bentuk
  Array jadi 1 dimensi
11 hasil = knn.predict(a)            #Memanggil fungsi
  prediksi
12 print(hasil)                    #menampilkan hasil
  prediksi

```

4. Mencoba Model Persistence

```

1 %% Model Persistense
2 from sklearn import svm # Digunakan untuk memanggil class svm
  dari library sklearn
3 from sklearn import datasets # Diguangkan untuk class datasets
  dari library sklearn
4 clf = svm.SVC()           # membuat variabel clf , dan
  memanggil class svm dan fungsi SVC
5 X, y = datasets.load_iris(return_X_y=True) #Mengambil dataset
  iris dan mengembalikan nilainya.
6 clf.fit(X, y)             #Perhitungan nilai label
7
8 from joblib import dump, load #memanggil class dump dan load
  pada library joblib
9 dump(clf, '1174087.joblib') #Menyimpan model kedalam 1174027.
  joblib
10 hasil = load('1174087.joblib') #Memanggil model 1174027
11 print(hasil) # Menampilkan Model yang dipanggil sebelumnya

```

5. Mencoba Conventions

```

1 %% Conventions

```

```

2 import numpy as np # memanggil library numpy dan dibuat alias
    np
3 from sklearn import random_projection #Memanggil class
    random_projection pada library sklean
4
5 rng = np.random.RandomState(0) #Membuat variabel rng , dan
    mendefinisikan np, fungsi random dan attr RandomState
    kedalam variabel
6 X = rng.rand(10, 2000) # membuat variabel X, dan menentukan
    nilai random dari 10 – 2000
7 X = np.array(X, dtype='float32') #menyimpan hasil nilai
    random sebelumnya, kedalam array , dan menentukan
    typedatanya sebagai float32
8 X.dtype # Mengubah data tipe menjadi float64
9
10 transformer = random_projection.GaussianRandomProjection() #
        membuat variabel transformer , dan mendefinisikan
        classrandom_projection dan memanggil fungsi
        GaussianRandomProjection
11 X_new = transformer.fit_transform(X) # membuat variabel baru
        dan melakukan perhitungan label pada variabel X
12 X_new.dtype # Mengubah data tipe menjadi float64
13 print(X_new) # Menampilkan isi variabel X_new

```

1.9.3 Penanganan Error

1. ScreenShoot Error

```

ImportError: cannot import name 'dataset' from 'sklearn' (C:\Users\Peneliti\anaconda\lib
\site-packages\sklearn\_init_.py)

```

Gambar 1.45 Import Error

2. Tuliskan Kode Error dan Jenis Error

- Import Error

3. Cara Penangan Error

- Import Error

Dengan Menginstall Library Yang Tidak Ditemukan atau Memperbaiki Penulisan Library

1.9.4 Bukti Tidak Plagiat



Gambar 1.46 **Bukti Tidak Melakukan Plagiat**

1.9.5 Link Youtube:

<https://www.youtube.com/watch?v=-qw8q7jhEmg>

1.10 1174084 - Muhammad Reza Syachrani

1.10.1 Teori

1. Definisi Sejarah dan Perkembangan Kecerdasan Buatan

Definisi Kecerdasan Buatan, Kecerdasan Buatan biasa disebut dengan istilah AI (Artificial Intelligence). AI merupakan tentang bagaimana cara untuk melengkapi sebuah komputer dengan kemampuan yang dimiliki oleh manusia. Dengan demikian, Diharapkan komputer mampu mengambil keputusan sendiri untuk berbagai kasus yang ditemuiinya kemudian itu lah yang disebut dengan kecerdasan buatan. Kecerdasan buatan adalah kemampuan komputer yang dikendalikan komputer untuk melakukan tugas yang umumnya dikaitkan dengan sesuatu yang cerdas.

Sejarah Kecerdasan Buatan, Kecerdasan buatan / Artificial intelligence mulai terbentuk pada tahun 1940 dan 1950. Pada awal 50-an, studi tentang “mesin berpikir” memiliki berbagai nama seperti cybernetics, teori automata, dan pemrosesan informasi. Pada tahun 1956, para ilmuan jenius seperti Alan Turing, Norbert, Wiener, Claude Shannon dan Warren McCullough telah bekerja secara independen dibidang cybernetics, matematika, algoritma dan teori jaringan. Namun, seprang ilmuan komputer dan kognitif John McCarthy adalah orang yang dating dengan ide untuk bergabung dengan upaya penelitian terpisah ini kedalam satu bidang yang akan mempelajari topic baru untuk imajinasi manusia yaitu kecerdasan buatan. Pada tahun 1956, McCarthy yang sama mendirikan Konferensi Dartmouth di Hanover, New Hampshire. Tujuan dari bidang penelitian yang baru dibuat adalah untuk mengembangkan mesin yang dapat mensimulasikan setiap aspek kecerdasan. Oleh karena itu Konferensi Dartmouth 1956 dianggap sebagai kelahiran Kecerdasan Buatan.

Perkembangan Kecerdasan Buatan, perkembangan kecerdasan buatan

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

2. Definisi Supervised learning, klasifikasi, regresi, unsupervised learning, dataset, training set dan testing set.

- **Supervised Learning**

Supervised Learning merupakan tugas pengumpulan data untuk menyimpulkan fungsi dari data pelatihan berlabel. Data pelatihan terdiri dari serangkaian contoh pelatihan. Dalam supervised learning, setiap contoh adalah pasangan yang terdiri dari objek input (biasanya vektor) dan nilai output yang diinginkan(juga disebut sinyal pengawasan super).

- **Klasifikasi**

Klasifikasi adalah pembagian sesuatu menurut kelas-kelas. Klasifikasi merupakan proses dari pengelompokan benda berdasarkan ciri-ciri persamaan dan juga perbedaan. Dalam masalah klasifikasi, kami mencoba memprediksi sejumlah nilai terpisah.

- **Regresi**

Regresi adalah metode analisis statistik yang digunakan untuk melihat pengaruh antara dua ataupun lebih variabel. Regresi adalah

membahas masalah ketika variabel output adalah nilai riil atau berkelanjutan, seperti "gaji" atau "berat".

- Unsupervised learning

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

- Data set

Data set objek yang merepresentasikan data dan relasinya di memory. Strukturnya mirip dengan data di database. Dataset berisi koleksi dari datatable dan datarelation.

- Training Set

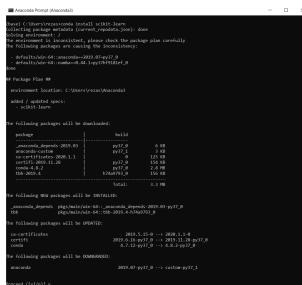
Training Set adalah set digunakan oleh algoritma klasifikasi. Dapat dicontohkan dengan decision tree, bayesian, neural network dll. Semuanya dapat digunakan untuk membentuk sebuah model classifier

- Testing Set

Testing Set merupakan set yang digunakan untuk mengukur sejauh mana classifier berhasil melakukan klasifikasi dengan benar. Dapat berfungsi sebagai meterai persetujuan, dan Anda tidak menggunakan-nya sampai akhir.

1.10.2 Praktek

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



Gambar 1.47 Instalasi Package Scikit Learn

Name	Type	Size	Value
iris	util.Bunch	4	Batch object of sklearn.utils module
x	float64	(150, 4)	[1, 2, 3, 4, 5, 6, 7, 8, 9]
y	int32	(150,)	[0, 0, 0, ..., 2, 2, 2]

Gambar 1.48 Isi Variabel Explorer

2. Mencoba loading an example dataset

```

1 #%% Mencoba loading an example dataset
2 from sklearn import datasets # Untuk memanggil class datasets
   dari library sklearn
3 iris = datasets.load_iris() # Menggunakan datasets iris
4 x = iris.data             # Menyimpan nilai data set iris
   pada variabel x
5 y = iris.target           # Menyimpan nilai data label iris
   pada variabel y

```

3. Mencoba Learning dan predicting

```

1 #%%Mencoba Learning dan predicting
2 from sklearn.neighbors import KNeighborsClassifier #Untuk
   Memanggil fungsi KNeighborsClassifier
3                                     # pada
   class sklearn dan library sklearn
4 import numpy as np # memanggil library numpy dan dibuat alias
   np
5 knn=KNeighborsClassifier(n_neighbors=1) #membuat variabel kkn
   , dan memanggil fungsi KNeighborsClassifier
6                                     #dan mendefinisikan k
   -nya adalah 1
7 knn.fit(x,y)                      #Perhitungan
   matematika library kkn
8 a=np.array([1.0,2.0,3.0,4.0])      #Membuat Array
9 a = a.reshape(1,-1)                 #Mengubah Bentuk
   Array jadi 1 dimensi
10 hasil = knn.predict(a)            #Memanggil fungsi
   prediksi
11 print(hasil)                    #menampilkan hasil
   prediksi

```

4. Mencoba Model Persistence

```

1 #%% Model Persistense
2 from sklearn import svm # Digunakan untuk memanggil class svm
   dari library sklearn
3 from sklearn import datasets # Diguankan untuk class datasets
   dari library sklearn
4 clf = svm.SVC()           # membuat variabel clf , dan
   memanggil class svm dan fungsi SVC
5 X, y = datasets.load_iris(return_X_y=True) #Mengambil dataset
   iris dan mengembalikan nilainya.
6 clf.fit(X, y)             #Perhitungan nilai label

```

```

7
8 from joblib import dump, load #memanggil class dump dan load
   pada library joblib
9 dump(clf, '1174084.joblib') #Menyimpan model kedalam 1174084.
   joblib
10 hasil = load('1174084.joblib') #Memanggil model 1174084
11 print(hasil) # Menampilkan Model yang dipanggil sebelumnya

```

5. Mencoba Conventions

```

1 %% Conventions
2 import numpy as np # memanggil library numpy dan dibuat alias
   np
3 from sklearn import random_projection #Memanggil
   random_projection pada library sklean
4
5 rng = np.random.RandomState(0) #Membuat variabel rng, dan
   mendefisikan np, fungsi random dan attr RandomState
   kedalam variabel
6 X = rng.rand(10, 2000) # membuat variabel X, dan menentukan
   nilai random dari 10 – 2000
7 X = np.array(X, dtype='float32') #menyimpan hasil nilai
   random sebelumnya, kedalam array, dan menentukan
   typedatanya sebagai float32
8 X.dtype # Mengubah data tipe menjadi float64
9
10 transformer = random_projection.GaussianRandomProjection() #
      membuat variabel transformer, dan mendefinisikan
      classrandom_projection dan memanggil fungsi
      GaussianRandomProjection
11 X_new = transformer.fit_transform(X) # membuat variabel baru
      dan melakukan perhitungan label pada variabel X
12 X_new.dtype # Mengubah data tipe menjadi float64
13 print(X_new) # Menampilkan isi variabel X_new

```

1.10.3 Penanganan Error

1. ScreenShoot Error

```
ModuleNotFoundError: No module named 'sklear'
```

Gambar 1.49 Module Error

2. Tuliskan Kode Error dan Jenis Error

- Module Error

3. Cara Penangan Error

- Module Error

Dengan memperbaiki penulisan atau kesalahan dalam kode atau melakukan install package atau modul yang belum terinstal

1.10.4 Bukti Tidak Plagiat



Gambar 1.50 Bukti Tidak Melakukan Plagiat

1.11 1174083 - Bakti Qilan Mufid

1.11.1 Teori

1.11.1.1 Kecerdasan Buatan

1. Definisi Kecerdasan Buatan

Kecerdasan Buatan biasa disebut dengan istilah AI (Artificial Intelligence). AI sendiri merupakan suatu cabang dalam bisnis sains komputer sains dimana mengkaji tentang bagaimana cara untuk men lengkapinya sebuah komputer dengan kemampuan atau kepintaran layaknya atau mirip dengan yang dimiliki manusia. Sebagai contoh, sebagaimana komputer dapat berkomunikasi dengan pengguna baik menggunakan kata, suara maupun lain sebagainya. Dengan kemampuan ini, diharapkan komputer mampu mengambil keputusan sendiri untuk berbagai kasus yang ditemuiinya kemudian itulah yang disebut dengan kecerdasan buatan. Kecerdasan buatan adalah kemampuan komputer digital atau robot yang dikendalikan komputer untuk melakukan tugas yang umumnya dikaitkan dengan sesuatu yang cerdas. Istilah ini sering diterapkan pada proyek pengembangan sistem yang diberkahi dengan karakteristik proses intelektual manusia, seperti kemampuan untuk berpikir, menemukan makna, menggeneralisasi, atau belajar dari pengalaman masa lalu.

Kecerdasan Buatan adalah salah satu bidang studi yang berhubungan dengan pemanfaatan mesin untuk memecahkan persoalan yang rumit dengan cara lebih manusiawi dan lebih bisa dipahami oleh manusia. Kecerdasan buatan makin canggih dengan kemampuan komputer dalam memperbarui pengetahuannya dengan banyaknya testing dan perkembangan target analisa. Untuk kecerdasan buatan ada banyak contoh dan jenisnya. Salah satu contoh yang paling terkenal dari Artificial Intelligence ialah Google Assistant. Google Assistant digunakan untuk kemudahan user dalam menemukan berbagai hal maupun penyetelan langsung terhadap smartphone yang digunakan dan masih banyak lagi.

2. Sejarah Kecerdasan Buatan

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

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

3. Perkembangan Kecerdasan Buatan

Teknologi Artificial Intelligence semakin ramai dibahas dalam berbagai diskusi teknologi di seluruh dunia. Menurut kebanyakan orang, pekerjaan seperti kasir, operator telepon, pengendara truk, dan lainnya sangat berpeluang besar untuk tergantikan oleh Artificial Intelligence. Mengapa terjadi hal demikian? dikarenakan memang bahwa AI lebih unggul dalam hal kinerja, fitur dan lain sebagainya. Namun, dalam beberapa aspek memang pekerja manusia masih unggul dibandingkan AI itu sendiri. Para generasi muda yang ada di dunia terutama di daerah Asia terlihat sudah memahami fungsi dan efek dari AI dalam kehidupan kita sehari-hari. Berdasarkan survei yang dilakukan oleh Microsoft, terdapat 39 persen responden yang mempertimbangkan untuk menggunakan mobil tanpa pengemudi dan 36 persen lainnya setuju bahwa robot masa depan dengan software untuk beroperasi mampu meningkatkan produktivitas. Dari survei tersebut kita sebagai pengguna AI harus lebih bijaksana dalam pengembangan dan penggunaan dari AI sehingga tanpa

memberikan efek samping terhadap etos kerja dan keseharian kita sebagai pengguna dalam kehidupan sehari-hari.

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

1.11.1.2 *Scikit-Learn*

1. Supervised Learning

Supervised Learning adalah tugas pengumpulan data untuk menyimpulkan fungsi dari data pelatihan berlabel. Data pelatihan terdiri dari serangkaian contoh pelatihan. Dalam supervised learning, setiap contoh adalah pasangan yang terdiri dari objek input (biasanya vektor) dan nilai output yang diinginkan(juga disebut sinyal pengawasan super). Algoritma pembelajaran yang diawasi menganalisis data pelatihan dan menghasilkan fungsi yang disimpulkan, yang dapat digunakan untuk memetakan contoh-contoh baru. Skenario optimal akan memungkinkan algoritma menentukan label kelas dengan benar untuk instance yang tidak terlihat. Ini membutuhkan algoritma pembelajaran untuk menggeneralisasi dari data pelatihan untuk situasi yang tidak terlihat dengan cara yang "masuk akal". Supervised Learning adalah pendekatan dimana sudah terdapat data yang dilatih selain itu juga terdapat variable yang ditargetkan sehingga tujuan dari pendekatan ini yaitu mengelompokkan suatu data ke dta yang sudah ada. Supervised Learning menyediakan algoritma pembelajaran dengan jumlah yang diketahui untuk men-

dukung penilaian dimasa depan. Chatbots, mobil self-driving, program pengenalan wajah, sistem pakar dan robot adalah beberapa sistem yang dapat menggunakan pembelajaran yang diawasi atau tidak diawasi. Supervised Learning sebagian besar terkait dengan AI berbasis pengambilan tetapi mereka juga mungkin mampu menggunakan model pembelajaran generatif. Data pelatihan untuk pembelajaran yang diawasi mencakup serangkaian contoh dengan subjek input berpasangan dan output yang diinginkan (yang juga disebut sebagai sinyal pengawasan).

Dalam pembelajaran yang diawasi untuk pemrosesan gambar, misalnya sistem AI mungkin dilengkapi dengan gambar berlabel kendaraan dalam kategori seperti mobil dan truk. Setelah jumlah pengamatan yang cukup, sistem harus dapat membedakan antara dan mengkategorikan gambar yang tidak berlabel, dimana waktu pelatihan dapat dikatakan lengkap. Model Supervised Learning memiliki beberapa keunggulan dibandingkan pendekatan tanpa pengawasan, tetapi mereka juga memiliki keterbatasan. Sistem lebih cenderung membuat penilaian bahwa manusia dapat berhubungan, misalnya karena manusia telah memberikan dasar untuk keputusan. Namun, dalam kasus metode berbasis pengambilan, Supervised Learning mengalami kesulitan dalam menangani informasi baru. Jika suatu sistem dengan kategori untuk mobil dan truk disajikan dengan sepeda, misalnya ia harus salah dikelompokkan dalam satu kategori ata yang lain. Namun, jika sistem AI bersifat generatif, ia mungkin tidak tahu apa sepeda itu tetapi akan dapat mengenalinya sebagai milik kategori yang terpisah.

2. Regresi

Regresi adalah metode analisis statistik yang digunakan untuk melihat pengaruh antara dua ataupun lebih variabel. Regresi adalah membahas masalah ketika variabel output adalah nilai riil atau berkelanjutan, seperti "gaji" atau "berat". Banyak model yang berbeda dapat digunakan makan, yang paling sederhana adalah regresi linier. Ia mencoba untuk menyesuaikan data dengan hyper-plane terbaik yang melewati poin.

3. Klasifikasi

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

surat itu spam atau bukan-spam) atau mungkin juga multi-kelas. Beberapa contoh masalah klasifikasi adalah: pengenalan ucapan, pengenalan tulisan tangan, identifikasi metrik, klasifikasi dokumen dll.

4. Unsupervised Learning

Unsupervised Learning berbeda dengan Supervised Learning. Perbedaannya ialah unsupervised learning tidak memiliki data latih, sehingga dari data yang ada kita mengelompokan data tersebut menjadi 2 ataupun 3 bagian dan seterusnya. Unsupervised Learning adalah pelatihan algoritma kecerdasan buatan (AI) menggunakan informasi yang tidak diklasifikasi atau diberi label dan memungkinkan algoritma untuk bertindak atas informasi tersebut tanpa bimbingan. Dalam Unsupervised Learning, sistem AI dapat mengelompokkan informasi yang tidak disortir berdasarkan persamaan dan perbedaan meskipun tidak ada kategori yang disediakan.

Dalam Unsupervised Learning, sistem AI disajikan dengan data yang tidak berlabel, tidak terkategorisasi dan algoritma sistem bekerja pada data tanpa pelatihan sebelumnya. Outputnya tergantung pada algoritma kode. Menundukkan suatu sistem pada Unsupervised Learning adalah salah satu cara untuk menguji AI. Algoritma Unsupervised Learning dapat melakukan tugas pemrosesan yang lebih kompleks daripada sistem pembelajaran yang diawasi. Namun, pembelajaran tanpa pengawasan bisa lebih tidak terduga daripada model alternatif. Sementara Unsupervised Learning mungkin, misalnya, mencari tahu sendiri cara memilah kucing dari anjing, mungkin juga menambahkan kategori yang tidak terduga dan tidak diinginkan untuk menangani breed yang tidak biasa, membuat kekacauan bukannya keteraturan

5. Data Set

Dataset adalah objek yang merepresentasikan data dan juga relasi yang ada di memory. Strukturnya mirip dengan data di database, namun bedanya dataset berisi koleksi dari data table dan data relation. mendapatkan data yang tepat berarti mengumpulkan atau mengidentifikasi data yang berkorelasi dengan hasil yang ingin Anda prediksi; yaitu data yang berisi sinyal tentang peristiwa yang Anda pedulikan. Data harus diselaraskan dengan masalah yang Anda coba selesaikan. Gambar kucing tidak terlalu berguna ketika Anda sedang membangun sistem identifikasi wajah. Memverifikasi bahwa data selaras dengan masalah yang ingin Anda selesaikan harus dilakukan oleh ilmuwan data. Jika Anda tidak memiliki data yang tepat, maka upaya Anda untuk membangun solusi AI harus kembali ke tahap pengumpulan data. Format ujung kanan untuk pembelajaran dalam umumnya adalah tensor, atau array multi-dimensi. Jadi jalur pipa data yang dibangun untuk pembelajaran mendalam umumnya akan mengkonversi semua data - baik itu gambar, video, suara, suara, teks atau deret waktu menjadi vektor dan tensor yang dapat diterapkan operasi aljabar linier. Data itu seringkali perlu

dinormalisasi, distandarisasi dan dibersihkan untuk meningkatkan kegunaannya, dan itu semua adalah langkah dalam ETL pembelajaran mesin. Deeplearning4j menawarkan alat ETV DataVec untuk melakukan tugas-tugas pemrosesan data tersebut.

Pembelajaran yang dalam, dan pembelajaran mesin yang lebih umum, membutuhkan pelatihan yang baik agar bekerja dengan baik. Mengumpulkan dan membangun set pelatihan badan yang cukup besar dari data yang diketahui membutuhkan waktu dan pengetahuan khusus domain tentang di mana dan bagaimana mengumpulkan informasi yang relevan. Perangkat pelatihan bertindak sebagai tolok ukur terhadap mana jaring pembelajaran dalam dilatih. Itulah yang mereka pelajari untuk direkonstruksi sebelum mereka melepaskan data yang belum pernah mereka lihat sebelumnya. Pada tahap ini, manusia yang berpengetahuan luas perlu menemukan data mentah yang tepat dan mengubahnya menjadi representasi numerik yang dapat dipahami oleh algoritma pembelajaran mendalam, tensor. Membangun set pelatihan, dalam arti tertentu, prapra pelatihan. Set pelatihan yang membutuhkan banyak waktu atau keahlian dapat berfungsi sebagai keunggulan dalam dunia ilmu data dan pemecahan masalah. Sifat keahlian sebagian besar dalam memberi tahu algoritma Anda apa yang penting bagi Anda dengan memilih apa yang masuk ke dalam set pelatihan. Ini melibatkan menceritakan sebuah kisah melalui data awal yang Anda pilih yang akan memandu jaring pembelajaran mendalam Anda saat mereka mengekstraksi fitur-fitur penting, baik di set pelatihan maupun dalam data mentah yang telah mereka ciptakan untuk dipelajari. Untuk membuat set pelatihan yang bermanfaat, Anda harus memahami masalah yang Anda selesaikan; yaitu apa yang Anda inginkan agar jaring pembelajaran mendalam Anda memperhatikan, di mana hasil yang ingin Anda prediksi.

6. Training Set

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

7. Testing Set

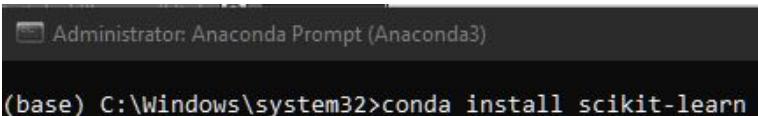
Testing Set adalah set yang digunakan untuk mengukur sejauh mana classifier berhasil melakukan klasifikasi dengan benar. Ini berfungsi sebagai meterai persetujuan, dan Anda tidak menggunakan sampai akhir.

Setelah Anda melatih dan mengoptimalkan data Anda, Anda menguji jaringan saraf Anda terhadap pengambilan sampel acak akhir ini. Hasil yang dihasilkannya harus memvalidasi bahwa jaring Anda secara akurat mengenali gambar, atau mengenalinya setidaknya [x] dari jumlah tersebut. Jika Anda tidak mendapatkan prediksi yang akurat, kembalilah ke set pelatihan, lihat hyperparameter yang Anda gunakan untuk menyetel jaringan, serta kualitas data Anda dan lihat teknik pra-pemrosesan Anda.

1.11.2 Praktek

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

1. Pastikan anda telah menginstall anaconda lalu buka aplikasi Anaconda Prompt
2. Lalu pastikan anda telah menginstall python
3. Pada Anaconda Prompt install scikit dengan cara conda install scikit-learn

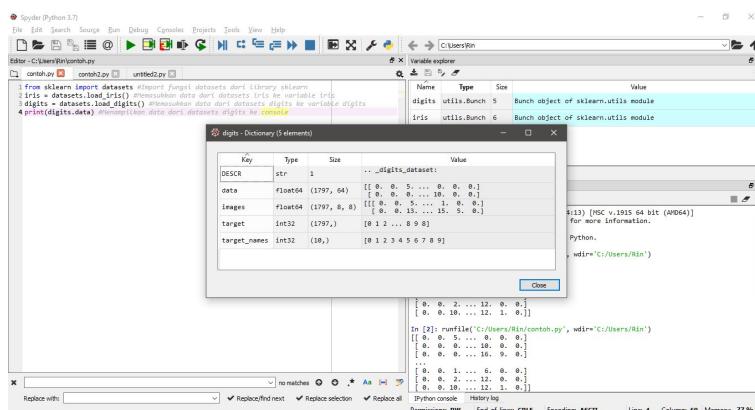


```
(base) C:\Windows\system32>conda install scikit-learn
```

Gambar 1.51 Instalasi Scikit Dari Anaconda Prompt

4. Lalu tulis kode yang ada dibawah ini dan run menggunakan spyder

```
1 from sklearn import datasets #Import fungsi datasets dari
   library sklearn
2 iris = datasets.load_iris() #Memasukkan data dari datasets
   iris ke variable iris
3 digits = datasets.load_digits() #Memasukkan data dari
   datasets digits ke variable digits
4 print(digits.data) #Menampilkan data dari datasets digits ke
   console
```



Gambar 1.52 Running Kode dari Spyder dan Hasil Variable Explorer

```

In [2]: runfile('C:/Users/Rin/contoh.py', wdir='C:/Users/Rin')
[[ 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 1.53 Running Loading an example dataset dari Spyder

1.11.2.2 Loading an example dataset

- Import fungsi datasets dari library sklearn

```

1 from sklearn import datasets #Import fungsi datasets dari
    library sklearn
```

- Memasukkan data dari datasets iris ke variable iris

```

1 iris = datasets.load_iris() #Memasukkan data dari datasets
    iris ke variable iris
```

- Memasukkan data dari datasets digits ke variable digits

```

1 digits = datasets.load_digits() #Memasukkan data dari
    datasets digits ke variable digits
```

- Menampilkan data dari datasets digits ke console

```

1 print(digits.data) #Menampilkan data dari datasets digits ke
    console
```

```
>>> from sklearn import datasets
```

Gambar 1.54 Hasil Running kode loading an example dataset

1.11.2.3 Learning and predicting

- Buka Anaconda Prompt



Gambar 1.55 Anaconda Prompt

- Lalu kita import datasets dari sklearn seperti dibawah ini

```
>>> from sklearn import datasets
```

Gambar 1.56 Menggunakan datasets

- lalu kita mendefinisikan iris dan digits menjadi variable

```
>>> iris = datasets.load_iris()
```

Gambar 1.57 mendefinisikan iris

```
>>> digits = datasets.load_digits()
```

Gambar 1.58 mendefinisikan digits

- Lalu kita import svm dari sklearn yang nantinya digunakan untuk menjadi estimasi angka kita

```
>>> from sklearn import svm
```

Gambar 1.59 Menggunakan `svm`

- Lalu, kita definisikan `clf` sebagai classifier, disini gamma didefinisikan secara manual

```
>>> clf = svm.SVC(gamma=0.001, C=100.)
```

Gambar 1.60 Mendefinisikan Classifier

- Estimator `clf` (for classifier) pertama kali dipasang pada model. Ini dilakukan dengan melewati training set ke metode `fit`. Untuk training set, akan menggunakan semua gambar dari set data yang ada, kecuali untuk gambar terakhir, yang dicadangkan untuk prediksi. Pada skrip dibawah memilih training set dengan sintaks Python `[:-1]`, yang menghasilkan array baru yang berisi semua kecuali item terakhir dari digits.

```
>>> clf.fit(digits.data[:-1], digits.target[:-1])
SVC(C=100.0, 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 1.61 Memanggil Classifier

- Menunjukkan prediksi angka baru

```
>>> clf.predict(digits.data[-1:])
array([8])
```

Gambar 1.62 Prediksi nilai baru

```
1 lastlinelastline
2 from sklearn import datasets #Import fungsi datasets dari library
   sklearn
3 iris = datasets.load_iris() #Memasukkan data dari datasets iris
   ke variable iris
4 digits = datasets.load_digits() #Memasukkan data dari datasets
   digits ke variable digits
```

```

5 from sklearn import svm #Mengimport sebuah Support Vector Machine
6   (SVM) yang merupakan algoritma classification yang akan
7   diambil dari Scikit-Learn.
8 clf = svm.SVC(gamma=0.001, C=100.) #Mendeklarasikan suatu value
9   yang bernama clf yang berisi gamma.
10 clf.fit(digits.data[:-1], digits.target[:-1]) #Estimator clf (for
11 classifier)
12 hasil = clf.predict(digits.data[-1:]) #Menunjukkan prediksi
13 angka baru
14 print(hasil)

```

1.11.2.4 Model Persistance Model Persistance

```

1 lastlinelastline
2 %%Cara Dump Pertama
3 from sklearn import svm #Mengimport sebuah Support Vector
4   Machine(SVM) yang merupakan algoritma classification yang
5   akan diambil dari Scikit-Learn.
6 from sklearn import datasets #Import fungsi datasets dari library
7   sklearn
8 clf = svm.SVC() #Mendefinisikan clf dengan fungsi svc dari
9   library svm
10 X, y = datasets.load_iris(return_X_y=True) #Mengisi variable x
11   dan y dengan data dari datasets
12 clf.fit(X, y) #Estimator clf (for classifier)
13
14
15 %%Cara Dump Kedua
16 from sklearn import svm # Digunakan untuk memanggil class svm
17   dari library sklearn
18 from sklearn import datasets # Diguakan untuk class datasets
19   dari library sklearn
20 clf = svm.SVC()           # membuat variabel clf , dan
21   memanggil class svm dan fungsi SVC
22 X, y = datasets.load_iris(return_X_y=True) #Mengambil dataset
23   iris dan mengembalikan nilainya.
24 clf.fit(X, y)             #Perhitungan nilai label
25
26 from joblib import dump, load #memanggil class dump dan load pada
27   library joblib
28 dump(clf, '1174083.joblib') #Menyimpan model kedalam 1174066.
29   joblib
30 hasil = load('1174083.joblib') #Memanggil model 1174066
31 hasil.predict(X[0:1])
32 print(y[0]) # Menampilkan Model yang dipanggil sebelumnya

```

1.11.2.5 Conventions Conversion

```

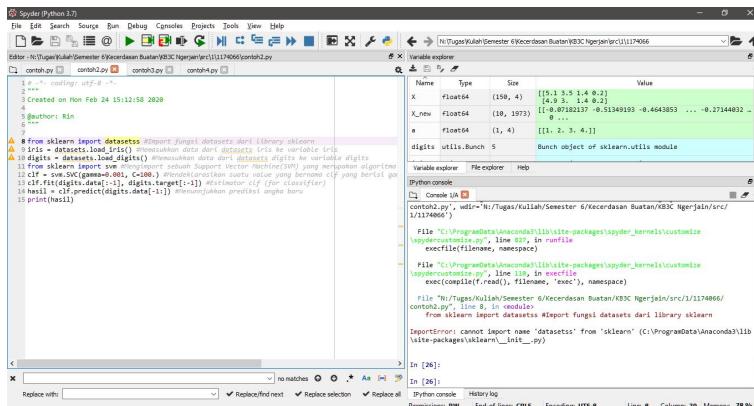
1 lastlinelastline
2 import numpy as np # memanggil library numpy dan dibuat alias np
3 from sklearn import random_projection #Memanggil class
        random_projection pada library sklearn
4
5 rng = np.random.RandomState(0) #Membuat variabel rng , dan
        mendefinisikan np, fungsi random dan attr RandomState kedalam
        variabel
6 X = rng.rand(10, 2000) # membuat variabel X, dan menentukan nilai
        random dari 10 – 2000
7 X = np.array(X, dtype='float32') #menyimpan hasil nilai random
        sebelumnya, kedalam array, dan menentukan typedatanya sebagai
        float32
8 X.dtype # Mengubah data tipe menjadi float64
9
10 transformer = random_projection.GaussianRandomProjection() #
        membuat variabel transformer , dan mendefinisikan
        classrandom_projection dan memanggil fungsi
        GaussianRandomProjection
11 X_new = transformer.fit_transform(X) # membuat variabel baru dan
        melakukan perhitungan label pada variabel X
12 X_new.dtype # Mengubah data tipe menjadi float64
13 print(X_new) # Menampilkan isi variabel X_new

```

1.11.3 Penanganan Error

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

1. Screenshoot Error



Gambar 1.63 ImportError: cannot import name 'datasetss' from 'sklearn'

2. Tuliskan kode eror dan jenis errornya [hari ke 2](10)

- ImportError

3. Solusi pemecahan masalah error tersebut [hari ke 2] (10)

- ImportError

Cek kembali jika ada yang typo

1.11.4 Bukti Tidak Plagiat



Gambar 1.64 Bukti Tidak Melakukan Plagiat Chapter 1

1.11.5 Link Youtube

<https://youtu.be/w4lTVoumb1g>

1.12 Nurul Izza Hamka - 1174062

1.12.1 Pemahaman Teori

1. Definisi Sejarah Kecerdasan Buatan

Kecerdasan buatan atau yang dikenal dengan Artificial Intelligence (AI) adalah suatu perkembangan teknologi yang muncul untuk membentuk suatu mesin teknologi yang lebih pintar yang mana agar lebih memudahkan setiap pekerjaan manusia. Selain itu AI ini juga untuk memahami kecerdasan dalam artian membuat sebuah mesin yang dapat membantu memahami kecerdasan contohnya dapat memecahkan sebuah masalah dengan lebih cepat.

2. Definisi Kecerdasan Buatan

Kecerdasan buatan atau yang disebut dengan Artificial Intelligence mulai muncul sekitar tahun 1940 dan 1950 sejak adanya komputer. Munculnya AI ini memberikan banyak keuntungan seperti AI ini berisfat permanen, artinya bisa digunakan secara berulang-ulang dimana saja dan kapan saja. Selain itu menawarkan kemudahan dalam artian data yang telah disimpan sebelumnya akan mudah untuk di akses kembali. Kerja AI ini juga lebih cepat jika dibandingkan dengan kerja manusia

3. Definisi Perkembangan Kecerdasan Buatan Tahun 1960 s/d 1970, mulailah berbagai diskusi tentang bagaimana komputer dapat menirukan dengan sedetail mungkin kemampuan otak manusia, saat itu dikategorikan dengan "classical AI". Kemudian pada tahun 1980, saat itu komputer sudah mudah didapatkan dengan harga yang terjangkau yang memudahkan berbagai riset dibidang kecerdasan buatan berkembangan dengan pesat di berbagai universitas dunia.

John McCarthy dari Massachusetts Institute of Technology atau yang dikenal sebagai Bapak AI, pada tahun 1956 McCarthy mengadakan konferensi Dartmouth Workshop yang melahirkan suatu bidang baru dengan nama "Artificial Intelligence". Pada konferensi Dartmouth itu mempertemukan semua para pendiri AI, dimana John McCarthy yang mengusulkan defisi dari AI itu. AI adalah cabang dari ilmu komputer yang berfokus pada pengembangan komputer yang dapat memiliki kemampuan layaknya manusia.

4. Definisi supervised learning Supervised learning mempunyai input dan output yang bisa dibuat menjadi model hubungan matematis, dan juga sebuah pendekatan dimana sudah terdapat data yang dilatih selain itu juga ada sebuah variable yang sudah ditargetkan sebagai tujuan dari pendekatan ini yaitu pengelompokan data ke data yang sudah ada sebelumnya.

Pengertian dalam konteks AI, supervised learning adalah sistem dimana sebuah input dan output data yang kita inginkan sudah tersedia. Input dan output data ini diberi label untuk klasifikasi dasar pembelajaran untuk pemrosesan data yang akan datang. Supervised learning ini menyediakan algoritma untuk pembelajaran dengan jumlah diketahui untuk mendukung sebuah penilaian yang akan datang seperti: Regresi Linear Berganda, Analisis Deret Waktu, Decision tree dan Random Forest, Artificial Neural Network, dan lain sebagainya.

5. Definisi Klasifikasi Klasifikasi merupakan sebuah proses pengelompokan benda berdasarkan ciri-ciri persamaan dan perbedaan. Artinya kita memberitahu mesin tersebut bagaimana cara pengerjaannya berdasarkan kelompok.
6. Definisi Regresi Regresi adalah bagian dari problem Supervised Learning, regresi ini menggunakan metode statistika.
7. Definisi Unsupervised Learning Unsupervised Learning berbeda dengan supervised learning. Unsupervised Learning tidak memiliki data latih, sehingga dari data yang telah ada kita kelompokkan menjadi dua atau tiga bagian begitupun seterusnya. Unsupervised Learning ini merupakan pelatihan algoritma kecerdasan buatan emnggunakan informasi

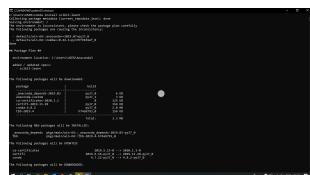
yang tidak diklasifikasikan atau diberi label dan memungkinkan algoritma untuk bertindak atas informasi tersebut tanpa panduan.

Tujuan dari algoritma tersebut adalah untuk mengelompokkan sebuah objek yang hampir mirip atau sama ke dalam area tertentu

8. Definisi Data Set Dataset merupakan objek yang merepresentasikan sebuah data dan relasi yang ada di memory. Struktur data set mirip dengan data yang ada didalam sebuah database. Namun, didalam dataset berisi sebuah koleksi dari data tabel dan data relation.
 9. Definisi Training Set Training set merupakan set yang digunakan oleh algoritma klasifikasi. Contohnya adalah decision tree, bayesian, neural network, dan lain sebagainya.
 10. Testing Set Testing set merupakan sebuah set yang digunakan untuk mengukur sejauh mana sebuah classifier berhasil melakukan klasifikasi dengan benar. Testing set berfungsi sebagai materai persetujuan, tapi tidak dapat kita gunakan sampai akhir. Setelah data di optimalkan, kita dapat melakukan pengujian jaringan saraf terhadap pengambilan sampel acak. Kemudian hasil yang diperolah harus valid bahwa jaringan kita akurat dalam menegnali gambar.

1.12.2 Intalasi

- ## 1. Instalasi Library Scikit Anaconda



Gambar 1.65 Instalasi Package Scikit Learn

Name	Type	Size		Value
X	Float32	(10, 200)		[0.5488115, 0.7158934, 0.6627633, ..., 0.4901803, 0.6456400, 0.5817731, ...]
X_mean	Float64	(10, 1373)		[0.5488115, 0.68074178, 1.14068481, ..., 0.4281331, 0.70515418]
y	Float64	(1, 4)		[1.2, 3.2, 4.1]
ysig	Int32	(1, 1)		[3]
iris				
iris.util.Bunch				
x	Float64	(150, 4)		[0.1, 2.5, 4.5, 6.5]
y	Int32	(150, 1)		[0, 0, ..., 2, 2, 2]

Gambar 1.66 Hasil Variabel Explorer

- ## 2. Mencoba Loading an Example Dataset

```

1 #%% Mencoba loading an Example Dataset
2 from sklearn import datasets #digunakan untuk memanggil class
   dataset dari library sklearn
3 iris = datasets.load_iris() # artinya kita menggunakan
   dataset iris
4 x = iris.data # artinya kita menyimpan data set iris di
   variable x
5 y = iris.target #artinya kita menyimpan data label iris pada
   variavle y

```

3. Mencoba Learning and Predicting

```

1 #%%Mmencoba Learning dan Predicting
2 from sklearn.neighbors import KNeighborsClassifier # kta
   menggunakan fungsi KNighborsClassifer pada kelas sklearn
   dan libarry sklearn
3 knn=KNeighborsClassifier(n_neighbors=1) # kita membuat
   variable knn, dan memanggil function KNighbors
4
5 knn.fit(x,y) #kita membuat perhitungan matematika library knn
6 a=np.array([1.0 ,2.0 ,3.0 ,4.0]) # artinya kita membuat array
7 a = a.reshape(1,-1) # mengubah bentuk array jadi 1
   dimensi
8 hasil = knn.predict(a) #kita memanggil fungsi prediksi
9 print(hasil) #menampilkan hasil prediksi

```

4. Mencoba Model Persintence

```

1 #%% Model Persistense
2 from sklearn import svm #untuk memanggil class svm dari
   library sklearn
3 from sklearn import datasets #untuk class dataset dati
   library sklearn
4 clf = svm.SVC() #kita membuat variable clf, dan memanggil
   class svm dan fungsi SVC
5 X, y = datasets.load_iris(return_X_y=True) #Memanggil dataset
   iris dan mengembalikan nilainya
6 clf.fit(X, y) # untuk menampilkan model yang dipanggil sebelumnya
7
8 from joblib import dump, load #memanggil class dump dan load
   pada library joblib
9 dump(clf, '1174062.joblib') #menyimpan model kedalam 1174062.
   joblib
10 hasil = load('1174062.joblib') #memanggil model 1174062
11 print(hasil) #untuk menampilkan model yang dipanggil
   sebelumnya

```

5. Mencoba Conventions

```

1 #%% mencoba Conventions

```

```

2 import numpy as np #digunakan untuk memanggil library numpy
    dan dibuat alias np
3 from sklearn import random_projection #memanggil class
    random_projection pada library sklearn
4
5 rng = np.random.RandomState(0) #untuk membuat variable rng,
    mendefinisikan np, function random dan attr randomstate
    kedalam variable
6 X = rng.rand(10, 2000) #membuat variable X, dan menentukan
    nilai random dari 10–2000
7 X = np.array(X, dtype='float32') #untuk menyimpan hasil nilai
    random senelumnya, kedalam array dan menentukan
    typedatanya sebagai float32
8 X.dtype #untuk mengubah data tipe menjadi float64
9
10 transformer = random_projection.GaussianRandomProjection() #
        membuat varibale transformer dan mendefinisikan
        classrandom_projection dan memanggil fungsi
        GaussianRandomProjection
11 X_new = transformer.fit_transform(X) #untuk membuat variable
        baru serta melakukan perhitungan label pada variabel X
12 X_new.dtype #mengubah data tipe menjadi float64
13 print(X_new) #menampilkan isi variable X_new

```

1.12.3 Penanganan Error

(a) Hasil ScreenShoot Error

```

file "D:\Kuliah\Semester 6\Kecerdasan Buatan\Bahan\1\sklearn.py", line 8, in <module>
    from sklearn import datasets
ImportError: cannot import name 'datasets' from 'sklearn' (D:\Kuliah\Semester
6\Kecerdasan Buatan\Bahan\1\sklearn.py)

```

Gambar 1.67 Import Error

(b) Tuliskan Kode Error dan Jenis Error

- Import Error

(c) Cara Penangan Error

- Import Error
Dengan Menginstall Library Yang Tidak Ditemukan

1.12.4 Bukti Tidak Melakukan Plagiat



Gambar 1.68 Bukti Tidak Plagiat

1.12.5 Link Youtube

- (d) Link Youtube Chapter 1
<https://www.youtube.com/watch?v=AdvP3Y18GE4>

1.13 Alvan Alvanzah (1174077)

1.13.1 Teori

1. Definisi, Sejarah dan Perkembangan Kecerdasan Buatan

Definisi kecerdasan buatan itu sendiri adalah suatu sistem teknologi yang didalamnya ditambahkan kecerdasan oleh manusia, kecerdasan buatan diatur dan dikembangkan dalam konteks ilmiah, dan bentukan dari kecerdasan entitas ilmiah yang ada. Kecerdasan Buatan atau dalam Bahasa Inggris sering disebut Artificial Intelligence yang sering disebut juga sebagai AI, pada 10 tahun lalu masyarakat belum terlalu mengetahui hal tersebut dan masih menjadi bahan candaan dikalangan masyarakat. Awal perkembangan AI dimulai pada tahun 1952-1969 yang dimulai dengan kesuksesan Newell dan temannya Simon menggunakan sebuah program yang disebut dengan General Problem Solver. Program ini dibangun untuk tujuan penyelesaian masalah secara manusiawi. Pada tahun 1966-1974 perkembangan kecerdasan buatan mulai melambat. Ada 3 faktor utama yang menyebabkan hal itu terjadi:

- Banyak subjek pada program AI yang bermunculan hanya mengandung sedikit atau bahkan sama sekali tidak mengandung sama sekali pengetahuan (knowledge).
- Kecerdasan buatan harus bisa menyelesaikan banyak masalah.
- Untuk menghasilkan perlakuan intelijensia ada beberapa batasan pada struktur yang bisa digunakan.

2. Definisi

Supervised learning, klasifikasi, regresi, unsupervised learning, dataset, trainingset dan testingset.

- Supervised Learning

Supervised Learning merupakan sebuah tipe learning yang mempun-

yai variable input dan variable output, tipe ini juga menggunakan satu algoritma atau lebih dari satu algoritma yang digunakan untuk mempelajari fungsi pemetaan dari input ke output.

- **Klasifikasi**

Klasifikasi adalah pengelompokan data di mana data yang digunakan memiliki label atau kelas target. Sehingga algoritma untuk menyelesaikan masalah klasifikasi dikategorikan ke dalam pembelajaran ter-bimbing.

- **Regresi**

Regresi metode analisis statistik yang digunakan untuk dapat melihat efek antara dua atau lebih variabel. Hubungan variabel dalam pertanyaan adalah fungsional yang diwujudkan dalam bentuk model matematika. Dalam analisis regresi, variabel dibagi menjadi dua jenis, yaitu variabel respons atau yang biasa disebut variabel dependen dan variabel independen atau dikenal sebagai variabel independen. Ada beberapa jenis analisis regresi, yaitu regresi sederhana yang mencakup linear sederhana dan regresi non-linear sederhana dan regresi berganda yang mencakup banyak linier atau non-linear berganda. Analisis regresi digunakan dalam pembelajaran mesin pembelajaran dengan metode pembelajaran terawasi.

- **Unsupervised Learning**

Unsupervised Learning jenis pembelajaran di mana kita hanya memiliki data input (input data) tetapi tidak ada variabel output yang terkait. Tujuan dari pembelajaran tanpa pengawasan adalah untuk memodelkan struktur dasar atau distribusi data dengan tujuan mempelajari data lebih lanjut, dengan kata lain, itu adalah fungsi simpulan yang menggambarkan atau menjelaskan data.

- **Data Set**

Data Set objek yang merepresentasikan data dan relasinya di memory. Strukturnya mirip dengan data di database. Dataset berisi koleksi dari datatable dan datarelation.

- **Training Set**

Training Set adalah bagian dari dataset yang di latih untuk membuat prediksi atau menjalankan fungsi dari algoritma ML lain sesuai dengan masing-masing. Memberikan instruksi melalui algoritma sehingga mesin yang di praktikkan dapat menemukan korelasinya sendiri.

- **Testing Set**

testing set adalah bagian dari dataset yang kami uji untuk melihat akurasinya, atau dengan kata lain untuk melihat kinerjanya.

1.13.2 Praktek

- Instalasi Library scikit dari ianaconda, mencoba kompilasi dan uji coba ambil contoh kode dan lihat variabel explorer



Gambar 1.69 Instalasi Package Scikit Learn

Name	Type	Size	Value
X	float64	(16, 2000)	[0.54683359 0.71531034 0.0829339 ... -0.4003976
X_new	float64	(16, 1973)	[0.64386044 0.3927731 ... 0.00000000 0.00000000 0.00000000 ... -0.79318666 0.31266443 ... -1.06165433 -0.00000000]
a	float64	(1, 4)	[1. 1. 1. 1.]
iris	utilist.Bunch	6	Բառապահ օբյեկտ sklearn.utils module
x	float64	(150, 4)	[5.0 3.4 1.5 0.2] [4.6 3.0 1.4 0.2] ...
y	int32	(150,)	[0 0 ... 2 2 2]

Gambar 1.70 Isi Variabel Explorer

- Mencoba loading an example dataset

```

1 %% Mencoba loading an example dataset
2 from sklearn import datasets # Digunakan Untuk Memanggil
   class datasets dari library sklearn
3 iris = datasets.load_iris() # Menggunakan contoh datasets
   iris
4 x = iris.data           # Menyimpan nilai data sets iris
   pada variabel x
5 y = iris.target         # Menyimpan nilai data label iris
   pada variabel y

```

- Mencoba Learning dan predicting

```

1 %%Mencoba Learning dan predicting
2 from sklearn.neighbors import KNeighborsClassifier #Digunakan
   Untuk Memanggil fungsi KNeighborsClassifier
3                                     # pada
   class sklearn dan library sklearn
4 import numpy as np # memanggil library numpy dan dibuat alias
   np
5 knn=KNeighborsClassifier(n_neighbors=1) #membuat variabel kkn
   , dan memanggil fungsi KNeighborsClassifier
6                                     #dan mendefinisikan k
   -nya adalah 1
7 knn.fit(x,y)                      #Perhitungan
   matematika library kkn

```

```

8 a=np.array([1.0 ,2.0 ,3.0 ,4.0])          #Membuat Array
9 a = a.reshape(1,-1)                         #Mengubah Bentuk
    Array jadi 1 dimensi
10 hasil = knn.predict(a)                     #Memanggil fungsi
     prediksi

```

4. Mencoba Model Persistence

```

1 %% Model Persistense
2 from sklearn import svm # Digunakan untuk memanggil class svm
    dari library sklearn
3 from sklearn import datasets # Diguankan untuk class datasets
    dari library sklearn
4 clf = svm.SVC()           # membuat variabel clf , dan
    memanggil class svm dan fungsi SVC
5 X, y = datasets.load_iris(return_X_y=True) #Mengambil dataset
    iris dan mengembalikan nilainya.
6 clf.fit(X, y)             #Perhitungan nilai label
7
8 from joblib import dump, load #memanggil class dump dan load
    pada library joblib
9 dump(clf, '1174077.joblib') #Menyimpan model kedalam 1174027.
    joblib
10 hasil = load('1174077.joblib') #Memanggil model 1174027

```

5. Mencoba Conventions

```

1 %% Conventions
2 import numpy as np # memanggil library numpy dan dibuat alias
    np
3 from sklearn import random_projection #Memanggil class
    random_projection pada library sklean
4
5 rng = np.random.RandomState(0) #Membuat variabel rng , dan
    mendefisikan np, fungsi random dan attr RandomState
    kedalam variabel
6 X = rng.rand(10, 2000) # membuat variabel X, dan menentukan
    nilai random dari 10 – 2000
7 X = np.array(X, dtype='float32') #menyimpan hasil nilai
    random sebelumnya, kedalam array , dan menentukan
    typedatanya sebagai float32
8 X.dtype # Mengubah data tipe menjadi float64
9
10 transformer = random_projection.GaussianRandomProjection() #
    membuat variabel transformer , dan mendefinisikan
    classrandom_projection dan memanggil fungsi
    GaussianRandomProjection
11 X_new = transformer.fit_transform(X) # membuat variabel baru
    dan melakukan perhitungan label pada variabel X
12 X_new.dtype # Mengubah data tipe menjadi float64

```

1.13.3 Penanganan Error

1. ScreenShoot Error



Gambar 1.71 Import Error



Gambar 1.72 Value Error

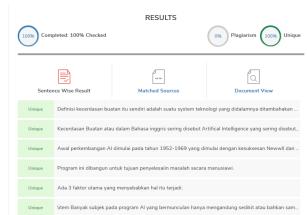
2. Tuliskan Kode Error dan Jenis Error

- Import Error
- Value Error

3. Cara Penanganan Error

- Import Error
Dengan Menginstall Library Yang Tidak Ditemukan
- Value Error
Mengubah Bentuk Arraynya, Menjadi 1 Dimensi

1.13.4 Bukti Tidak Plagiat



Gambar 1.73 Bukti Tidak Melakukan Plagiat Chapter 1

1.14 Difa Al Fansha

1.14.1 Teori

1. Definisi, Sejarah dan Perkembangan Kecerdasan Buatan

▪ Definisi Kecerdasan Buatan

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

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

▪ Sejarah Kecerdasan Buatan

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

Artificial intelligence merupakan inovasi baru di bidang ilmu pengetahuan. Mulai terbentuk sejak adanya komputer modern dan kira-kira terjadi sekitaran tahun 1940 dan 1950. Ilmu pengetahuan komputer ini khusus ditujukan dalam perancangan otomatisasi tingkah

laku cerdas dalam sistem kecerdasan komputer. Pada awal 50-an, studi tentang “mesin berpikir” memiliki berbagai nama seperti cybernetics, teori automata, dan pemrosesan innformasi. Pada tahun 1956, para ilmuan jenius seperti Alan Turing, Norbert, Wiener, Claude Shannon dan Warren McCullough telah bekerja secara independen dibidang cybernetics, matematika, algoritma dan teori jaringan. Namun, seprang ilmuan komputer dan kognitif John McCarthy adalah orang yang dating dengan ide untuk bergabung dengan upaya penelitian terpisah ini kedalam satu bidang yang akan mempelajari topic baru untuk imajinasi manusia yaitu kecerdasan buatan. Dia adalah orang yang menciptakan istilah tersebut dan kemudian mendirikan laboratorium Kecerdasan Buatan di MIT dan Stan ford.

- Perkembangan Kecerdasan Buatan

Teknologi Artificial Intelligence semakin ramai dibahas dalam berbagai diskusi teknologi di seluruh dunia. Menurut kebanyakan orang, pekerjaan seperti kasir, operator telepon, pengendara truk, dan lainnya sangat berpeluang besar untuk tergantikan oleh Artificial Intelligence. Mengapa terjadi hal demikian? dikarenakan memang bahwa AI lebih unggul dalam hal kinerja, fitur dan lain sebagainya. Namun, dalam beberapa aspek memang pekerja manusia masih unggul dibandingkan AI itu sendiri. Para generasi muda yang ada di dunia terutama di daerah Asia terlihat sudah memahami fungsi dan efek dari AI dalam kehidupan kita sehari-hari. Berdasarkan survei yang dilakukan oleh Microsoft, terdapat 39 persen responden yang mempertimbangkan untuk menggunakan mobil tanpa pengemudi dan 36 persen lainnya setuju bahwa robot masa depan dengan software untuk beroperasi mampu meningkatkan produktivitas. Dari survey tersebut kita sebagai pengguna AI harus lebih bijaksana dalam pengembangan dan penggunaan dari AI sehingga tanpa memberikan efek samping terhadap etos kerja dan keseharian kita sebagai pengguna dalam kehidupan sehari-hari.

AI Summer 1 (1956-1973) KOnferensi Dartmounth diikuti oleh 17 tahun kemajuan luar biasa. Proyek penelitian yang dilakukan di MIT, universitas di Edinburgh, Stanford dan Carnegie Mellon menerima dana besar-besaran, yang akhirnya membawa hasil. Selama tahun-tahun itulah komputer pemrograman mulai melakukan masalah aljabar, membuktikan teorema geometris, memahami dan menggunakan sintaks dan tata bahasa Inggris. Terlepas dari ditinggalkannya koneksiisme dan terjemahan mesin yang gagal, yang menunda penelitian Natural Language Processing (NLP) selama bertahun-tahun, banyak prestasi dari masa lalu yang membuat sejarah. Berikut ini beberapa diantaranya : Pelopor pembelajaran mesin, Ray Solomonoff mele-

takkan dasar-dasar teori metematika AI, memperkenalkan metode Bayesian universal untuk inferensi dan preddiksi induktif Thomas Evans menciptakan program ANALOGI heuristik, yang memungkinkan komputer memecahkan masalah geometrianalogi Unimation, perusahaan robotika pertama didunia, menciptakan robot industri Unimate, yang bekerja pada jalur perakitan modil Genenral Motors. Joseph Weizenbaum membangun ELIZA-program interaktif yang dapat membawa percakapan dalam bahasan Inggris tentang topik apapun. Ross Quillian menunjukkan jaring semantik, sedangkan Jaime Carbonell (Sr.) mengembangkan Cendikia-program interaktif untuk instruksi yang dibantu komputer berdasarkan jaring semantik. Edward Feigenbaum dan Julian Feldman menerbitkan Computeks and Thought, kumpulan artikel pertama tentang AI.

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

▪ Definisi Supervised Learning

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

▪ Klasifikasi

Klasifikasi adalah pembagian sesuatu menurut kelas-kelas (class). Menurut Ilmu Pengetahuan, Klasifikasi merupakan proses pengelom-

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

- Regresi

Regresi adalah metode analisis statistik yang digunakan untuk melihat pengaruh antara dua ataupun lebih variabel. Regresi adalah membahas masalah ketika variabel output adalah nilai riil atau berkelanjutan, seperti "gaji" atau "berat". Banyak model yang berbeda dapat digunakan makan, yang paling sederhana adalah regresi linier. Ia mencoba untuk menyesuaikan data dengan hyper-plane terbaik yang melewati poin.

- Unsupervised Learning

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

- Data set

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

untuk membangun solusi AI harus kembali ke tahap pengumpulan data.

- Training Set

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

- Testing set

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

1.14.2 Praktek

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

- Buka prompt anaconda lalu ketikkan conda scikit-learn
- Biarkan proses berjalan hingga selesai

```
(base) C:\Users\DiffaAl21>conda install scikit-learn
```

Gambar 1.74 Install library scikit

```
(base) C:\Users\Diffa121>conda install scikit-learn
Collecting package metadata (repodata.json): done
Solving environment: done

## Package Plan ##

environment location: C:\Users\Diffa121\Anaconda3

added / updated specs:
- scikit-learn

The following packages will be downloaded:

  package          build
ca-certificates-2020.1.1      0         165 KB
conda-4.8.2                   py37_0     3.1 MB
scikit-learn-0.20.3           py37h343c172_0   4.7 MB
                                           Total:    7.9 MB

The following packages will be UPDATED:

  ca-certificates              2019.11.27-0 --> 2020.1.1-0
  conda                        4.7.12-py37_0 --> 4.8.2-py37_0
  scikit-learn                  0.20.1-py37h343c172_0 --> 0.20.3-py37h343c172_0

Proceed ([y]/n)?
```

Downloading and Extracting Packages
ca-certificates-2020 | 165 KB | #####| 100%
conda-4.8.2 | 3.1 MB | #####| 100%
scikit-learn-0.20.3 | 4.7 MB | #####| 100%
Preparing transaction: done
Verifying transaction: done
Executing transaction: done

Gambar 1.75 Proses jika berhasil

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

- Ketikkan script berikut di spyder, lalu run

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

```
In [1]: runfile('E:/Difa/Tugas Kuliah/Tingkat 3/Semester 6/Kecerdasan Buatan/src1/test.py', wdir='E:/Difa/Tugas Kuliah/Tingkat 3/Semester 6/Kecerdasan Buatan/src1')
[[ 0.  0.  5. ... 0.  0.  0.]
 [ 0.  0.  0. ... 10.  0.  0.]
 [ 0.  0.  0. ... 16.  9.  0.]
 ...
 [ 0.  0.  1. ... 6.  0.  0.]
 [ 0.  0.  2. ... 12.  0.  0.]
 [ 0.  0.  10. ... 12.  1.  0.]]
In [2]: |
```

Gambar 1.76 hasil test.py

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

```
1 #Import fungsi datasets dari library sklearn
2 from sklearn import datasets
3
4 #Memasukkan data dari datasets iris ke variable iris
5 iris = datasets.load_iris()
6
7 #Memasukkan data dari datasets digits ke variable digits
8 digits = datasets.load_digits()
9
10 #Mengimport sebuah Support Vector Machine(SVM) yang merupakan
11 #algoritma classification yang akan diambil dari Scikit –
12 #Learn.
13 from sklearn import svm
14 #Mendeklarasikan suatu value yang bernama clf yang berisi
15 #gamma.
16 clf = svm.SVC(gamma=0.001, C=100.)
17 #Estimator clf (for classifier)
18 clf.fit(digits.data[:-1], digits.target[:-1])
19
20 #Menunjukkan prediksi angka baru
21 hasil = clf.predict(digits.data[-1:])
22
23 #Menampilkan
24 print(hasil)
```

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

Gambar 1.77 variabel explorer test2.py

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

```
1 %%Cara Dump Pertama
2
3
4
5 #Mengimport sebuah Support Vector Machine(SVM) yang merupakan
   algoritma classification yang akan diambil dari Scikit-
   Learn.
6 from sklearn import svm
7 #Import fungsi datasets dari library sklearn
8 from sklearn import datasets
9
10 #Mendefinisikan clf dengan fungsi svc dari library svm
11 clf = svm.SVC()
12
13 #Mengisi variable x dan y dengan data dari datasets
14 X, y = datasets.load_iris(return_X_y=True)
15
16 #Estimator clf (for classifier)
17 clf.fit(X, y)
18
19 #Mengimport Library pickle
20 import pickle
21
22 #Menyimpan hasil dari clf kedalam sebuah dump
23 s = pickle.dumps(clf)
24
25 #Memanggil dump yang dihasilkan pickle lalu memasukkan hasil
   dumpnya ke variable
26 clf2 = pickle.loads(s)
27
28
29 #Memprediksi angka yang akan muncul
30 clf2.predict(X[0:1])
31
32 #Menampilkan data prediksi
33 print(y[0])
34
35 %%Cara Dump Kedua
36 # Digunakan untuk memanggil class svm dari library sklearn
37 from sklearn import svm
38
39 # Digunakan untuk class datasets dari library sklearn
40 from sklearn import datasets
41
42 # membuat variabel clf , dan memanggil class svm dan fungsi
   SVC
43 clf = svm.SVC()
44
45 #Mengambil dataset iris dan mengembalikan nilainya.
46 X, y = datasets.load_iris(return_X_y=True)
47
48 #Perhitungan nilai label
49 clf.fit(X, y)
```

```

51
52 #memanggil class dump dan load pada library joblib
53 from joblib import dump, load
54
55
56 #Menyimpan model kedalam 1174066.joblib
57 dump( clf , '1174083.joblib' )
58
59 #Memanggil model 1174066
60 hasil = load('1174083.joblib')
61 hasil.predict(X[0:1])
62
63 # Menampilkan Model yang dipanggil sebelumnya

```

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

Gambar 1.78 variabel explorer test3.py

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

```

1
2 # memanggil library numpy dan dibuat alias np
3 import numpy as np
4
5 #Memanggil class random_projection pada library sklearn
6 from sklearn import random_projection
7
8 #Membuat variabel rng, dan mendefinisikan np, fungsi random dan
9 #attr RandomState kedalam variabel
10 rng = np.random.RandomState(0)
11
12 # membuat variabel X, dan menentukan nilai random dari 10 –
13 # 2000
14 X = rng.rand(10, 2000)
15
16 #menyimpan hasil nilai random sebelumnya, kedalam array, dan
17 #menentukan typedatanya sebagai float32
18 X = np.array(X, dtype='float32')
19
20 # Mengubah data tipe menjadi float64
21 X.dtype
22
23 #membuat variabel transformer, dan mendefinisikan
#classrandom_projection dan memanggil fungsi
#GaussianRandomProjection

```

```

21 transformer = random_projection.GaussianRandomProjection()
22 # membuat variabel baru dan melakukan perhitungan label pada
23 X_new = transformer.fit_transform(X)
24 # Mengubah data tipe menjadi float64
25 X_new.dtype
26 # Menampilkan isi variabel X_new
27 print(X_new)

```

Name	Type	Size	Value
X	float32	(10, 2000)	[[0.5488135 0.71518934 0.60276335 ... 0.4801078 0.64386404 0.5017731 ...
X_new	float64	(10, 1973)	[[0.16482812 0.90149112 0.16933195 ... -0.83564827 -0... -0 ...
digits	utils.Bunch	5	Bunch object of sklearn.utils module
iris	utils.Bunch	6	Bunch object of sklearn.utils module
y	int32	(150,)	[0 0 0 ... 2 2 2]

Gambar 1.79 variabel explorer test4.py

```

In [2]: runfile('E:/Difa/Tugas Kuliah/Tingkat 3/Semester 6/Kecerdasan Buatan/src1/test4.py', wdir='E:/Difa/Tugas Kuliah/Tingkat 3/Semester 6/Kecerdasan Buatan/src1')
[[ 0.21633181  0.46361945 -0.0893285 ... -0.20120145 -0.89976685
-0.07772901]
[ -0.06005892  0.37376534 -0.15962054 ...  0.2372093 -0.74607398
-0.76761648]
[ 0.40926478  0.43447018  0.3673702 ... -0.2036962 -0.23671443
-0.28807406]
...
[ 0.40032337  0.93748013 -0.38694532 ... -0.28596616 -0.58826772
-1.24867457]
[ 0.46108146  0.4797264 -0.34775959 ...  0.22005661 -0.64094577
-0.73309616]
[ 0.17155332  0.64167509 -0.51863713 ... -0.41102532  0.15211431
-0.28959053]]
```

Gambar 1.80 hasil test4.py

1.14.3 Penanganan Error

1. Screenshoots Error

ModuleNotFoundError: No module named 'joblib'

Gambar 1.81 error

2. Jenis Error

- Not Module name "joblib"

3. Cara Penanganan Error

- install library joblib

1.14.4 Bukti tidak plagiat

1.14.5 Link Youtube

1.15 1174074 — Mochamad Arifqi Ramadhan

1.15.1 Teori

1. Definisi Sejarah dan Perkembangan Kecerdasan Buatan

Kecerdasan Buatan biasa disebut dengan istilah AI (Artificial Intelligence). AI sendiri merupakan suatu cabang dalam bisnis sains komputer sains dimana mengkaji tentang bagaimana cara untuk men lengkap se buah komputer dengan kemampuan atau kepintaran layaknya atau mirip dengan yang dimiliki manusia. Sebagai contoh, sebagaimana komputer dapat berkomunikasi dengan pengguna baik menggunakan kata, suara maupun lain sebagainya. Dengan kemampuan ini, diharapkan komputer mampu mengambil keputusan sendiri untuk berbagai kasus yang ditemui nya kemudian itulah yang disebut dengan kecerdasan buatan. Kecerdasan buatan adalah kemampuan komputer digital atau robot yang dikendalikan komputer untuk melakukan tugas yang umumnya dikaitkan dengan sesuatu yang cerdas. Istilah ini sering diterapkan pada proyek pengembangan sistem yang diberkahi dengan karakteristik proses intelektual manusia, seperti kemampuan untuk berpikir, menemukan makna, menggeneralisasi, atau belajar dari pengalaman masa lalu.

Kecerdasan Buatan adalah salah satu bidang studi yang berhubungan dengan pemanfaatan mesin untuk memecahkan persoalan yang rumit dengan cara lebih manusiawi dan lebih bisa di pahami oleh manusia. Kecerdasan buatan makin canggih dengan kemampuan komputer dalam memperbarui pengetahuannya dengan banyaknya testing dan perkembangan target analisa. Untuk kecerdasan buatan ada banyak contoh dan jenisnya. Salah satu contoh yang paling terkenal dari Artificial Intelligence ialah Google Assistant. Google Assistant digunakan untuk kemudahan user dalam menemukan berbagai hal maupun penyetelan langsung terhadap smartphone yang digunakan dan masih banyak lagi.

Sejarah Kecerdasan Buatan, Artificial intelligence merupakan inovasi baru di bidang ilmu pengetahuan. Mulai terbentuk sejak adanya komputer modern dan kira-kira terjadi sekitaran tahun 1940 dan 1950. Ilmu pengetahuan komputer ini khusus ditujukan dalam perancangan otomatisasi

tingkah laku cerdas dalam sistem kecerdasan komputer. Pada awal 50-an, studi tentang “mesin berpikir” memiliki berbagai nama seperti cybernetics, teori automata, dan pemrosesan informasi. Pada tahun 1956, para ilmuwan jenius seperti Alan Turing, Norbert, Wiener, Claude Shannon dan Warren McCullough telah bekerja secara independen dibidang cybernetics, matematika, algoritma dan teori jaringan. Namun, seprang ilmuwan komputer dan kognitif John McCarthy adalah orang yang dating dengan ide untuk bergabung dengan upaya penelitian terpisah ini kedalam satu bidang yang akan mempelajari topic baru untuk imajinasi manusia yaitu kecerdasan buatan. Dia adalah orang yang menciptakan istilah tersebut dan kemudian mendirikan laboratorium Kecerdasan Buatan di MIT dan Stan ford.

Perkembangan Kecerdasan Buatan, Teknologi Artificial Intelligence semakin ramai dibahas dalam berbagai diskusi teknologi di seluruh dunia. Menurut kebanyakan orang, pekerjaan seperti kasir, operator telepon, pengendara truk, dan lainnya sangat berpeluang besar untuk tergantikan oleh Artificial Intelligence. Mengapa terjadi hal demikian? dikarenakan memang bahwa AI lebih unggul dalam hal kinerja, fitur dan lain sebagainya. Namun, dalam beberapa aspek memang pekerja manusia masih unggul dibandingkan AI itu sendiri. Para generasi muda yang ada di dunia terutama di daerah Asia terlihat sudah memahami fungsi dan efek dari AI dalam kehidupan kita sehari-hari. Berdasarkan survei yang dilakukan oleh Microsoft, terdapat 39 persen responden yang mempertimbangkan untuk menggunakan mobil tanpa pengemudi dan 36 persen lainnya setuju bahwa robot masa depan dengan software untuk beroperasi mampu meningkatkan produktivitas. Dari survey tersebut kita sebagai pengguna AI harus lebih bijaksana dalam pengembangan dan penggunaan dari AI sehingga tanpa memberikan efek samping terhadap etos kerja dan keseharian kita sebagai pengguna dalam kehidupan sehari-hari.

2. Definisi Supervised learning, klasifikasi, regresi, unsupervised learning, dataset, training set dan testing set.

- **Supervised Learning**

Supervised Learning merupakan tugas pengumpulan data untuk menyimpulkan fungsi dari data pelatihan berlabel. Data pelatihan terdiri dari serangkaian contoh pelatihan. Dalam supervised learning, setiap contoh adalah pasangan yang terdiri dari objek input (biasanya vektor) dan nilai output yang diinginkan(juga disebut sinyal pengawasan super).

- **Klasifikasi**

Klasifikasi adalah pembagian sesuatu menurut kelas-kelas. Klasifikasi merupakan proses dari pengelompokan benda berdasarkan ciri-ciri

persamaan dan juga perbedaan. Dalam masalah klasifikasi, kami mencoba memprediksi sejumlah nilai terpisah.

- Regresi

Regresi adalah metode analisis statistik yang digunakan untuk melihat pengaruh antara dua ataupun lebih variabel. Regresi adalah membahas masalah ketika variabel output adalah nilai riil atau berkelanjutan, seperti "gaji" atau "berat".

- Unsupervised learning

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

- Data set

Data set objek yang merepresentasikan data dan relasinya di memory. Strukturnya mirip dengan data di database. Dataset berisi koleksi dari datatable dan datarelation.

- Training Set

Training Set adalah set digunakan oleh algoritma klasifikasi. Dapat dicontohkan dengan decision tree, bayesian, neural network dll. Semuanya dapat digunakan untuk membentuk sebuah model classifier

- Testing Set

Testing Set merupakan set yang digunakan untuk mengukur sejauh mana classifier berhasil melakukan klasifikasi dengan benar. Dapat berfungsi sebagai meterai persetujuan, dan Anda tidak menggunakan-nya sampai akhir.

1.15.2 Praktek

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

Gambar 1.82 Instalasi Package Scikit Learn



Gambar 1.83 Isi Variabel Explorer

2. Mencoba loading an example dataset

```
1 #%% Mencoba loading an example dataset
2 from sklearn import datasets # Digunakan Untuk Memanggil
3 # class datasets dari library sklearn
4 iris = datasets.load_iris() # Menggunakan contoh datasets
5 # iris
6 x = iris.data # Menyimpan nilai data sets iris
7 pada variabel x
```

3. Mencoba Learning dan predicting

```
1 y = iris.target # Menyimpan nilai data label iris  
2 pada variabel y  
3  
4 #%%Mencoba Learning dan predicting  
5 from sklearn.neighbors import KNeighborsClassifier #Digunakan  
Untuk Memanggil fungsi KNeighborsClassifier  
6 # pada  
7 class sklearn dan library sklearn  
8 import numpy as np # memanggil library numpy dan dibuat alias  
np  
9 knn=KNeighborsClassifier(n_neighbors=1) #membuat variabel knn  
, dan memanggil fungsi KNeighborsClassifier  
10 #dan mendefinisikan k  
11 -nya adalah 1  
knn.fit(x,y) #Perhitungan  
12 matematika library kkn  
13 a=np.array([1.0 ,2.0 ,3.0 ,4.0]) #Membuat Array  
14 a = a.reshape(1, -1) #Mengubah Bentuk  
15 Array jadi 1 dimensi
```

4. Mencoba Model Persistence

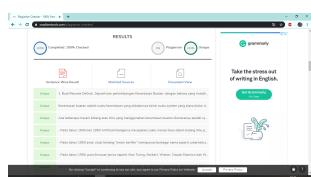
```
1 hasil = knn.predict(a) #Memanggil fungsi  
2 prediksi  
3 print(hasil) #menampilkan hasil  
4 prediksi  
5  
#%% Model Persistense  
6 from sklearn import svm # Digunakan untuk memanggil class svm  
7 dari library sklearn
```

```
6 from sklearn import datasets # Digunakan untuk class datasets  
    dari library sklearn  
7 clf = svm.SVC()             # membuat variabel clf , dan  
    memanggil class svm dan fungsi SVC  
8 X, y = datasets.load_iris(return_X_y=True) #Mengambil dataset  
    iris dan mengembalikan nilainya .  
9 clf.fit(X, y)               #Perhitungan nilai label  
10  
11 from joblib import dump, load #memanggil class dump dan load  
    pada library joblib
```

5. Mencoba Conventions

```
1 dump(clf, '1174074.joblib') #Menyimpan model kedalam 1174074.joblib
2 hasil = load('1174074.joblib') #Memanggil model 1174074
3 print(hasil) # Menampilkan Model yang dipanggil sebelumnya
4
5 ##% Conventions
6 import numpy as np # memanggil library numpy dan dibuat alias np
7 from sklearn import random_projection #Memanggil class random_projection pada library sklean
8
9 rng = np.random.RandomState(0) #Membuat variabel rng , dan mendefinisikan np, fungsi random dan attr RandomState kedalam variabel
10 X = rng.rand(10, 2000) # membuat variabel X, dan menentukan nilai random dari 10 – 2000
11 X = np.array(X, dtype='float32') #menyimpan hasil nilai random sebelumnya, kedalam array , dan menentukan typedatanya sebagai float32
12 X.dtype # Mengubah data tipe menjadi float64
```

1.15.3 Bukti Tidak Plagiat



Gambar 1.84 Bukti Tidak Melakukan Plagiat

1.16 Advent Nopele Olansi Damiahan Sihite (1174089)

1.16.1 Teori

1. Definisi Kecerdasan buatan

Kecerdasan Buatan biasa disebut dengan istilah AI (Artificial Intelligence). Kecerdasan Buatan adalah salah satu bidang studi yang berhubungan dengan pemanfaatan mesin untuk memecahkan persoalan yang rumit dengan cara lebih manusiawi dan lebih bisa di pahami oleh manusia. Kecerdasan buatan makin canggih dengan kemampuan komputer dalam memperbarui pengetahuannya dengan banyaknya testing dan perkembangan target analisa. Untuk kecerdasan buatan ada banyak contoh dan jenisnya. Salah satu contoh yang paling terkenal dari Artificial Intelligence ialah Google Assistant. Google Assistant digunakan untuk kemudahan user dalam menemukan berbagai hal maupun penyetelan langsung terhadap smartphone yang digunakan dan masih banyak lagi.

2. Sejarah dan Perkembangan Kecerdasan Buatan

- Pada tahun 1943, pekerjaan pertama yang dikenal sebagai AI telah dilakukan oleh Warren McCulloch dan juga Walter Pitts yang dinamakan sebagai artificial neurons
- Pada tahun 1955, Allen Newell dan Herbert A. Simon membuat program kecerdasan buatan pertama yang dinamakan Logic Theorist
- Pada tahun 1972, robot pertama dibuat di jepang dengan nama Wabot-1 dengan kecerdasan buatan
- Pada tahun 1980, muncul bidang baru dari kecerdasan buatan yaitu Expert System yang membantu dalam pemberian keputusan
- Tahun 1997, IBM deep blue mengalahkan juara catur dunia Gary Kasparov dan menjadi komputer pertama yang mengalahkannya
- Tahun 2006, perusahaan sudah mulai menerapkan kecerdasan buatan pada produknya seperti Netflix dan Twitter.
- Tahun 2018, Project Debater dari IBM melakuakn debat tentang topik yang kompleks dan berakhir dengan hasil memuaskan

3. Definisi Supervised Learning

Supervised Learning adalah tugas pengumpulan data untuk menyimpulkan fungsi dari data pelatihan berlabel. Data pelatihan terdiri dari serangkaian contoh pelatihan. Dalam supervised learning, setiap contoh adalah pasangan yang terdiri dari objek input (biasanya vektor) dan nilai output yang diinginkan(juga disebut sinyal pengawasan super).

4. Klasifikasi Supervised Learning

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

5. Regresi dan Unsupervised Learning

Regresi adalah metode analisis statistik yang digunakan untuk melihat pengaruh antara dua ataupun lebih variabel. Regresi adalah membahas masalah ketika variabel output adalah nilai riil atau berkelanjutan, seperti "gaji" atau "berat". Banyak model yang berbeda dapat digunakan makan, yang paling sederhana adalah regresi linier. Ia mencoba untuk menyesuaikan data dengan hyper-plane terbaik yang melewati poin.

Unsupervised Learning berbeda dengan Supervised Learning. Perbedaannya ialah unsupervised learning tidak memiliki data latih, sehingga dari data yang ada kita mengelompokan data tersebut menjadi 2 ataupun 3 bagian dan seterusnya. Unsupervised Learning adalah pelatihan algoritma kecerdasan buatan (AI) menggunakan informasi yang tidak diklasifikasikan atau diberi label dan memungkinkan algoritma untuk bertindak atas informasi tersebut tanpa bimbingan. Algoritma Unsupervised Learning dapat melakukan tugas pemrosesan yang lebih kompleks dari pada sistem pembelajaran yang diawasi

6. Dataset

Dataset adalah objek yang merepresentasikan data dan juga relasi yang ada di memory. Strukturnya mirip dengan data di database, namun bedanya dataset berisi koleksi dari data table dan data relation. mendapatkan data yang tepat berarti mengumpulkan atau mengidentifikasi data yang berkorelasi dengan hasil yang ingin Anda prediksi; yaitu data yang berisi sinyal tentang peristiwa yang Anda pedulikan. Data harus diselaraskan dengan masalah yang Anda coba selesaikan.

7. Training Set

Training Set adalah set digunakan oleh algoritma klasifikasi . Dapat di-contohkan dengan : decision tree, bayesian, neural network dll. Semuanya dapat digunakan untuk membentuk sebuah model classifier. Menjalankan

pelatihan yang diatur melalui jaringan saraf mengajarkan pada net cara menimbang berbagai fitur, menyesuaikan koefisien berdasarkan kemungkinan mereka meminimalkan kesalahan dalam hasil Anda. Koefisien-koefisien tersebut, juga dikenal sebagai parameter, akan terkandung dalam tensor dan bersama-sama mereka disebut model, karena mereka mengkodekan model data yang mereka latih. Mereka adalah takeaways paling penting yang akan Anda dapatkan dari pelatihan jaringan saraf.

8. Testing Set

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

1.16.2 Instalasi

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

Gambar 1.85 Instalasi Package Scikit Learn

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

Gambar 1.86 Isi Variabel Explorer

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

1 %%Loading an example dataset

```

1 from sklearn import datasets # Load library dataset
2 iris = datasets.load_iris()
3 # variable iris diisi dengan contoh data
4 a = iris.data # Menyimpan value data ke variable A
5 b = iris.target # Menyimpan value data ke variable B

```

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

```

1 %% Learning dan predicting
2 from sklearn.neighbors import KNeighborsClassifier
3 #Load library
4 import numpy as np
5 #load library
6
7 knn = KNeighborsClassifier(n_neighbors=1)
8 #mendefinisikan variabel bernama kkn, dan memanggil fungsi
     KNeighborsClassifier
9 # dan memberikan value 1
10 knn.fit(a,b) # perhitungan library knn
11
12 x = np.array([1.0 ,2.0 ,3.0 ,4.0])
13 # membuat array
14 x = x.reshape(1,-1)
15 #Convert array menjadi 1 dimensi
16
17 hasil = knn.predict(x)
18 #Memanggil fungsi predict dari KNN
19 print(hasil)
20 #menampilkan value dari variable hasil

```

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

```

1 %% Model Persistense
2 from sklearn import svm
3 # Load library
4 from sklearn import datasets
5 # Load Library
6 clf = svm.SVC()
7 # mendefinisikan variabel clf, dan memanggil fungsi SVC dari
     class svm
8 a, b = datasets.load_iris(return_X_y=True)
9 #Variable a dan b diisi dengan dataset iris dan mengembalikan
     nilainya.
10 clf.fit(a, b)
11 #memanggil fungsi fit dari clf
12
13 from joblib import dump, load
14 #Load library
15 dump(clf, '1174079.joblib')
16 #Menyimpan model kedalam 1174079.joblib
17 hasil = load('1174079.joblib')

```

```
18 #memuat model 1174079
19 print(hasil) # Menampilkan Hasil
```

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

```
1 %% Conventions
2 import numpy as np
3 # Load Library
4 from sklearn import random_projection
5 #Load class random_projection dari library sklearn
6
7 rng = np.random.RandomState(0)
8 #Membuat variabel rng, dan mendefinisikan np, fungsi random dan
  attr RandomState kedalam variabel
9 X = rng.rand(10, 2000)
10 # membuat variabel X, dan menentukan nilai random dari 10 -
  2000
11 X = np.array(X, dtype='float32')
12 #menyimpan hasil nilai random sebelumnya, kedalam array, dan
  menentukan typedatanya sebagai float32
13 X.dtype
14 # Mengubah data tipe menjadi float64
15
16 transformer = random_projection.GaussianRandomProjection()
17 #membuat variabel transformer, dan mendefinisikan
  classrandom_projection dan memanggil fungsi
  GaussianRandomProjection
18 X_new = transformer.fit_transform(X)
19 # membuat variabel baru dan melakukan perhitungan label pada
  variabel X
20 X_new.dtype
21 # Mengubah data tipe menjadi float64
22 print(X_new)
23 # Menampilkan isi variabel X_new
```

1.16.3 Penanganan Error

1. ScreenShoot Error

```
In [2]: from sklearn import datasets # Load Library dataset
...: iris = datasets.load_iris()
...: print("Dataset yang akan diolah")
...: a = iris.data # Menyimpan value data ke variable A
...: b = iris.target # Menyimpan value data ke variable B
tracesback (most recent call last):
File "<ipython-input-2-55cf92ea2d3>", line 5, in <module>
    b = iris.target # Menyimpan value data ke variable B
NameError: name 'iri' is not defined
```

Gambar 1.87 No Module Named Numpya

2. Tuliskan Kode Error dan Jenis Error

- ModuleNotFoundError

3. Cara Penangan Error

- ModuleNotFoundError

Mengecek Typo dan menulis kembali library yang akan diimport

1.16.4 Bukti Tidak Plagiat



Gambar 1.88 Bukti Tidak Melakukan Plagiat Chapter 1

1.17 Handi Hermawan (1174080)

1.17.1 Teori

1. Definisi Kecerdasan buatan

Kecerdasan buatan atau Artificial intelligence merupakan kecerdasan yang ditambahkan kedalam suatu sistem yang diatur secara ilmiah. Kecerdasan buatan dibuat untuk mengantikan pekerjaan yang dilakukan oleh manusia menjadi dikerjakan oleh sistem.

2. Sejarah Kecerdasan Buatan

- Abad 17, Rene Descartes berkata bahwa tubuh hewan adalah sekumpulan mesin yang rumit.
- 1642, Blaise Pascal menciptakan mesin penghitung digital mekanis pertama.
- Abad 19, Charles Babbage dan Ada Lovelace bekerja di program penghitung mekanis.
- 1950, John McCarthy membuat istilah “Kecerdasan Buatan”.
- 1960-1970, Joel Moses membuat program yang pertama kali sukses dalam bidang matematika.
- 1980, jaringan saraf digunakan secara meluas dengan algoritme perambatan balik.
- 2004, DARPA membuat kendaraan yang bisa dijalankan sendiri tanpa manusia.

3. Perkembangan kecerdasan buatan

- Masa persiapan (1943-1946) Warren McCulloch dan Walter Pitt mengemukakan tiga hal : pengetahuan fisiologi dasar dan fungsi sel syaraf

dalam otak, analisa formal tentang logika proposisi, dan teori komputasi Turing.

Pada tahun 1950, Norbert Wiener membuat penelitian mengenai prinsip-prinsip teori feedback.

Pada tahun 1956, John McCarthy meyakinkan Minsky, Claude Shannon dan Nathaniel Rochester untuk membantunya melakukan penelitian dalam bidang Otomata, Jaringan Syaraf dan pembelajaran intelejensi.

- Awal perkembangan (1952-1969) Pada tahun 1958, McCarthy di MIT AI Lab Memo No.1 mendefinisikan bahasa pemrograman tingkat tinggi yaitu LISP,
- Pada tahun 1959, Nathaniel Rochester dari IBM dan mahasiswa-mahasiswanya mengeluarkan program kecerdasan buatan yaitu Geometry Theorem Prover.
- Pada tahun 1963, program yang dibuat James Slagle mampu menyelesaikan masalah integral tertutup untuk mata kuliah Kalkulus. Pada tahun 1986, program analogi buatan Tom Evan menyelesaikan masalah analogi geometris yang ada pada tes IQ.
- Perkembangan Kecerdasan Buatan Melambat (1969-1979) Bruce Buchanan dan Joshua Lederberg yang membuat program untuk memecahkan masalah struktur molekul dari informasi yang didapatkan dari spectrometer massa.
- AI Menjadi sebuah industri Industrialisasi kecerdasan buatan diawali dengan ditemukannya sistem pakar yang dinamakan R1 yang mampu mengkonfigurasi sistem-sistem computer baru.
- Kembalinya Jaringan Syaraf Tiruan (1986-sekarang) Pada tahun 1985-an setidaknya empat kelompok riset menemukan kembali algoritma belajar propagasi balik (Back-Propagation Learning). Algoritma ini berhasil diimplementasikan ke dalam bidang ilmu computer dan psikologi.

4. Definisi Supervised Learning

Supervised Learning merupakan cabang dari Artificial Intelligence. supervised learning adalah suatu ilmu yang mempelajari perancangan dan pengembangan algoritma.

5. Klasifikasi Supervised Learning

- Logistic regression.
- K-nearest neighbors.
- Support vector machine (SVM)
- Naive Bayes.
- Decision tree classification.
- Random forest classification.

6. Regresi dan Unsupervised Learning

Regresi merupakan sebuah metode analisis statistic yang digunakan untuk mengetahui pengaruh antara dua variable atau lebih.

Untuk mempelajari Unsupervised learning kita tidak perlu data training untuk melakukan prediksi maupun klasifikasi.

7. Dataset

Dataset merupakan objek yang mempresentasikan data dan relasinya pada memori.

8. Training Set

Training Set merupakan bagian dari dataset untuk membuat prediksi atau menjalankan fungsi dari sebuah algoritma Machine Learning.

9. Testing Set

Testing set digunakan untuk mengukur apakah classifier berhasil melakukan klasifikasi dengan benar.

1.17.2 Instalasi

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

Gambar 1.89 Instalasi Package Scikit Learn

Name	Type	Size	Value
digits	utils.Bunch	5	Bunch object of sklearn.utils module
iris	utils.Bunch	6	Bunch object of sklearn.utils module

Gambar 1.90 Isi Variabel Explorer

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

1
2 #%% Learning dan predicting

```
from sklearn.neighbors import KNeighborsClassifier
```

```

4 #Load library
5 import numpy as np

```

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

```

1
2 knn = KNeighborsClassifier(n_neighbors=1)
3 #mendefinisikan variabel bernama kkn, dan memanggil fungsi
4     KNeighborsClassifier
5 # dan memberikan value 1
6 knn.fit(a,b) # perhitungan library knn
7
8 x = np.array([1.0 ,2.0 ,3.0 ,4.0])
9 # membuat array
10 x = x.reshape(1,-1)
11 #Convert array menjadi 1 dimensi

```

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

```

1 #Memanggil fungsi predict dari KNN
2 print(hasil)
3 #menampilkan value dari variable hasil
4
5 %% Model Persistense
6 from sklearn import svm
7 # Load library
8 from sklearn import datasets
9 # Load Library
10 clf = svm.SVC()

```

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

```

1 #Variable a dan b diisi dengan dataset iris dan mengembalikan
2     nilainya.
3 clf.fit(a, b)
4 #memanggil fungsi fit dari clf
5
6 from joblib import dump, load
7 #Load library
8 dump(clf, '1174080.joblib')
9 #Menyimpan model kedalam 1174080.joblib
10 hasil = load('1174080.joblib')
11 #memuat model 1174080
12 print(hasil) # Menampilkan Hasil

```

1.17.3 Penanganan Error

1. ScreenShoot Error

```
File "D:\Semester 6\Kecerdasan Buatan\KBC-master\src\1174000\1\1174000.py", line 9
    from sklearn.neighbors import *
          ^
SyntaxError: invalid syntax
```

Gambar 1.91 Import Error

2. Tuliskan Kode Error dan Jenis Error

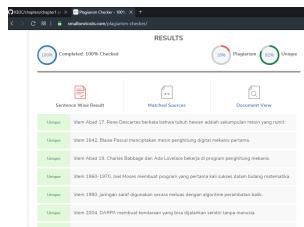
- Import Error

3. Cara Penangan Error

- Import Error

Dengan Menginstall Library Yang Tidak Ditemukan

1.17.4 Bukti Tidak Plagiat



Gambar 1.92 Bukti Tidak Melakukan Plagiat

1.17.5 Link Youtube

<https://youtu.be/V4NWVnF2bo>

1.18 Muhammad Abdul Gani Wijaya (1174071)

1.18.1 Teori Soal 1

1.18.1.1 Definisi Kecerdasan Buatan Kecerdasan buatan merupakan kecerdasan yang diterapkan pada teknologi, dan diatur serta dikembangkan dalam bidang ilmiah, sebagai bentuk kecerdasan yang dibuat dari kecerdasan ilmu ilmiah yang telah ada.

1.18.1.2 Sejarah Kecerdasan Buatan Kecerdasan buatan telah ada pada di zaman kuno dalam cerita dongeng tentang atau benda buatan yang diberkahi dengan kecerdasan atau kesadaran oleh pengrajin dan pembuatnya. Kecerdasan buatan modern menggambarkan proses cara berpikir manusia secara mekanis. Kecerdasan buatan memuncak pada penemuan komputer digital yang sudah dapat diprogram pada tahun 1940-an, sebuah mesin yang didasarkan pada perhitungan dan esensi penalaran matematika. Perangkat ini dan ide-ide nya menginspirasi para ilmuwan untuk mulai serius membahas kemungkinan membangun otak elektronik atau yang sekarang dikenal dengan kecerdasan buatan/artificial intelligence.

1.18.1.3 Perkembangan Kecerdasan Buatan

1. Awal Mula Kecerdasan Buatan (1943 – 1955)

- Awal Mula AI dikerjakan oleh McCulloh dan Pitts yang membuat Neuron buatan dengan menirukan cara kerja neuron manusia dengan logika proposisional. Project tersebut bisa menyelesaikan fungsi komputasi dengan struktur neuron network.
- Hebbian learning, memperkenalkan aturan-aturan sederhana untuk meng-update kekuatan antar neuron.
- Minsky dan Edmonds berhasil membangun komputer neural network pertama pada 1950.
- Allan Turing dianggap sebagai orang pertama yang mengeluarkan pikiran mengenai Artificial Intelligence secara utuh pada artikelnya yang berjudul “Computing machinery and Intelligent” pada tahun 1950.

2. Kelahiran Kecerdasan Buatan (1956)

- McCarthy menginisiasi Dartmouth Workshop pada tahun 1956 dan melahirkan suatu bidang baru yaitu “Artificial Intelligence”.

3. Awal mula AI yang penuh dengan antusias dan harapan besar di masa depan (1952 – 1969)

- Sebuah tahap pengembangan aplikasi AI yang sukses jika dibandingan dengan program komputer primitif. Banyak dari aplikasi AI yang berhasil sehingga muncul istilah “evolusi mesin”

4. AI menjadi industry (1980 – sekarang)

- Aplikasi komersial pertama yang menggunakan sistem pakar bernama R1 yang digunakan oleh perusahaan Amerika (1982).
- Jepang juga membentuk proyek jangka panjang menggunakan komputer cerdas dengan berbasis Prolog.

5. Kecerdasan Buatan menjadi disiplin ilmu (1987 – sekarang)

6. AI menampakkan diri di semua bidang (1995 – sekarang)

1.18.2 Teori Soal 2

1.18.2.1 Definisi Supervised Learning Supervised Learning adalah pembelajaran dengan diiringi oleh supervisornya. Maksud dari supervisornya adalah label pada tiap data nya. Maksud dari label adalah tag oada data yang ditambahkan dalam machine learning model. Pada contoh gambar burung di tag burung” pada setiap masing masing image burung dan gambar ikan di tag “ikan” pada setiap masing masing gambar ikan. Machine learning kategori dapat berupa clasification (“ikan”, “kucing”, “burung”, dsb) dan regression (berat, tinggi dsb). Supervised learning digunakan untuk memprediksi pola-pola dengan contoh data yang diberikan, jadi pola yang terbentuk adalah hasil pembelajaran dari data lengkap tersebut. Tentunya jika kita mengisi data baru, lalu setelah kita melakukan ETL (Extract Transform Load) maka kita akan mendapat info feature-feature dari sample baru tersebut. Kemudian dari feature-feature tersebut akan di compare dengan pattern clasification dari model yang didapat dari labeled data. Setiap label dicompare sampai selesai, dan yang memiliki percentage lebih banyak diambil sebagai prediksi akhir.

1.18.2.2 Definisi Klasifikasi Klasifikasi adalah penggolongan atau pengelompokan. Menurut KBBI klasifikasi adalah penyusunan bersistem dalam kelompok atau golongan menurut kaidah atau standar yang ditetapkan. Harrolds Librarians Glossary menjelaskan bahwa klasifikasi adalah pengelompokkan benda secara logis menurut ciri-ciri kesamaannya. Lalu klasifikasi menurut Sulistyo Basuki yang menjelaskan bahwa klasifikasi merupakan proses yang digunakan untuk pengelompokan/pengumpulan benda atau entitas yang sama, serta memisahkan benda atas entitas yang tidak sama. Namun secara umum klasifikasi adalah suatu kegiatan yang mengelompokkan benda-benda yang memiliki beberapa ciri-ciri yang sama dan memisahkan benda yang tidak sama.

1.18.2.3 Definisi Regresi Regresi merupakan suatu metode analisis statistik yang digunakan untuk melihat pengaruh antara dua atau lebih banyak variabel. Hubungan variabel-variabel tersebut bersifat fungsional yang diwujudkan dalam suatu model yang matematis. Analisis regresi pada variabel dibagi menjadi dua, yaitu variabel respons (response variable) atau variabel bergantung (dependent variable), dan variabel explanatory atau penduga (predictor variable) atau disebut variabel bebas (independent variable).

1.18.2.4 Definisi Unsupervised Learning Unsupervised learning memiliki keunggulan dari unsupervised learning. Jika unsupervised learning memiliki label sebagai dasar prediksi serta membuat clasification dan regression algorithm memungkinkan. Namun pada realitanya, data real itu banyak yang tidak memiliki label. Label data akan masuk ke ERP apapun bentuk ERPnya, sedangkan jika datanya berupa natural input seperti suara, gambar, dan video tidak bisa. Pada unsupervised learning tidak menggunakan label pada saat memprediksi target feauture / variable nya. Namun menggunakan kesamaan

dari attribut yang dimiliki. Jika attribut dan sifat dari data feature yang diekstrak memiliki kemiripan, maka akan dikelompokan (clustering). Sehingga nantinya akan menimbulkan kelompok kelompok (cluster). Jumlah cluster bisa tak terbatas. Dari kelompok kelompok itu model akan melabelkan, dan jika data baru yang mau di prediksi, maka akan dicocokkan dengan kelompok yang mirip featurenya.

1.18.2.5 Definisi Data Set Dataset adalah kumpulan dari data. Yang paling umum satu data set sesuai dengan isi tabel pada database tunggal, atau matriks data pada statistik tunggal, di mana setiap kolom tabel mewakili suatu variabel tertentu, dan setiap baris sesuai dengan anggota tertentu dari dataset yang dipertanyakan.

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

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

1.18.3 Instalasi

- Instalasi Library scikit dari anaconda, mencoba kompilasi dan uji coba ambil contoh kode dan lihat variabel explorer
- Mencoba Loading an example dataset, menjelaskan maksud dari tulisan tersebut dan mengartikan per baris

```

1 #%% Mencoba loading an example dataset
2 from sklearn import datasets
3 # Memanggil class datasets dari library sklearn

```

```

  % Total    Downloaded   Speed     Estimated Left Time
  0%          0B       0B/s      00:00:00
  1%          0B       0B/s      00:00:00
  2%          0B       0B/s      00:00:00
  3%          0B       0B/s      00:00:00
  4%          0B       0B/s      00:00:00
  5%          0B       0B/s      00:00:00
  6%          0B       0B/s      00:00:00
  7%          0B       0B/s      00:00:00
  8%          0B       0B/s      00:00:00
  9%          0B       0B/s      00:00:00
 10%          0B       0B/s      00:00:00
 11%          0B       0B/s      00:00:00
 12%          0B       0B/s      00:00:00
 13%          0B       0B/s      00:00:00
 14%          0B       0B/s      00:00:00
 15%          0B       0B/s      00:00:00
 16%          0B       0B/s      00:00:00
 17%          0B       0B/s      00:00:00
 18%          0B       0B/s      00:00:00
 19%          0B       0B/s      00:00:00
 20%          0B       0B/s      00:00:00
 21%          0B       0B/s      00:00:00
 22%          0B       0B/s      00:00:00
 23%          0B       0B/s      00:00:00
 24%          0B       0B/s      00:00:00
 25%          0B       0B/s      00:00:00
 26%          0B       0B/s      00:00:00
 27%          0B       0B/s      00:00:00
 28%          0B       0B/s      00:00:00
 29%          0B       0B/s      00:00:00
 30%          0B       0B/s      00:00:00
 31%          0B       0B/s      00:00:00
 32%          0B       0B/s      00:00:00
 33%          0B       0B/s      00:00:00
 34%          0B       0B/s      00:00:00
 35%          0B       0B/s      00:00:00
 36%          0B       0B/s      00:00:00
 37%          0B       0B/s      00:00:00
 38%          0B       0B/s      00:00:00
 39%          0B       0B/s      00:00:00
 40%          0B       0B/s      00:00:00
 41%          0B       0B/s      00:00:00
 42%          0B       0B/s      00:00:00
 43%          0B       0B/s      00:00:00
 44%          0B       0B/s      00:00:00
 45%          0B       0B/s      00:00:00
 46%          0B       0B/s      00:00:00
 47%          0B       0B/s      00:00:00
 48%          0B       0B/s      00:00:00
 49%          0B       0B/s      00:00:00
 50%          0B       0B/s      00:00:00
 51%          0B       0B/s      00:00:00
 52%          0B       0B/s      00:00:00
 53%          0B       0B/s      00:00:00
 54%          0B       0B/s      00:00:00
 55%          0B       0B/s      00:00:00
 56%          0B       0B/s      00:00:00
 57%          0B       0B/s      00:00:00
 58%          0B       0B/s      00:00:00
 59%          0B       0B/s      00:00:00
 60%          0B       0B/s      00:00:00
 61%          0B       0B/s      00:00:00
 62%          0B       0B/s      00:00:00
 63%          0B       0B/s      00:00:00
 64%          0B       0B/s      00:00:00
 65%          0B       0B/s      00:00:00
 66%          0B       0B/s      00:00:00
 67%          0B       0B/s      00:00:00
 68%          0B       0B/s      00:00:00
 69%          0B       0B/s      00:00:00
 70%          0B       0B/s      00:00:00
 71%          0B       0B/s      00:00:00
 72%          0B       0B/s      00:00:00
 73%          0B       0B/s      00:00:00
 74%          0B       0B/s      00:00:00
 75%          0B       0B/s      00:00:00
 76%          0B       0B/s      00:00:00
 77%          0B       0B/s      00:00:00
 78%          0B       0B/s      00:00:00
 79%          0B       0B/s      00:00:00
 80%          0B       0B/s      00:00:00
 81%          0B       0B/s      00:00:00
 82%          0B       0B/s      00:00:00
 83%          0B       0B/s      00:00:00
 84%          0B       0B/s      00:00:00
 85%          0B       0B/s      00:00:00
 86%          0B       0B/s      00:00:00
 87%          0B       0B/s      00:00:00
 88%          0B       0B/s      00:00:00
 89%          0B       0B/s      00:00:00
 90%          0B       0B/s      00:00:00
 91%          0B       0B/s      00:00:00
 92%          0B       0B/s      00:00:00
 93%          0B       0B/s      00:00:00
 94%          0B       0B/s      00:00:00
 95%          0B       0B/s      00:00:00
 96%          0B       0B/s      00:00:00
 97%          0B       0B/s      00:00:00
 98%          0B       0B/s      00:00:00
 99%          0B       0B/s      00:00:00
100%          0B       0B/s      00:00:00

```

Gambar 1.93 Instalasi Package Scikit



Gambar 1.94 Isi Variabel Explorer Package Scikit

```

4 iris = datasets.load_iris()
5 # Menggunakan contoh datasets iris
6 x = iris.data
7 # Menyimpan data sets iris pada variabel x
8 y = iris.target
9 # Menyimpan data label iris pada variabel y

```

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

```

1 %%Mencoba Learning dan predicting
2 from sklearn.neighbors import KNeighborsClassifier
3 #Memanggil fungsi KNeighborsClassifier dari library sklearn
4 import numpy as np
5 #Memamnggil library numpy dengan alias np
6 knn=KNeighborsClassifier(n_neighbors=1)
7 #Membuat variabel kkn dan memanggil fungsi
8 #KNeighborsClassifier lalu mendefinisikan k adalah 1
9 knn.fit(x,y)
10 #Perhitungan matematika kkn
11 a=np.array([1.0 ,2.0 ,3.0 ,4.0])
12 #Membuat Array
13 a = a.reshape(1,-1)
14 #Mengubah Bentuk Array jadi 1 dimensi
15 hasil = knn.predict(a)
16 #Memanggil fungsi prediksi
17 print(hasil)
18 #menampilkan hasil prediksi

```

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

```

1 %% Model Persistense
2 from sklearn import svm
3 #Memanggil class svm dari library sklearn

```

```

4 from sklearn import datasets
5 #Memanggil class datasets dari library sklearn
6 clf = svm.SVC()
7 #Membuat variabel dengan nama clf , dan memanggil class svm
    dan fungsi SVC
8 X, y = datasets.load_iris(return_X_y=True)
9 #Mengambil dataset iris dan mengembalikan nilainya .
10 clf.fit(X, y)
11 #Perhitungan nilai label
12
13 from joblib import dump, load
14 #memanggil class dump dan load pada library joblib
15 dump(clf, '1174071.joblib')
16 #Menyimpan model kedalam 1174071.joblib
17 hasil = load('1174071.joblib')
18 #Memanggil model 1174071 dan disimpan pada variable hasil
19 print(hasil)
20 #Menampilkan variable hasil

```

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

```

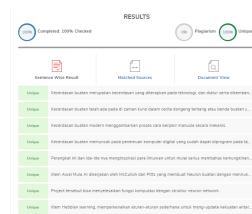
1 %% Conventions
2 import numpy as np
3 #memanggil library numpy dengan alias np
4 from sklearn import random_projection
5 #Memanggil class random_projection dari library sklearn
6
7 rng = np.random.RandomState(0)
8 #Membuat variabel rng, dan mendefinisikan np, memanggil fungsi
    random dan attr RandomState kedalam variabel
9 X = rng.rand(10, 2000)
10 #Membuat variabel X, dan menentukan nilai random dari 10 –
    2000
11 X = np.array(X, dtype='float32')
12 #Menyimpan hasil nilai random sebelumnya , kedalam array , dan
    menentukan typedatanya yaitu float32
13 X.dtype
14 #Mengubah data tipe menjadi float64
15
16 transformer = random_projection.GaussianRandomProjection()
17 #Membuat variabel transformer, dan mendefinisikan
    classrandom_projection dan memanggil fungsi
    GaussianRandomProjection
18 X_new = transformer.fit_transform(X)
19 #Membuat variabel X_new dan melakukan perhitungan label pada
    variabel X
20 X_new.dtype
21 #Mengubah data tipe menjadi float64
22 print(X_new)
23 #Menampilkan isi variabel X_new

```

```

File "c:\users\muham\anaconda3\lib\site-packages\spyder_kernels\customize
\syspathcontext.py", line 110, in _execfile
    exec(compile(f.read(), filename, "exec"), namespace)
File "c:\users\muham\documents\sklearn.py", line 8, in <module>
    from sklearn import datasets
ModuleNotFoundError: No module named 'sklearn'

In [2]:
```

Gambar 1.95 Import Library Error**Gambar 1.96** Install Library**Gambar 1.97** Bukti Tidak Melakukan Plagiarisme Chapter 1

1.18.4 Penanganan Error

1. Screenshot Error
2. Tuliskan Kode Error dan Jenis Error
 - Import Library Error
3. Cara Penangan Error
 - Impor Library Error
Menginstall Library yang belum ada

1.18.5 Bukti Tidak Plagiarisme

1.18.6 Link Youtube

<http://bit.ly/AIkem71>

1.19 Kaka Kamaludin

1.19.1 Teori

1. Definisi, Sejarah dan Perkembangan Kecerdasan Buatan

- Definisi Kecerdasan Buatan

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

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

- Sejarah Kecerdasan Buatan

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

Artificial intelligence merupakan inovasi baru di bidang ilmu pengetahuan. Mulai terbentuk sejak adanya komputer modern dan kira-kira terjadi sekitaran tahun 1940 dan 1950. Ilmu pengetahuan komputer ini khusus ditujukan dalam perancangan otomatisasi tingkah laku cerdas dalam sistem kecerdasan komputer. Pada awal 50-an, studi tentang “mesin berpikir” memiliki berbagai nama seperti cybernetics, teori automata, dan pemrosesan informasi. Pada tahun 1956, para ilmuan jenius seperti Alan Turing, Norbert, Wiener, Claude Shannon dan Warren McCullough telah bekerja secara independen di bidang cybernetics, matematika, algoritma dan teori jaringan. Na-

mun, seprang ilmuan komputer dan kognitif John McCarthy adalah orang yang dating dengan ide untuk bergabung dengan upaya penelitian terpisah ini kedalam satu bidang yang akan mempelajari topic baru untuk imajinasi manusia yaitu kecerdasan buatan. Dia adalah orang yang menciptakan istilah tersebut dan kemudian mendirikan laboratorium Kecerdasan Buatan di MIT dan Stan ford.

- Perkembangan Kecerdasan Buatan

Teknologi Artificial Intelligence semakin ramai dibahas dalam berbagai diskusi teknologi di seluruh dunia. Menurut kebanyakan orang, pekerjaan seperti kasir, operator telepon, pengendara truk, dan lainnya sangat berpeluang besar untuk tergantikan oleh Artificial Intelligence. Mengapa terjadi hal demikian? dikarenakan memang bahwa AI lebih unggul dalam hal kinerja, fitur dan lain sebagainya. Namun, dalam beberapa aspek memang pekerja manusia masih unggul dibandingkan AI itu sendiri. Para generasi muda yang ada di dunia terutama di daerah Asia terlihat sudah memahami fungsi dan efek dari AI dalam kehidupan kita sehari-hari. Berdasarkan survei yang dilakukan oleh Microsoft, terdapat 39 persen responden yang mempertimbangkan untuk menggunakan mobil tanpa pengemudi dan 36 persen lainnya setuju bahwa robot masa depan dengan software untuk beroperasi mampu meningkatkan produktivitas. Dari survey tersebut kita sebagai pengguna AI harus lebih bijaksana dalam pengembangan dan penggunaan dari AI sehingga tanpa memberikan efek samping terhadap etos kerja dan keseharian kita sebagai pengguna dalam kehidupan sehari-hari.

AI Summer 1 (1956-1973) KOnferensi Dartmounth diikuti oleh 17 tahun kemajuan luar biasa. Proyek penelitian yang dilakukan di MIT, universitas di Edinburgh, Stanford dan Carnegie Mellon menerima dana besar-besaran, yang akhirnya membuaikan hasil. Selama tahun-tahun itulah komputer pemrograman mulai melakukan masalah aljabar, membuktikan teorema geometris, memahami dan menggunakan sintaks dan tata bahasa Inggris. Terlepas dari ditinggalkannya koneksiisme dan terjemahan mesin yang gagal, yang menunda penelitian Natural Language Processing (NLP) selama bertahun-tahun, banyak prestasi dari masa lalu yang membuat sejarah. Berikut ini beberapa diantaranya : Pelopor pembelajaran mesin, Ray Solomonoff meletakkan dasar-dasar teori metematika AI, memperkenalkan metode Bayesian universal untuk inferensi dan prediksi induktif Thomas Evans menciptakan program ANALOGI heuristik, yang memungkinkan komputer memecahkan masalah geometrianalogi Unimation, perusahaan robotika pertama didunia, menciptakan robot industri Unimate, yang bekerja pada jalur perakitan modil Genenal Motors. Joseph

Weizenbaum membangun ELIZA-program interaktif yang dapat membawa percakapan dalam bahasan Inggris tentang topik apapun. Ross Quillian menunjukkan jaring semantik, sedangkan Jaime Carbonell (Sr.) mengembangkan Cendikia-program interaktif untuk instruksi yang dibantu komputer berdasarkan jaring semantik. Edward Feigenbaum dan Julian Feldman menerbitkan *Computeks and Thought*, kumpulan artikel pertama tentang AI.

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

▪ Definisi Supervised Learning

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

▪ Klasifikasi

Klasifikasi adalah pembagian sesuatu menurut kelas-kelas (class). Menurut Ilmu Pengetahuan, Klasifikasi merupakan proses pengelompokan benda berdasarkan ciri-ciri persamaan dan juga perbedaan. Dalam masalah klasifikasi, kami mencoba memprediksi sejumlah nilai terpisah. Label (y) umumnya datang dalam bentuk kategorikal dan mewakili sejumlah kelas. Dalam pembelajaran mesin dan statistik, klasifikasi adalah pendekatan pembelajaran yang diawasi di mana program komputer belajar dari input data yang diberikan kepadanya dan

kemudian menggunakan pembelajaran ini untuk mengklasifikasikan pengamatan baru. Kumpulan data ini mungkin hanya bersifat dua kelas (seperti mengidentifikasi apakah orang tersebut berjenis kelamin laki-laki atau perempuan atau bahwa surat itu spam atau bukan-spam) atau mungkin juga multi-kelas. Beberapa contoh masalah klasifikasi adalah: pengenalan ucapan, pengenalan tulisan tangan, identifikasi metrik, klasifikasi dokumen dll.

- Regresi

Regresi adalah metode analisis statistik yang digunakan untuk melihat pengaruh antara dua ataupun lebih variabel. Regresi adalah membahas masalah ketika variabel output adalah nilai riil atau berkelanjutan, seperti "gaji" atau "berat". Banyak model yang berbeda dapat digunakan makan, yang paling sederhana adalah regresi linier. Ia mencoba untuk menyesuaikan data dengan hyper-plane terbaik yang melewati poin.

- Unsupervised Learning

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

- Data set

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

- Training Set

Training Set adalah set digunakan oleh algoritma klasifikasi . Dapat dicontohkan dengan : decision tree, bayesian, neural network

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

- Testing set

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

1.19.2 Praktek

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

- Buka prompt anaconda lalu ketikkan conda scikit-learn
- Biarkan proses berjalan hingga selesai

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

Gambar 1.98 Install library scikit

```
(Base) C:\Users\root>conda install scikit-learn
Collecting package metadata (current_repodata.json): done
Solving environment: done
## Package Plan ##
environment location: C:\Users\root\Anaconda3
added / updated specs:
  scikit-learn

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

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

Proceed ([y]/n)? y

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

Gambar 1.99 Proses jika berhasil

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

- Ketikkan script berikut di spyder, lalu run

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

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

Gambar 1.100 hasil test.py

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

```

1 #Import fungsi datasets dari library sklearn
2 from sklearn import datasets
3
4
5 #Memasukkan data dari datasets iris ke variable iris
6 iris = datasets.load_iris()
7
8 #Memasukkan data dari datasets digits ke variable digits
9 digits = datasets.load_digits()
10
11 #Mengimport sebuah Support Vector Machine(SVM) yang merupakan
12 #algoritma classification yang akan diambil dari Scikit-
13 #Learn.
14 from sklearn import svm
15
16
17 #Mendeklarasikan suatu value yang bernama clf yang berisi
18 #gamma.
19 clf = svm.SVC(gamma=0.001, C=100.)
20
21 #Estimator clf (for classifier)
22 clf.fit(digits.data[:-1], digits.target[:-1])
23
24 #Menunjukkan prediksi angka baru
25 hasil = clf.predict(digits.data[-1:])
26
27 #Menampilkan
28 print(hasil)

```

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

Gambar 1.101 variabel explorer test2.py

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

```

1 %%Cara Dump Pertama
2
3
4
5 #Mengimport sebuah Support Vector Machine(SVM) yang merupakan
6 #algoritma classification yang akan diambil dari Scikit-
7 #Learn.
8 from sklearn import svm
9 #Import fungsi datasets dari library sklearn
10 from sklearn import datasets
11
12 #Mendefinisikan clf dengan fungsi svc dari library svm
13 clf = svm.SVC()

```

```
12
13 #Mengisi variable x dan y dengan data dari datasets
14 X, y = datasets.load_iris(return_X_y=True)
15
16 #Estimator clf (for classifier)
17 clf.fit(X, y)
18
19 #Mengimport Library pickle
20 import pickle
21
22 #Menyimpan hasil dari clf kedalam sebuah dump
23 s = pickle.dumps(clf)
24
25 #Memanggil dump yang dihasilkan pickle lalu memasukkan hasil
#dumpnya ke variable
26 clf2 = pickle.loads(s)
27
28
29 #Memprediksi angka yang akan muncul
30 clf2.predict(X[0:1])
31
32 #Menampilkan data prediksi
33 print(y[0])
34
35 #%%Cara Dump Kedua
36 # Digunakan untuk memanggil class svm dari library sklearn
37 from sklearn import svm
38
39 # Diguangkan untuk class datasets dari library sklearn
40 from sklearn import datasets
41
42 # membuat variabel clf , dan memanggil class svm dan fungsi
# SVC
43 clf = svm.SVC()
44
45 #Mengambil dataset iris dan mengembalikan nilainya .
46 X, y = datasets.load_iris(return_X_y=True)
47
48 #Perhitungan nilai label
49 clf.fit(X, y)
50
51
52 #memanggil class dump dan load pada library joblib
53 from joblib import dump, load
54
55
56 #Menyimpan model kedalam 1174067.joblib
57 dump(clf, '1174067.joblib')
58
59 #Memanggil model 1174067
60 hasil = load('1174067.joblib')
61 hasil.predict(X[0:1])
62
63 # Menampilkan Model yang dipanggil sebelumnya
```

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

Gambar 1.102 variabel explorer test3.py

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

```

1 # memanggil library numpy dan dibuat alias np
2 import numpy as np
3
4 #Memanggil class random_projection pada library sklearn
5 from sklearn import random_projection
6
7 #Membuat variabel rng, dan mendefisikan np, fungsi random dan
8 #attr RandomState kedalam variabel
9 rng = np.random.RandomState(0)
10
11 # membuat variabel X, dan menentukan nilai random dari 10 –
12 # 2000
13 X = rng.rand(10, 2000)
14
15 #menyimpan hasil nilai random sebelumnya, kedalam array, dan
16 #menentukan typedatanya sebagai float32
17 X = np.array(X, dtype='float32')
18
19 # Mengubah data tipe menjadi float64
20 X.dtype
21
22 #membuat variabel transformer, dan mendefinisikan
23 #classrandom_projection dan memanggil fungsi
24 # GaussianRandomProjection
25 transformer = random_projection.GaussianRandomProjection()
26 # membuat variabel baru dan melakukan perhitungan label pada
27 # variabel X
28 X_new = transformer.fit_transform(X)
29 # Mengubah data tipe menjadi float64
30 X_new.dtype
31 # Menampilkan isi variabel X_new
32 print(X_new)

```

Name	Type	Size	Value
X	float32	(10, 2000)	[[0.5488135 0.71518934 0.60276335 ... 0.4801078 0.64386404 0.5017731 ...
X_new	float64	(10, 1973)	[[-0.74506698 0.45457383 0.88209808 ... -0.00255147 -0.19283521 -0 ...
digits	utils.Bunch	5	Bunch object of sklearn.utils module
iris	utils.Bunch	6	Bunch object of sklearn.utils module
y	int32	(150,)	[0 0 0 ... 2 2 2]

Gambar 1.103 variabel explorer test4.py

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

In [6]: |
```

Gambar 1.104 hasil test4.py

1.19.3 Penanganan Error

1. Screenshoots Error



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

Gambar 1.105 error

2. Jenis Error

- Not Module name "joblib"

3. Cara Penanganan Error

- install library joblib

1.19.4 Bukti tidak plagiat

1.19.5 Link Youtube

<https://youtu.be/FDQopV2LUIo>

1.20 Dini Permata Putri - 1174053

1.20.1 Teori

1.20.2 Pengertian Kecerdasan Buatan

Kecerdasan buatan adalah salah satu cabang ilmu pengetahuan yang berhubungan dengan mesin untuk memecahkan persoalan yang rumit dengan cara yang lebih manusiawi. Hal ini biasanya dilakukan dengan mengikuti karakteristik dan analogi berpikir dari kecerdasan atau inteligance manusia, dan menerapkan sebagai algoritma yang dikenal oleh komputer. Dengan suatu pendekatan yang kurang lebih fleksibilitas dan efisien dapat diambil tergantung keperluan yang mempengaruhi bagaimana wujud dari prilaku kecerdasan buatan. AI biasanya dihubungkan dengan ilmu komputer, akan tetapi juga terkait erat dengan bidang lain seperti matematika, Psikologi, Pengamatan, Biolog, filosofi, dan lainnya

1.20.3 Sejarah Kecerdasan Buatan

Kecerdasan buatan merupakan bidang ilmu komputer yang sangat penting di era kini dan masa yang akan datang untuk mewujudkan sistem komputer yang cerdas. Bidang ini telah berkembang sangat pesat di 20 tahun terakhir seiring dengan kebutuhan perangkat cerdas pada industry dan rumah tangga. Kata Intelligence berasal dari bahasa latin "intelligo" yang berarti "saya paham". Berarti dasar dari intelligence adalah kemampuan untuk memahami dan melakukan aksi. nyatanya, bidang Kecerdasan Buatan atau disingkat dengan AI, berawal dari kemunculan komputer sekitar tahun 1940-an, sedangkan perkembangan sejarah dapat ditelusuri sejak zaman Mesir kuno. Pada saat ini, perhatian mendesak diberikan pada kemampuan komputer untuk melakukan hal-hal yang dapat dilakukan manusia. Dalam hal ini, komputer ini dapat meningkatkan kemampuan kecerdasan dan kecerdasan manusia. Pada awal abad ke-17, René berbicara tentang tubuh binatang yang tidak meminta apa pun selain mesin yang rumit. Blaise Pascal membuat mesin hitung digital mekanis pertama pada tahun 1642. Pada 19, Charles Babbage dan Ada Lovelace bekerja pada mesin hitung mekanis yang dapat diprogram. Bertrand Russell dan Alfred Whitehead North menerbitkan Principia Mathematica, yang merombak logistik formal. Warren McCulloch dan Walter Pitts menerbitkan "Kalkulus Logika Gagasan yang Menjaga Aktivitas" pada tahun 1943 yang membentuk dasar bagi jaringan saraf. 1950-an adalah periode

upaya aktif dalam AI. program permainan catur yang ditulis oleh Dietrich Prinz. John McCarthy menciptakan istilah "kecerdasan buatan" pada konferensi pertama yang menjadi dasar perjanjian itu, pada tahun 1956. Dia juga menemukan bahasa pemrograman Lisp. Alan Turing memperkenalkan "tes Turing" sebagai cara untuk mengoperasionalkan tes kecerdasan cerdas.

1.20.4 Perkembangan dan Penggunaan Kecerdasan

Menurut studi Harvard Business Review dan ICM Unlimited pada tahun 2016, perusahaan besar memberikan kompensasi 10 persen lebih tinggi untuk setiap karyawan, Terrelong melanjutkan pengembangan Artificial Intelligence (AI) tidak hanya untuk membuat gambar atau video palsu lebih mudah, tetapi juga membuatnya sulit untuk membuktikannya. Meskipun pada saat ini, upaya untuk membuat dan mendistribusikan konten hoax, alias hoaks, masih dapat diatasi, tetapi berhasil, tantangan yang dihadapi semakin sulit. Selain itu, AI memungkinkan pembuatan gambar, video, atau audio palsu dari bahan yang relatif minim. Moody's, yang harus disetujui, membuktikan upaya itu akan semakin menantang dan membutuhkan teknik forensik yang lebih canggih. Pada Mei 2019, para peneliti di Samsung AI Center dan Institut Sains dan Teknologi Skolkovo di Moskow, Rusia menunjukkan bahwa mereka dapat membuat tayangan video yang menampilkan masing-masing individu. Video ini sangat realistik tetapi sebenarnya palsu, dibuat menggunakan model pembelajaran tertentu yang disebut Generative Adversarial Network (GAN). Hasil dari proses GAN disebut deepfakes karena mereka menggunakan teknik pembelajaran yang mendalam untuk membuat konten palsu. Untuk jangka pendek, perusahaan diharapkan untuk terus memainkan media sosial dan situs untuk melihat pentingnya disinformasi dan meminta mereka yang bertanggung jawab untuk media sosial dan situs terkait untuk mengunduh konten. Terrelonge menambahkan langkah lain yang bisa diam-bil untuk merilis materi resmi untuk melawan konten palsu."Perlindungan terhadap konten palsu membutuhkan kombinasi teknologi dan pendidikan,".

1.21 resume mengenai definisi supervised learning, klarifikasi, regresi, dan un-supervised learning. Data Set, training set dan testing set

1.21.1 Supervised Learning

Supervised Learning adalah tugas mengumpulkan data untuk melengkapi fungsi data pelatihan yang diberi label. Data pelatihan terdiri dari contoh pelatihan. Dalam pembelajaran terawasi, setiap contoh adalah pasangan yang terdiri dari objek input (biasanya vektor) dan nilai output dingin (juga disebut sinyal pengawasan super). Algoritma pembelajaran yang diawasi menganalisis data pelatihan dan menghasilkan fungsi yang lengkap, yang dapat digu-

nakan untuk memetakan contoh-contoh baru. Skenario akan memungkinkan algoritma menentukan lable kelas dengan benar untuk instance yang tidak terlihat. Ini membutuhkan algoritma pembelajaran untuk menggeneralisasi data pelatihan sehingga tidak muncul dengan cara yang "masuk akal". Pembelajaran terawasi semakin dekat di mana ada pelatihan praktis selain dapat bervariasi yang berarti tujuannya adalah di mana mengelompokkan data ke dalam database yang ada. Pembelajaran terawasi menyediakan jumlah pembelajaran yang direkomendasikan untuk mendukung penilaian di masa depan. Obrolan, program mengemudi mandiri, pengenalan wajah, tatap muka dan robot adalah beberapa sistem yang dapat menggunakan pembelajaran yang diawasi atau tidak diawasi. Pembelajaran terbimbing sebagian besar terkait dengan AI berdasarkan pengambilan mereka juga mungkin diperlukan menggunakan model pembelajaran generatif. Pelatihan data untuk pembelajaran dimulai dengan mendiskusikan contoh-contoh dengan subjek input berpasangan dan output yang diinginkan (juga disebut sebagai sinyal pengawasan). Dalam pembelajaran yang diawasi untuk pemrosesan gambar, misalnya sistem AI dapat lengkap dengan gambar mengemudi yang berlabel dalam kategori mobil dan truk. Setelah jumlah yang memadai, sistem harus dapat membedakan antara dan mengklasifikasikan gambar yang tidak berlabel, di mana waktu pelatihan dapat diselesaikan secara penuh. Model Pembelajaran Terpandu memiliki beberapa keunggulan dibandingkan pengawasan, tetapi mereka juga memiliki keterbatasan. Sistem lebih cenderung membuat penilaian bahwa hak asasi manusia dapat dihubungkan, misalnya karena manusia telah memberikan dasar untuk pengambilan keputusan. Namun, dalam hal metode berbasis pengambilan, Supervised Learning menghilangkan kesulitan dalam menangani informasi baru. Jika sistem dikategorikan untuk mobil dan truk, maka sepeda disediakan, misalnya, harus dikelompokkan dalam satu kategori atau yang lain. Namun. Jika sistem AI generatif, mungkin tidak tahu apa itu sepeda tetapi akan dapat mengenalinya sebagai milik kategori yang terpisah.

1.21.2 Klasifikasi

Klasifikasi adalah pembagian hal sesuai dengan kelas (kelas). Menurut Science, klasifikasi adalah proses pengelompokan materi berdasarkan karakteristik dan perbedaan yang sama. Dalam masalah klasifikasi, kami mencoba memprediksi sejumlah nilai yang terpisah. Label (y) Umumnya datang dalam bentuk kategorikal dan mewakili sejumlah kelas. Dalam pembelajaran statistik dan pembelajaran mesin statistik, klasifikasi adalah pembelajaran yang dimulai ketika sebuah program komputer belajar dari input data yang disediakan untuk mendukung dan kemudian menggunakan pembelajaran ini untuk mengklasifikasikan pembelajaran baru. Pengumpulan data ini mungkin hanya dua kelas (seperti mengidentifikasi apakah orang ini laki-laki atau perempuan atau orang itu adalah spam atau bukan-spam) atau mungkin juga multi-kelas.

Beberapa contoh masalah klasifikasi adalah: pengenalan ucapan, pengenalan tulisan tangan, metrik identifikasi, klasifikasi dokumen dll.

1.21.3 Regresi

Regresi adalah metode analisis statistik yang digunakan untuk melihat perbedaan antara dua atau lebih variabel. Regresi sedang membahas masalah kompilasi, variabel output adalah nilai nyata atau dipertahankan, seperti "gaji" atau "berat". Banyak model yang berbeda dapat digunakan untuk makan, cara paling sederhana adalah linearitas linear. Itu mencoba untuk mencocokkan data dengan pesawat-hyper terbaik yang melewati titik.

1.21.4 unsupervised learning

Belajar tanpa pengawasan berbeda dari Belajar dengan Supervisi. Perbedaannya adalah bahwa pembelajaran tanpa pengawasan tidak memiliki data pelatihan, jadi dari data yang tersedia kami mengelompokkan data menjadi 2 atau 3 bagian dan seterusnya. Unsupervised Learning adalah pelatihan dalam algoritma kecerdasan buatan (AI) menggunakan informasi yang tidak diklasifikasikan atau diberi label dan menyediakan algoritma untuk memperbaiki informasi yang diberikan tanpa bimbingan. Dalam Unattended Learning, sistem AI dapat mengklasifikasikan informasi yang tidak diurutkan berdasarkan ekuitas dan perbedaan dalam kategori mendadak yang disediakan. Dalam Supervised Learning Learning, sistem AI disajikan dengan sistem wajib yang tidak diberi label, tidak dikategorikan dan algoritma bekerja pada data tanpa pelatihan sebelumnya. Outputnya tergantung pada algoritma kode. Menyerahkan sistem untuk Belajar Tanpa Pengawasan adalah salah satu cara untuk menerima AI. Algoritma Pembelajaran tanpa pengawasan dapat melakukan tugas yang lebih kompleks daripada sistem pembelajaran yang diawasi. Namun, pembelajaran tanpa pengawasan dapat lebih tidak konsisten dengan model alternatif. Sementara Supervised Learning Mungkin, misalnya, mencari sendiri dengan memilih kucing dari anjing, ia juga dapat menambahkan kategori yang tidak diinginkan dari yang tidak diinginkan untuk ditingkatkan menjadi ras yang tidak biasa, membuat pesanan diperlukan.

1.21.5 Data Set

Dataset adalah objek yang mewakili data dan hubungan dalam memori. Strukturnya mirip dengan basis data basis data, tetapi hanya kumpulan data yang dikumpulkan dari catatan dan latar belakang yang diaktifkan. dapatkan persetujuan yang tepat untuk mengumpulkan atau mengidentifikasi data yang berkorelasi dengan hasil yang ingin Anda hasilkan; yaitu data yang berisi sinyal tentang acara yang Anda sukai. Data harus disinkronkan dengan masalah yang Anda coba selesaikan. Gambar kucing bukan kompilasi yang sangat berguna. Anda sedang membangun sistem identifikasi wajah.

Memodifikasi data yang selaras dengan masalah yang ingin Anda selesaikan harus dilakukan oleh para ahli data. Jika Anda tidak memiliki data yang benar, maka upaya Anda untuk membuat solusi AI harus kembali ke instalasi data. Format ujung kanan untuk belajar secara umum adalah array tensor, atau multi-dimensional. Jadi pipa data yang dibangun untuk pembelajaran dibangun secara umum untuk mengubah semua gambar, video, suara, suara, teks atau deret waktu menjadi vektor dan tensor yang dapat digunakan operasi aljabar linier. Data yang diperlukan perlu dinormalisasi, distandarisasi dan dikembalikan untuk meningkatkan kegunaannya, dan semua ini adalah langkah-langkah dalam pembelajaran mesin ETC. Deeplearning4j menawarkan alat ETV Data Vec untuk melakukan tugas memfasilitasi data. Pembelajaran yang mendalam, dan pembelajaran mesin yang lebih umum, membutuhkan pelatihan yang baik agar dapat bekerja dengan baik. Mengumpulkan dan membangun satu set badan pelatihan yang cukup besar dari data yang diketahui membutuhkan waktu dan pengetahuan khusus tentang pengetahuan dan cara untuk mengumpulkan informasi yang relevan. Perangkat pelatihan bertindak sebagai patokan terhadap mana jaring pembelajaran dalam pengeboran. Itulah yang mereka perbarui untuk direkonstruksi sebelum mereka merilis data yang belum pernah dilihat sebelumnya. Pada saat ini, manusia memiliki pengetahuan luas tentang mengidentifikasi instrumen yang tepat dan mengubahnya menjadi representasi numerik yang dapat dipahami oleh algoritma pembelajaran dalam, tensor. Membangun set pelatihan, dalam arti tertentu, pra-pelatihan. Kumpulan pelatihan yang membutuhkan banyak waktu atau keahlian yang dapat membantu dalam dunia data dan pemecahan masalah. Sifat keahlian terbesar Anda dalam memberi tahu algoritma Anda apa yang penting bagi Anda adalah memilih apa yang Anda masukkan dalam kursus pelatihan Anda. Ini melibatkan menceritakan kisah melalui data awal yang Anda pilih untuk memandu proses pembelajaran mendalam Anda dengan mengekstraksi fitur-fitur penting, baik dalam pengaturan pelatihan dan data yang ingin Anda buat untuk dipelajari. Agar pelatihan ini bermanfaat, Anda harus memecahkan masalah yang Anda selesaikan; yaitu, apa yang Anda inginkan agar sesuai dengan pembelajaran Anda, di mana hasil yang ingin Anda prediksi.

1.21.6 Training Set

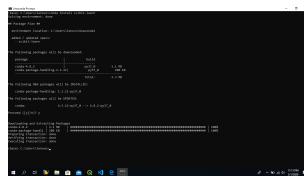
Set Pelatihan adalah set yang digunakan oleh algoritma klasifikasi. Dapat dicontohkan oleh: decisiontree, bayesian, neural network dll. Semuanya dapat digunakan untuk membuat model kelas. Terkait dengan pelatihan yang mengatur melalui jaringan saraf di internet bagaimana menimbang berbagai fitur, sesuaikan koefisien sesuai dengan apa yang mereka tingkatkan dalam hasil Anda. Koefisien, juga dikenal sebagai parameter, akan terkandung dalam sensor dan bersama-sama mereka disebut model, model data karena mereka menyandikan latihan yang mereka praktekkan.

1.21.7 testing Set

Tes ini digunakan untuk mengukur sejauh mana classifil berhasil mengklasifikasi dengan benar. Ini digunakan sebagai meterai persetujuan, dan Anda tidak dapat digunakan sampai akhir. Setelah Anda melatih dan mengoptimalkan data Anda, Anda menguji jaringan saraf Anda untuk mengambil sampel acak akhir ini. Hasilnya harus memvalidasi gambar bersih Anda, atau gambar mengenali [x] dari nomor itu. Jika Anda tidak mendapatkan prediksi yang akurat, kembalilah ke set pelatihan Anda, lihat mitra Anda yang Anda gunakan untuk mengelola jaringan Anda, dan kualitas data Anda dan lihat teknik pra-pemanfaatan yang dapat Anda gunakan.

1.21.8 Instalasi

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



Gambar 1.106 Instalasi Package Scikit Learn

Gambar 1.107 Isi Variabel Explorer

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

```
1 from sklearn import datasets # Digunakan Untuk Memanggil  
2     class datasets dari library sklearn  
3 iris = datasets.load_iris() # Menggunakan contoh datasets  
4     iris  
5 x = iris.data             # Menyimpan nilai data sets iris  
6     pada variabel x  
7 y = iris.target            # Menyimpan nilai data label iris  
8     pada variabel y
```

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

```

1 #%%Mencoba Learning dan predicting
2 from sklearn.neighbors import KNeighborsClassifier #Digunakan
   Untuk Memanggil fungsi KNeighborsClassifier
3                                     # pada
4     class sklearn dan library sklearn
5 import numpy as np # memanggil library numpy dan dibuat alias
   np
6 knn=KNeighborsClassifier(n_neighbors=1) #membuat variabel kkn
   , dan memanggil fungsi KNeighborsClassifier
7                                     #dan mendefinisikan k
   -nya adalah 1
8 knn.fit(x,y)                      #Perhitungan
   matematika library kkn
9 a=np.array([1.0,2.0,3.0,4.0])
10 a = a.reshape(1,-1)
    Array jadi 1 dimensi
11 hasil = knn.predict(a)            #Memanggil fungsi
   prediksi

```

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

```

1 #%% Model Persistense
2 from sklearn import svm # Digunakan untuk memanggil class svm
   dari library sklearn
3 from sklearn import datasets # Digunakan untuk class datasets
   dari library sklearn
4 clf = svm.SVC()           # membuat variabel clf , dan
   memanggil class svm dan fungsi SVC
5 X, y = datasets.load_iris(return_X_y=True) #Mengambil dataset
   iris dan mengembalikan nilainya.
6 clf.fit(X, y)             #Perhitungan nilai label
7
8 from joblib import dump, load #memanggil class dump dan load
   pada library joblib
9 dump(clf, '1174053.joblib') #Menyimpan model kedalam 1174069.
   joblib
10 hasil = load('1174053.joblib') #Memanggil model 1174069

```

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

```

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

```

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

1.21.9 Penanganan Error

- ## 1. ScreenShoot Error

```
File "D:\Kuliahan\Semester 6\Kecerdasan Buatan\Bahan\1\sklearn.py", line 8, in <module>
    from sklearn import datasets

ImportError: cannot import name 'datasets' from 'sklearn' (D:\Kuliahan\Semester 6\Kecerdasan Buatan\Bahan\1\sklearn.py)
```

Gambar 1.108 Import Error

2. Tuliskan Kode Error dan Jenis Error

- #### ▪ Import Error

- ### 3. Cara Penangan Error

- Import Error

Dengan Menginstall Library Yang Tidak Ditemukan

1.21.10 Bukti Tidak Plagiat



Gambar 1.109 Bukti Tidak Melakukan Plagiat Chapter 1

1.21.11 Link Youtube

Buka Link

BAB 2

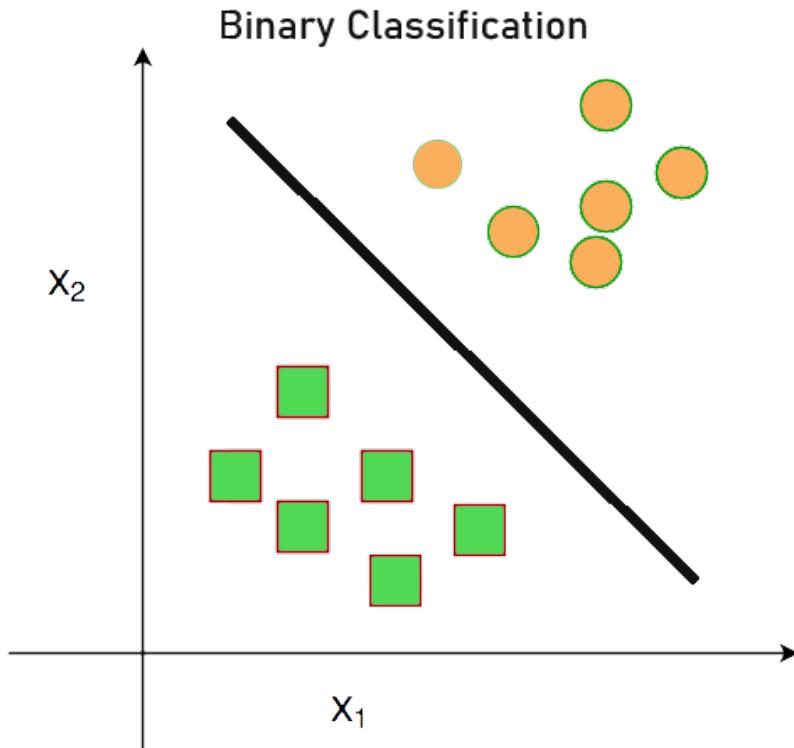
CHAPTER 2

2.1 1174083 - Bakti Qilan Mufid

Chapter 2 - Membangun model prediksi

2.1.1 Teori

2.1.1.1 *Jelaskan Apa Itu Binary Classification dilengkapi ilustrasi gambar sendiri.*



Gambar 2.1 gambaran binary classification

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

- Tes medis untuk menentukan apakah pasien memiliki penyakit tertentu atau tidak - properti klasifikasi adalah keberadaan penyakit.
- Metode uji "lulus atau gagal" atau kontrol kualitas di pabrik, yaitu memutuskan apakah suatu spesifikasi telah atau belum terpenuhi - klasifikasi Go/no go.
- Pengambilan informasi, yaitu memutuskan apakah suatu halaman atau artikel harus ada dalam hasil pencarian atau tidak - properti klasifikasi adalah relevansi artikel.

2.1.1.2 Jelaskan Apa itu supervised learning , unsupervised learning dan clustering dengan ilustrasi gambar sendiri.

1. supervised learning

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

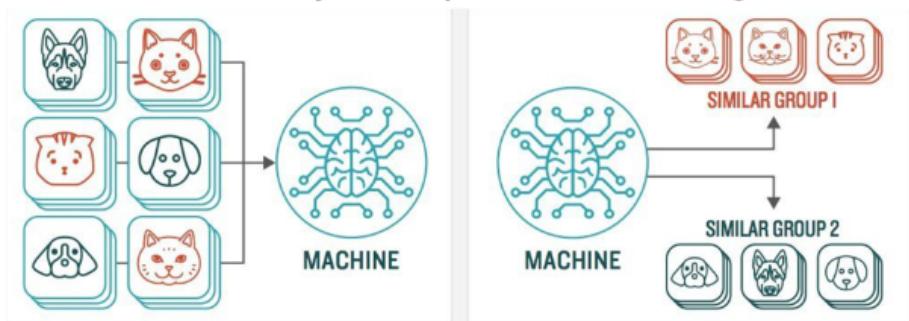


Gambar 2.2 gambaran cara kerja supervised

2. unsupervised learning

Unsupervised learning memiliki keunggulan dari supervised learning. Jika supervised learning memiliki label sebagai dasar prediksi baik serta membuat clasification dan regression algorithm yang memungkinkan. Tetapi dalam realitanya, data real itu banyak yang tidak memiliki label. Unsupervised learning menggunakan ke samaan dari attribut attribut yang dimiliki. Jika attribut dan sifat sifat dari data data feature yang diekstrak memiliki kemirip miripan, maka akan dikelompok kelompokan (clustering). Sehingga hal ini akan menimbulkan kelompok kelompok (cluster). Jumlah cluster bisa unlimited. Dari kelompok kelompok itu model mela belkan, dan jika data baru mau di prediksi, maka akan dicocok kan dengan kelompok yang mirip mirip featurenya.

Cara Kerja Unsupervised Learning



Gambar 2.3 gambaran cara kerja unsupervised

3. Clustering

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



Gambar 2.4 gambaran clustering

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

Evaluasi adalah tentang bagaimana kita dapat mengevaluasi seberapa baik model bekerja dengan mengukur akurasinya. Dan akurasi akan didefinisikan sebagai persentase kasus yang diklasifikasikan dengan benar. Kita dapat menganalisis kesalahan yang dibuat oleh model, atau tingkat kebingungannya, menggunakan matriks kebingungan. Matriks kebingungan mengacu pada kebingungan dalam model, tetapi matriks kebingungan ini bisa menjadi sedikit sulit untuk dipahami ketika mereka menjadi sangat besar.

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

Gambar 2.5 contoh confusion matrix

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

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

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

	Data asli 1 (positive)	Data asli 0 (negative)
Prediksi 1 (positive)	TP (True Positive)	FP (False Positive)
Prediksi 0 (negative)	FN (False Negative)	TN (True Negative)

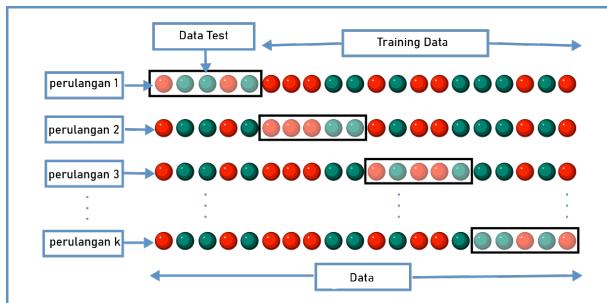
Gambar 2.6 contoh confusion matrix

2.1.1.5 Jelaskan bagaimana K-fold cross validation bekerja dengan gambar ilustrasi contoh buatan sendiri.

Cara kerja k-fold validation:

- Total instance dibagi menjadi N bagian.
- Fold yang pertama adalah bagian pertama menjadi data uji (testing data) dan sisanya menjadi training data.
- Lalu hitung akurasi berdasarkan porsi data tersebut dengan menggunakan persamaan.
- Fold yang kedua adalah bagian kedua, yang menjadi data uji(testing data)dan sisanya training data.
- Kemudian hitung akurasi berdasarkan porsi data tersebut.
- Dan seterusnya hingga habis mencapai fold ke-K.

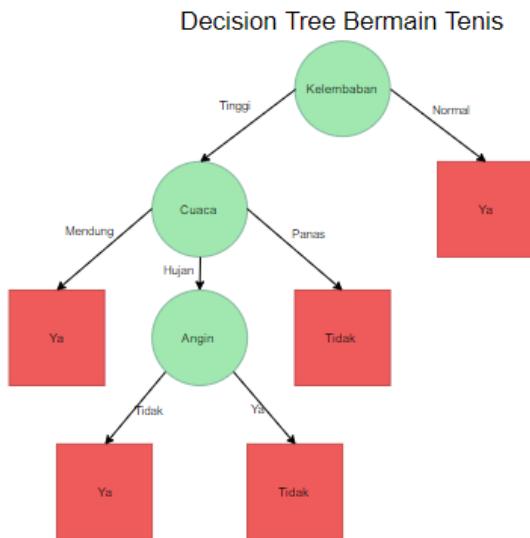
- Terakhir hitung rata-rata akurasi K buah.



Gambar 2.7 contoh K-Fold Validation

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

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



Gambar 2.8 Decision Tree bermain Tenis

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

1. information Gain

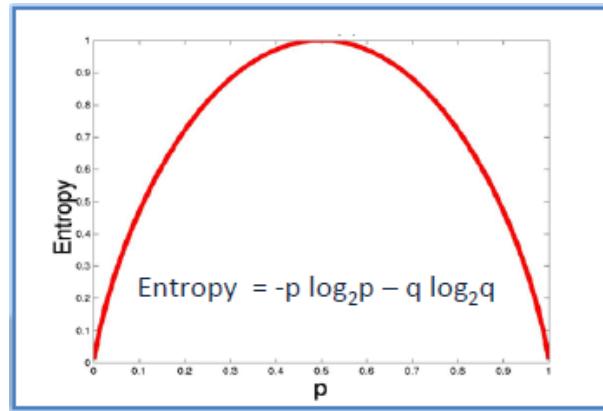
Information Gain merupakan total data yang didapat dari data - data acak yang data tersebut akan digunakan untuk analisis data lainnya. Information Gain ini digunakan pada decision tree sebagai label setiap aksi - aksi yang perlu dinilai validasinya.



Gambar 2.9 information gain

2. Entropi

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



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

Gambar 2.10 penggambaran entropi

2.1.2 Praktek

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

```
1 1174083 % 3
2 #Hasilnya = 0 maka mengambil variable dengan nama makanan
```

2.1.2.1 Praktek No. 1

```
1 # load dataset (menggunakan student-mat)
2 import pandas as pd
3 mochi = pd.read_csv('E://backup/sem 6/Kecerdasan Buatan/KB3C -
    Copy/src/1174083/src2/dataset/student-mat.csv', sep=';')
4 len(mochi)
```

mengimport library pandas lalu menamainya dengan pd. membuat variable mochi yang didalamnya terdefinisikan perintah untuk membaca file csv. dan len(mochi) untuk mengembalikan panjang jumlah anggota yang dimiliki variabel mochi.

```
In [6]: import pandas as pd
...: mochi = pd.read_csv('E://backup/sem 6/Kecerdasan Buatan/KB3C -
src2/dataset/student-mat.csv', sep=';')
...: len(mochi)
Out[6]: 395
```

Gambar 2.11 Loading Dataset

2.1.2.2 Praktek No. 2

```

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

```

pada bagian ini mendeklarasikan pass/fail nya data berdasarkan G1+G2+G3. Dengan ketentuan nilai passnya yaitu lebih besar/sama dengan 30. kemudian pada variabel mochi dideklarasikan jika baris dengan G1+G2+G3 ditambahkan,dan hasilnya lebih besar/sama dengan 35 maka axisnya 1. ketika dijalankan hasilnya seperti berikut

```

In [7]: mochi['pass'] = mochi.apply(lambda row: 1 if (row['G1']+row['G2']+row['G3']) >=
35 else 0, axis=1)
...: mochi = mochi.drop(['G1', 'G2', 'G3'], axis=1)
...: mochi.head()
Out[7]:
   school sex  age address famsize ... Dalc  Walc  health absences  pass
0      GP    F  18      U     GT3 ...  1    1     3       6     0
1      GP    F  17      U     GT3 ...  1    1     3       4     0
2      GP    F  15      U     LE3 ...  2    3     3      10     0
3      GP    F  15      U     GT3 ...  1    1     5       2     1
4      GP    F  16      U     GT3 ...  1    2     5       4     0
[5 rows x 31 columns]

```

Gambar 2.12 Generate Binary Label

2.1.2.3 Praktek No. 3

```

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

```

One-hot encoding adalah proses di mana variabel kategorikal dikonversi menjadi bentuk yang dapat disediakan untuk algoritma ML agar melakukan pekerjaan yang lebih baik dalam prediksi. Karena kita memuat data menggunakan pandas, disini digunakan fungsi panda pdgetdummies untuk jenis kelamin , sekolah, alamat dll. Metode head ini digunakan untuk mengembalikan baris n atas 5 secara default dari frame atau seri data. hasilnya seperti berikut

```
In [8]: mochi = pd.get_dummies(mochi, columns=['sex', 'school', 'address', 'famsize',
'Pstatus', 'Mjob', 'Fjob',
..., 'reason', 'guardian', 'schoolsup', 'famsup',
'paid', 'activities',
...: 'nursery', 'higher', 'internet', 'romantic'])
Out[8]:
   age  Medu  Fedu ...  internet_yes  romantic_no  romantic_yes
0    18      4     4 ...          0            1            0
1    17      1     1 ...          1            1            0
2    15      1     1 ...          1            1            0
3    15      4     2 ...          1            0            1
4    16      3     3 ...          0            1            0
[5 rows x 57 columns]
```

Gambar 2.13 One-hot Encoding

2.1.2.4 Praktek No. 4

```
1 # shuffle rows
2 mochi = mochi.sample(frac=1)
3 # split training and testing data
4 mochi_train = mochi[:500]
5 mochi_test = mochi[500:]
6
7 mochi_train_att = mochi_train.drop(['pass'], axis=1)
8 mochi_train_pass = mochi_train['pass']
9
10 mochi_test_att = mochi_test.drop(['pass'], axis=1)
11 mochi_test_pass = mochi_test['pass']
12
13 mochi_att = mochi.drop(['pass'], axis=1)
14 mochi_pass = mochi['pass']
15
16 # number of passing students in whole dataset:
17 import numpy as np
18 print("Passing: %d out of %d (%.2f%%)" % (np.sum(mochi_pass), len(mochi_pass),
   100*float(np.sum(mochi_pass)) / len(mochi_pass)
   ))
```

Sample digunakan untuk mengembalikan(return) sampel secara acak(item) dari objek. Pada bagian tersebut, terdapat train dan test yang digunakan untuk untuk membagi train, test dan kemudian membagi lagi train ke validasi dan test. Kemudian akan mengimport module numpy sebagai np yang akan digunakan untuk mengembalikan nilai passing dari pelajar dari keseluruhan dataset dengan cara print.

```
In [9]: mochi = mochi.sample(frac=1)
...: # split training and testing data
...: mochi_train = mochi[:500]
...: mochi_test = mochi[500:]
...:
...: mochi_train_att = mochi_train.drop(['pass'], axis=1)
...: mochi_train_pass = mochi_train['pass']
...:
...: mochi_test_att = mochi_test.drop(['pass'], axis=1)
...: mochi_test_pass = mochi_test['pass']
...:
...: mochi_att = mochi.drop(['pass'], axis=1)
...: mochi_pass = mochi['pass']
...:
...: # number of passing students in whole dataset:
...: import numpy as np
...: print("Passing: %d out of %d (%.2f%%)" % (np.sum(mochi_pass), len(mochi_pass),
100*float(np.sum(mochi_pass)) / len(mochi_pass)))
Passing: 166 out of 395 (42.03%)
```

Gambar 2.14 Shuffle Rows**2.1.2.5 Praktek No. 5**

```
1 # fit a decision tree
2 from sklearn import tree
3 cilok = tree.DecisionTreeClassifier(criterion="entropy",
4                                     max_depth=5)
4 cilok = cilok.fit(mochi_train_att, mochi_train_pass)
```

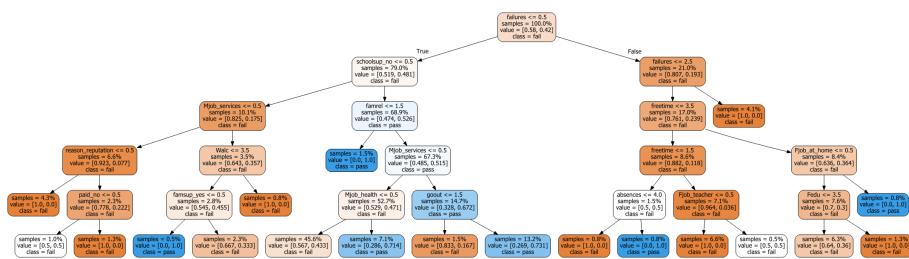
Dari librari scikit-learn import modul tree. Kemudian definisikan variabel Cilok dengan menggunakan DecisionClassifier.Kemudian pada variabel cilok terdapat Criterion yaitu suatu fungsi untuk mengukur kualitas split, setelah itu agar DecisionTreeClassifier dapat dijalankan gunakan perintah fit. hasilnya seperti dibawah

```
In [11]: from sklearn import tree
...: cilok = tree.DecisionTreeClassifier(criterion="entropy", max_depth=5)
...: cilok.fit(mochi_train_att, mochi_train_pass)
```

Gambar 2.15 Fit Decision tree**2.1.2.6 Praktek No. 6**

```
1 import graphviz
2 donat = tree.export_graphviz(cilok, out_file=None, label="all",
3                             impurity=False, proportion=True,
3                             feature_names=list(
4                               mochi_train_att), class_names=["fail", "pass"],
4                             filled=True, rounded=True)
5 kue = graphviz.Source(donat)
6 kue
```

Graphviz adalah perangkat lunak visualisasi grafik open source. Visualisasi grafik adalah cara mewakili informasi struktural sebagai diagram grafik dan jaringan abstrak. TREEEXPORTGRAPHVIZ merupakan fungsi yang menghasilkan representasi Graphviz dari decision tree,yang kemudian ditulis ke out file. Sehingga akan muncul gambar diagram grafik bercabang.



Gambar 2.16 Visualize tree

2.1.2.7 Praktek No. 7

```

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

```

TREEEXPORTGRAPHVIZ merupakan fungsi yang menghasilkan representasi Graphviz dari decision tree,yang kemudian dituliske out file.Disini akan disimpan classifiernya, yang setelahnya akan mengekspor file student performance dan jika salah akan mengembalikan nilai fail.

```
In [9]: tree.export_graphviz(cilok, out_file="student-performance.dot", label="all", impurity=False,
proportion=True,
...: feature_names=list(mochi_train_att), class_names=["fail", "pass"],
...: filled=True, rounded=True)
```

Gambar 2.17 menyimpan(save) tree

2.1.2.8 Praktek No. 8

```
1 cilok.score(mochi_test_att, mochi_test_pass)
```

Score juga disebut prediksi,dan merupakan proses yang menghasilkan nilai berdasarkan model pembelajaran mesin yang terlatih, diberi beberapa data input baru. Nilai atau skor yang dibuat dapat mewakili prediksi nilai masa depan, tetapi mereka juga mungkin mewakili kategori atau hasil yang mungkin. Jadi disini cilok akan memprediksi nilai dari mochi_ test_ att dan mochi_ test_ pass. Hasilnya seperti dibawah ini

```
File "C:\ProgramData\Anaconda3\lib\site-packages\sklearn\utils\validation.py", line 550, in
check_array
    context)
ValueError: Found array with 0 sample(s) (shape=(0, 56)) while a minimum of 1 is required.
```

Gambar 2.18 Score(masih error)

2.1.2.9 Praktek No. 9

```

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

```

Script ini akan mengevaluasi score dengan validasi silang. Dimana variabel biskuit berisikan cross valscore yang merupakan fungsi pembantu pada estimator dan dataset. Kemudian akan menampilkan score rata rata dan kurang lebih dua standar deviasi yang mencakup 95% score. dengan menggunakan perintah print hasil yang didapatkan sebagai berikut

```

In [35]: from sklearn.model_selection import cross_val_score
...: biskuit = cross_val_score(cilok, mochi_att, mochi_pass, cv=5)
...: # show average score and +/- two standard deviations away (covering 95% of scores)
...: print("Accuracy: %0.2f (+/- %0.2f)" % (biskuit.mean(), biskuit.std() * 2))
Accuracy: 0.57 (+/- 0.06)

```

Gambar 2.19 Cross Val Score

2.1.2.10 Praktek No. 10

```

1 for max_depth in range(1, 20):
2     cilok = tree.DecisionTreeClassifier(criterion="entropy",
3                                         max_depth=max_depth)
4     biskuit = cross_val_score(cilok, mochi_att, mochi_pass, cv=5)
5     print("Max depth: %d, Accuracy: %0.2f (+/- %0.2f)" % (
        max_depth, biskuit.mean(), biskuit.std() * 2))

```

Pada script ini menunjukkan seberapa dalam tree itu. Semakin dalam tree, semakin banyak perpecahan yang dimilikinya dan menangkap lebih banyak informasi tentang data. variabel cilok akan mendefinisikan tree nya yang kemudian variabel biskuit akan mengevaluasi score dengan validasi silang. disini mendefinisikan decision tree dengan kedalaman mulai dari 1 hingga 19 dan merencanakan pelatihan dan menguji skor auc. Jika di run hasilnya seperti berikut

```
In [36]: for max_depth in range(1, 20):
...:     t = tree.DecisionTreeClassifier(criterion="entropy", max_depth=max_depth)
...:     biskuit = cross_val_score(cilok, mochi_att, mochi_pass, cv=5)
...:     print("Max depth: %d, Accuracy: %.2f (%-.2f)" % (max_depth, biskuit.mean(),
biskuit.std() * 2))
Max depth: 1, Accuracy: 0.57 (+/- 0.05)
Max depth: 2, Accuracy: 0.57 (+/- 0.03)
Max depth: 3, Accuracy: 0.57 (+/- 0.04)
Max depth: 4, Accuracy: 0.57 (+/- 0.06)
Max depth: 5, Accuracy: 0.57 (+/- 0.06)
Max depth: 6, Accuracy: 0.57 (+/- 0.03)
Max depth: 7, Accuracy: 0.57 (+/- 0.06)
Max depth: 8, Accuracy: 0.57 (+/- 0.06)
Max depth: 9, Accuracy: 0.57 (+/- 0.06)
Max depth: 10, Accuracy: 0.56 (+/- 0.05)
Max depth: 11, Accuracy: 0.56 (+/- 0.05)
Max depth: 12, Accuracy: 0.57 (+/- 0.05)
Max depth: 13, Accuracy: 0.57 (+/- 0.06)
Max depth: 14, Accuracy: 0.58 (+/- 0.05)
Max depth: 15, Accuracy: 0.57 (+/- 0.05)
Max depth: 16, Accuracy: 0.58 (+/- 0.05)
Max depth: 17, Accuracy: 0.57 (+/- 0.06)
Max depth: 18, Accuracy: 0.57 (+/- 0.05)
Max depth: 19, Accuracy: 0.57 (+/- 0.06)
```

Gambar 2.20 Max Depth

2.1.2.11 Praktek No. 11

```
1 depth_acc = np.empty((19,3), float)
2 i = 0
3 for max_depth in range(1, 20):
4     cilok = tree.DecisionTreeClassifier(criterion="entropy",
5         max_depth=max_depth)
6     biskuit = cross_val_score(cilok, mochi_att, mochi_pass, cv=5)
7     depth_acc[i,0] = max_depth
8     depth_acc[i,1] = biskuit.mean()
9     depth_acc[i,2] = biskuit.std() * 2
10    i += 1
depth_acc
```

Depth acc akan membuat array kosong dengan mengembalikan array baru dengan bentuk dan tipe yang diberikan, tanpa menginisialisasi entri. Dengan 19 sebagai bentuk array kosong, 3 sebagai output data-type dan float urutan kolom utama (gaya Fortran) dalam memori. variabel cilok yang akan melakukan split score dan biskuit akan memvalidasi score secara silang. dan pada akhirnya biskuit.std() yaitu menghitung standar deviasi dari data yang diberikan (elemen array) di sepanjang sumbu yang ditentukan (jika ada), hasilnya sebagai berikut

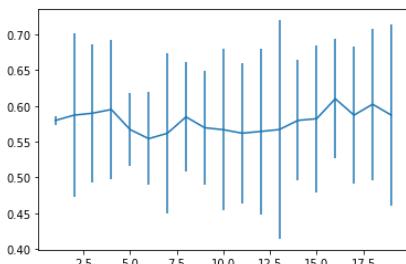
```
In [37]: depth_acc = np.empty((19,3), float)
...: i = 0
...: for max_depth in range(1, 20):
...:     ciklok = tree.DecisionTreeClassifier(criterion="entropy", max_depth=max_depth)
...:     biskuit = cross_val_score(ciklok, mochi_att, mochi_pass, cv=5)
...:     depth_acc[i,0] = max_depth
...:     depth_acc[i,1] = biskuit.mean()
...:     depth_acc[i,2] = biskuit.std() * 2
...:     i += 1
...:
...: depth_acc
Out[37]:
array([[1.00000000e+00, 5.79751704e-01, 6.30768599e-03],
       [2.00000000e+00, 5.87251704e-01, 1.14964230e-01],
       [3.00000000e+00, 5.89976459e-01, 9.63921191e-02],
       [4.00000000e+00, 5.95137131e-01, 9.70350479e-02],
       [5.00000000e+00, 5.67157579e-01, 5.14546483e-02],
       [6.00000000e+00, 5.54338689e-01, 6.48484045e-02],
       [7.00000000e+00, 5.61887043e-01, 1.12852270e-01],
       [8.00000000e+00, 5.84783350e-01, 7.60786737e-02],
       [9.00000000e+00, 5.69625933e-01, 7.96744216e-02],
       [1.00000000e+01, 5.66871957e-01, 1.12614727e-01],
       [1.10000000e+01, 5.61997728e-01, 9.84139767e-02],
       [1.20000000e+01, 5.64402761e-01, 1.15727593e-01],
       [1.30000000e+01, 5.67094268e-01, 1.52875902e-01],
       [1.40000000e+01, 5.79880721e-01, 8.40431882e-02],
       [1.50000000e+01, 5.82253327e-01, 1.02691451e-01],
       [1.60000000e+01, 6.10197176e-01, 8.37263419e-02],
       [1.70000000e+01, 5.87220058e-01, 9.59407620e-02],
       [1.80000000e+01, 6.02505680e-01, 1.059530898e-01],
       [1.90000000e+01, 5.87218436e-01, 1.26173883e-01]])
```

Gambar 2.21 Depth in Range**2.1.2.12 Praktek No. 12**

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

Mengimpor librari dari matplotlib yaitu pyplot sebagai plt, fig dan puding menggunakan subplots untuk membuat gambar dan satu set subplot. puding.errorbar akan membuat error bar kemudian grafik akan ditampilkan menggunakan show. Grafiknya seperti berikut

```
In [38]: import matplotlib.pyplot as plt
...: fig, puding = plt.subplots()
...: puding.errorbar(depth_acc[:,0], depth_acc[:,1], yerr=depth_acc[:,2])
...: plt.show()
```

**Gambar 2.22** Matplotlib

2.1.3 Penanganan Error

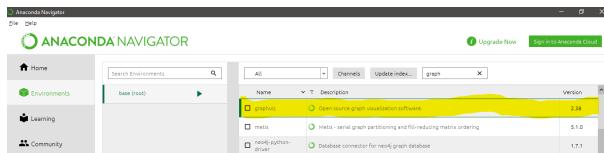
2.1.3.1 Error ke-1

```
Traceback (most recent call last):
  File "<ipython-input-12-952848583a25>", line 1, in <module>
    import graphviz
ModuleNotFoundError: No module named 'graphviz'
```

Gambar 2.23 No module named graphviz

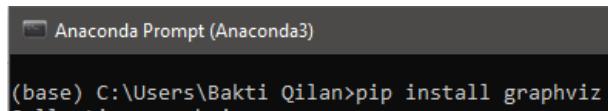
Solusi:

- buka anaconda navigator > Environment > lalu install graphviz



Gambar 2.24 Solusi dengan anaconda navigator

- buka anaconda prompt(Run As Admin) > lalu ketikan kode seperti berikut



Gambar 2.25 Solusi dengan anaconda prompt

2.1.3.2 Error ke-2

```
raise ExecutableNotFoundError(cmd)
ExecutableNotFoundError: failed to execute ['dot', '-Tsvg'], make sure the Graphviz
executables are on your systems' PATH
Out[13]: <graphviz.files.Source at 0x2274f1e6b48>
```

Gambar 2.26 graphviz executables

Pada gambar diatas kode erornya adalah ExecutableNotFoundError failed to execute dot Tsvg. Eror ini terjadi karena tidak terdaftarnya environment variable dari Graphviz pada PATH di PC.

Solusi: menambahkan kode berikut:

```
1 import os
2 os.environ["PATH"] += os.pathsep + 'C:/ProgramData/Anaconda3/
   Library/bin/graphviz/'
```

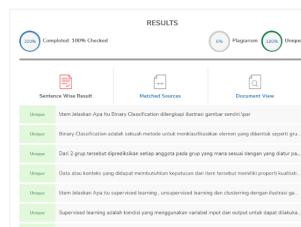
2.1.3.3 Error ke-3

```
In [34]: cilok.score(mochi_test_att, mochi_test_pass)
Traceback (most recent call last):
  File "<ipython-input-34-df50da7a7eb4>", line 1, in <module>
    cilok.score(mochi_test_att, mochi_test_pass)
  File "C:\ProgramData\Anaconda3\lib\site-packages\sklearn\base.py", line 357, in score
    return accuracy_score(y, self.predict(X), sample_weight=sample_weight)
  File "C:\ProgramData\Anaconda3\lib\site-packages\sklearn\tree.py", line 430, in predict
    X = self._validate_X_predict(X, check_input)
  File "C:\ProgramData\Anaconda3\lib\site-packages\sklearn\tree.py", line 391, in
    _validate_X_predict
    X = check_array(X, dtype=DTYPE, accept_sparse="csr")
  File "C:\ProgramData\Anaconda3\lib\site-packages\sklearn\utils\validation.py", line 550, in
    check_array
    context)
ValueError: Found array with 0 sample(s) (shape=(0, 56)) while a minimum of 1 is required.
```

Gambar 2.27 Masih belum Paham

Solusi: 404 Not Found. Pusing slur!

2.1.4 Bukti Tidak Plagiat



Gambar 2.28 plagiarism di smallseotools

2.1.5 Link Video Youtube

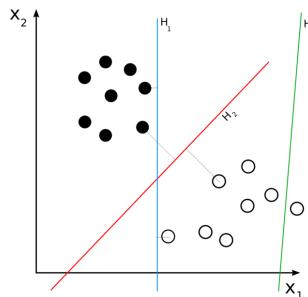
<https://youtu.be/19GeYn55DvU>

2.2 Fanny Shafira Damayanti (1174069)

2.2.1 Teori

1. Jelaskan apa itu binary classification dilengkapi ilustrasi gambar sendiri
Klasifikasi biner tugasnya yaitu untuk mengklasifikasikan elemen-elemen

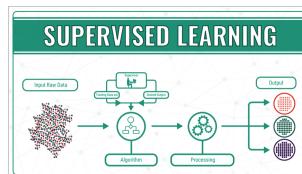
dari himpunan yang diberikan ke dalam dua kelompok berdasarkan aturan klasifikasi.



Gambar 2.29 Binary Classification

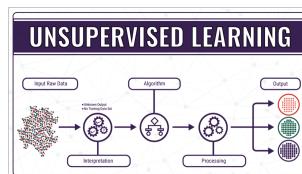
2. Jelaskan apa itu supervised learning dan unsupervised learning dan clustering dengan ilustrasi gambar sendiri

Supervised Learning adalah tipe learning di mana kita mempunyai variable input dan variable output, dan menggunakan satu algoritma atau lebih untuk mempelajari fungsi pemetaan dari input ke output.



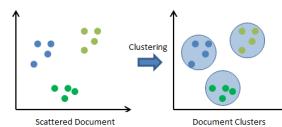
Gambar 2.30 Supervised Learning

Unsupervised learning adalah tipe learning di mana kita hanya mempunyai data masukan (input data) tetapi tidak ada output variable yang berhubungan.



Gambar 2.31 Unsupervised learning

Analisis atau pengelompokan klaster adalah tugas pengelompokan sekumpulan objek sedemikian rupa sehingga objek dalam kelompok yang sama lebih mirip satu sama lain daripada pada kelompok lain.



Gambar 2.32 Clustering

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

Evaluasi merupakan saduran dari bahasa Inggris "evaluation" yang diartikan sebagai penaksiran atau penilaian. Nurkancana (1983) menyatakan bahwa evaluasi adalah kegiatan yang dilakukan berkenaan dengan proses untuk menentukan nilai dari suatu hal.

Akurasi adalah derajat kesesuaian, yaitu tingkat yang mana pengukuran adalah tepat ketika dibandingkan dengan nilai absolut.

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

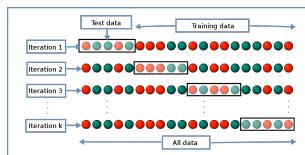
Confusion matrix adalah suatu metode yang biasanya digunakan untuk melakukan perhitungan akurasi pada konsep data mining atau Sistem Pendukung Keputusan. Pada pengukuran kinerja menggunakan confusion matrix, terdapat 4 (empat) istilah sebagai representasi hasil proses klasifikasi. Keempat istilah tersebut adalah True Positive (TP), True Negative (TN), False Positive (FP) dan False Negative (FN). Nilai True Negative (TN) merupakan jumlah data negatif yang terdeteksi dengan benar, sedangkan False Positive (FP) merupakan data negatif namun terdeteksi sebagai data positif. Sementara itu, True Positive (TP) merupakan data positif yang terdeteksi benar. False Negative (FN) merupakan kebalikan dari True Positive, sehingga data positif, namun terdeteksi sebagai data negatif.

		True Values	
		True	False
Prediction	True	TP Correct result	FP Unexpected result
	False	FN Missing result	TN Correct absence of result

Gambar 2.33 Clustering

5. Jelaskan bagaimana K-fold cross validation bekerja dengan gambar ilustrasi contoh buatan sendiri.

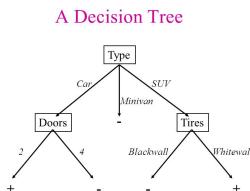
Cross-validasi, atau bisa disebut estimasi rotasi, adalah sebuah teknik validasi model untuk menilai bagaimana hasil statistik analisis akan menggeneralisasi kumpulan data independen. Teknik ini utamanya digunakan untuk melakukan prediksi model dan memperkirakan seberapa akurat sebuah model prediktif ketika dijalankan dalam praktiknya. Dalam sebuah masalah prediksi, sebuah model biasanya diberikan kumpulan data (dataset) yang diketahui untuk digunakan dalam menjalankan pelatihan (dataset pelatihan), serta kumpulan data yang tidak diketahui (atau data yang pertama kali dilihat) terhadap model yang diuji (pengujian dataset). Tujuan dari validasi silang adalah untuk mendefinisikan dataset untuk "menguji" model dalam tahap pelatihan (yaitu, validasi data), dalam rangka untuk membatasi masalah seperti terjadinya overfitting, memberikan wawasan tentang bagaimana model akan menggeneralisasi independen dataset (yaitu, dataset tidak diketahui, misalnya dari masalah nyata), dll.



Gambar 2.34 K-fold cross validation

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

Pohon keputusan adalah alat pendukung keputusan yang menggunakan model keputusan seperti pohon dan konsekuensinya yang mungkin, termasuk hasil acara kebetulan, biaya sumber daya, dan utilitas. Ini adalah salah satu cara untuk menampilkan algoritma yang hanya berisi pernyataan kontrol bersyarat.



Gambar 2.35 Decision Tree

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

Metode Information Gain adalah metode yang menggunakan teknik scoring untuk pembobotan sebuah fitur dengan menggunakan maksimal entropy.

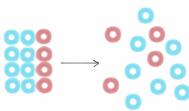
Information Gain

$$IG(D_p, f) = I(D_p) - \frac{N_{left}}{N} I(D_{left}) - \frac{N_{right}}{N} I(D_{right})$$

t: feature selected
 D_p: dataset of the parent node
 D_{left}: dataset of the left child node
 D_{right}: dataset of the right child node
 I: impurity measure (Gini or Entropy)
 N: total number of samples
 N_{left}: number of samples at left child node
 N_{right}: number of samples at right child node
 I_t: impurity of feature t

Gambar 2.36 Information Gain

Entropi adalah salah satu besaran termodinamika yang mengukur energi dalam sistem per satuan temperatur yang tak dapat digunakan untuk melakukan usaha.



Gambar 2.37 Entropy

2.2.2 scikit-learn

```

1 # -*- coding: utf-8 -*-
2 """
3 Created on Sat Mar  7 23:25:07 2020
4
5 @author: FannyShafira
6 """
7
8 print(1174069%3)
9 %% 1.Load Dataset
10 import pandas as pd # load dataset (menggunakan student-mat.csv)
11 padalarang = pd.read_csv('F://Semester 6/Artificial Intelligence/
   Tugas 2/src/dataset/student-mat.csv', sep=';') #variabel
   padalarang berfungsi untuk read file student-mat.csv
12 len(padalarang) #mengetahui jumlah baris pada data yang dipanggil
13
14 %% 2.generate binary label (pass/fail) based on G1+G2+G3
15 padalarang['pass'] = padalarang.apply(lambda row: 1 if (row['G1']+
   row['G2']+row['G3'])>= 35 else 0, axis=1)#mendeklarasikan
   pass/fail nya data berdasarkan G1+G2+G3.

```

```

16 padalarang = padalarang.drop(['G1', 'G2', 'G3'], axis=1)#untuk
   mengetahui baris G1+G2+G3 ditambahkan, dan hasilnya sama
   dengan 35 maka axisnya 1.
17 padalarang.head() #memanggil variabel padalarang dengan ketentuan
   head ini digunakan untuk mengembalikan baris n atas 5 secara
   default dari frame atau seri data
18
19 #%% 3.use one-hot encoding on categorical columns
20 padalarang = pd.get_dummies(padalarang,columns=['sex','school',
   'address','famsize','Pstatus','Mjob','Fjob','reason','guardian',
   'schoolsup','famsup','paid','activities','nursery','higher',
   'internet','romantic'])
21 padalarang.head()#memanggil variabel padalarang dengan ketentuan
   head ini digunakan untuk mengembalikan baris n atas 5 secara
   default dari frame atau seri data
22
23 #%% 4.shuffle rows
24 padalarang = padalarang.sample(frac=1)#mengembalikan variabel
   padalarang menjadi sampel acak dengan frac=1
25 padalarang_train = padalarang[:500]
26 padalarang_test = padalarang[500:]
27 padalarang_train_att = padalarang_train.drop(['pass'], axis=1)
28 padalarang_train_pass = padalarang_train['pass']
29 padalarang_test_att = padalarang_test.drop(['pass'], axis=1)
30 padalarang_test_pass = padalarang_test['pass']
31 padalarang_att = padalarang.drop(['pass'], axis=1)
32 padalarang_pass = padalarang['pass']
33
34 import numpy as np #mengimport module numpy sebagai np y
35 print("Passing: %d out %d (%.2f%%)" %(np.sum(padalarang_pass),len(
   padalarang_pass),100* float(np.sum(padalarang_pass))/len(
   padalarang_pass)))
36 #%% 5.fit a decision tree
37 from sklearn import tree
38 bandung = tree.DecisionTreeClassifier(criterion="entropy",
   max_depth=5) #membuat variabel bandung sebagai decisiontree ,
   dengan criterion fungsi mengukur kualitas split
39 bandung = bandung.fit(padalarang_train_att, padalarang_train_pass)
40
41 #%% 6.visualize tree
42 import os
43 os.environ["PATH"] += os.pathsep + 'D:/graphviz-2.38/release/bin'
44
45 import graphviz
46 bogor = tree.export_graphviz(bandung, out_file=None, label ="all",
   impurity=False, proportion=True, feature_names=list(
   padalarang_train_att), class_names=["fail","pass"], filled=True,
   rounded=True)#mengambil data untuk diterjemahkan ke grafik
47 jakarta = graphviz.Source(bogor)
48 jakarta
49
50 #%% 7.save tree
51 tree.export_graphviz(bandung, out_file="student-performance.dot",
   label ="all",impurity=False, proportion=True, feature_names=
   list(padalarang_train_att), class_names=["fail","pass"], filled

```

```

= True, rounded=True) #save tree sebagai export graphviz ke file
student-performance.dot

52
53 %% 8
54 bandung.score(padalarang_test_att, padalarang_test_pass) #score
    juga disebut prediksi dengan diberi beberapa data input baru
55
56 %% 9
57 from sklearn.model_selection import cross_val_score
58 depok = cross_val_score(bandung, padalarang_att, padalarang_pass, cv
    = 5) #meng evaluasi score dengan validasi silang
59 # show average score and +/- two standard deviations away (
    covering 95% of scores)
60 print("Accuracy : %0.2f (+/- %0.2f)" % (depok.mean(), depok.std()
    * 2))
61
62 %% 10
63 for surabaya in range(1,20):
    bandung = tree.DecisionTreeClassifier(criterion="entropy",
        max_depth=surabaya)
    depok = cross_val_score(bandung, padalarang_att,
        padalarang_pass, cv=5)
    print("Max depth : %d, Accuracy : %0.2f (+/- %0.2f)" %(surabaya,
        depok.mean(), depok.std() * 2))
64 #Disini ini menunjukkan seberapa dalam di tree itu. Semakin dalam
    tree, semakin banyak perpecahan yang dimilikinya dan
    menangkap lebih banyak informasi tentang data.
65 %% 11
66 medan = np.empty((19,3),float)
67 sidoarjo = 0
68 for surabaya in range(1,20):
    bandung = tree.DecisionTreeClassifier(criterion="entropy",
        max_depth=surabaya) #variabel bandung untuk decision tree
        dengan ketentuan entropy
    depok = cross_val_score(bandung, padalarang_att,
        padalarang_pass, cv=5)
    medan[sidoarjo,0] = surabaya
    medan[sidoarjo,1] = depok.mean()
    medan[sidoarjo,2] = depok.std() * 2
    sidoarjo += 1
69 medan
70
71 %% 12
72 import matplotlib.pyplot as plt
73 blitar, kediri = plt.subplots()
74 kediri.errorbar(medan[:,0], medan[:,1], yerr=medan[:,2]) #membuat
    error bar kemudian grafik akan ditampilkan menggunakan show
75 plt.show()

```

2.2.3 Pengangan Error

1. Screenshot Error

FileNotFoundException: [Errno 2] File b'student-mat.csv' does not exist: b'student-mat.csv'

Gambar 2.38 File Not Found

```
File "<ipython-input-13-d089b825cled>", line 1, in <module>
    import graphviz
ModuleNotFoundError: No module named 'graphviz'
```

Gambar 2.39 Module Not Found

```
|   file "<ipython-input-13-d089b825cled>", line 1, in <module>
|       import graphviz
|   ModuleNotFoundError: No module named 'graphviz'
|
|   ExecutableNotFoundError: failed to execute ['dot', '-Tsvg'], make sure the Graphviz
|   executables are on your systems' PATH
|
Out[35]: <graphviz.files.Source at 0x2f0fae3d80>
```

Gambar 2.40 Executable Not Found

2. Jenis Error

- File Not Found
- Module Not Found
- Executable Not Found

3. Solusi Error

- File Not Found
Mendownload filenya di github

**Gambar 2.41** File Dataset

- Module Not Found
Mendownload library graphviz menggunakan pip install graphviz di Anaconda Prompt

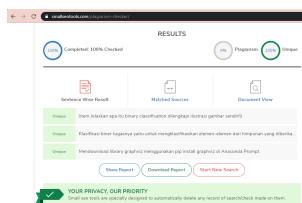
```
ImportError: No module named 'graphviz'
Requirement: https://github.com/jonathanschlueter/python-graphviz/archive/v0.11.tar.gz
  -> https://github.com/jonathanschlueter/python-graphviz/archive/v0.11.tar.gz
  -> https://github.com/jonathanschlueter/python-graphviz/commit/0.11.0
  -> https://github.com/jonathanschlueter/python-graphviz/commit/0.11.0
```

Gambar 2.42 Install graphviz

- Executable Not Found Memasukkan PATH dari graphviz

```
1 import os
2 os.environ["PATH"] += os.pathsep + 'D:/graphviz-2.38/
release/bin'
```

2.2.4 Scan Plagiarisme



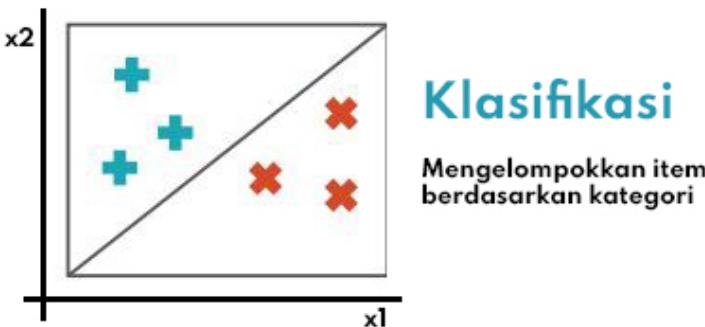
Gambar 2.43 Scan plagiarisme

2.2.5 Link Youtube

<https://youtu.be/ZLN9sCwl9-I>

2.3 D.Irga B. Naufal Fakhri (1174066)

2.3.1 Teori



Gambar 2.44 Binary Classification

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

1. Tes medis

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

2. Metode uji "lulus atau gagal"

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

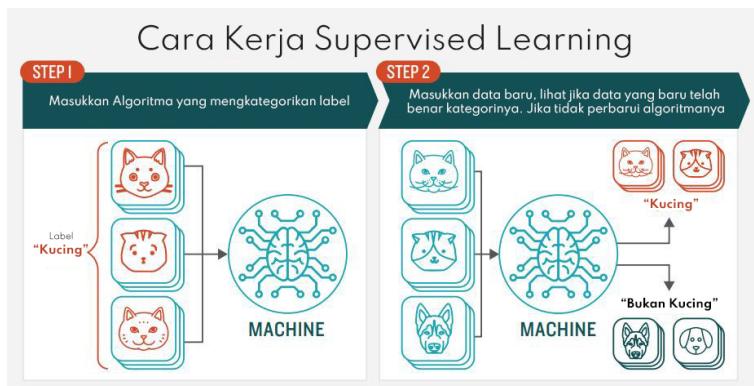
3. Pengambilan informasi

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

2.3.1.2 *Supervised Learning , Unsupervised Learning dan Clustering*

1. Supervised Learning

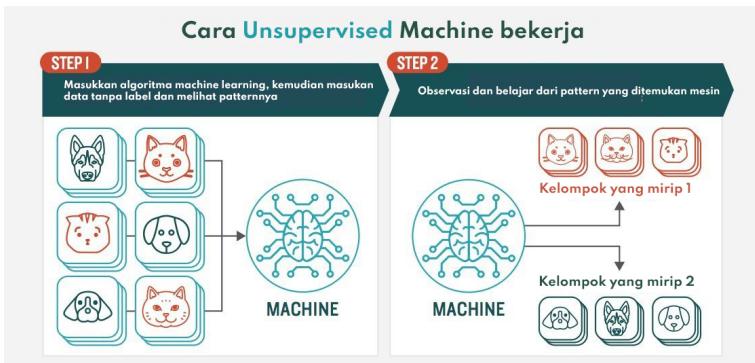
Supervised learning adalah suatu pembelajaran yang terawasi dimana jika output yang diharapkan telah diketahui sebelumnya. Biasanya pembelajaran ini dilakukan dengan menggunakan data yang telah ada. Dan Supervised Learning dalam bahasa indonesia adalah pembelajaran yang ada supervisornya. Maksud disini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



Gambar 2.45 Supervised Learning

2. Unsupervised Learning

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



Gambar 2.46 Unsupervised Learning

3. Clustering

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



Gambar 2.47 Clustering

2.3.1.3 Evaluasi dan Akurasi Evaluasi adalah tentang bagaimana kita dapat mengevaluasi seberapa baik model bekerja dengan mengukur tingkat akurasinya. Dan akurasi akan didefinisikan sebagai persentase kasus yang diklasifikasikan dengan benar. Kita dapat menganalisis kesalahan yang dibuat oleh model, atau tingkat kebingungannya, menggunakan matriks kebingungan(confusion matrix). Matriks kebingungan mengacu pada kebingungan dalam model,tetapi matriks kebingungan ini bisa menjadi sedikit sulit untuk dipahami ketika mereka menjadi sangat besar.

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

Gambar 2.48 Evaluasi

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

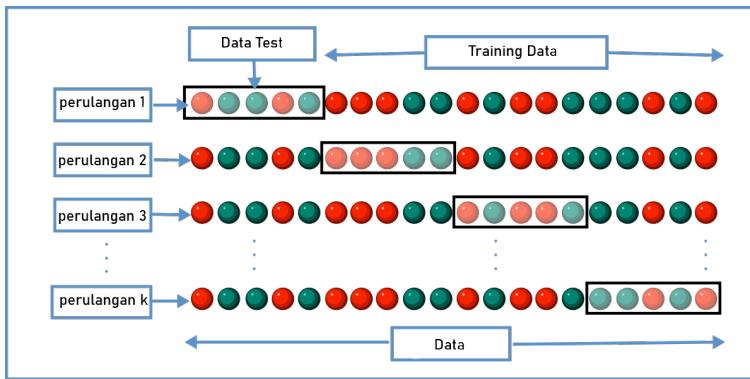
- True Positive: Data positif yang terdeteksi memiliki hasil benar
- False Positive: Data Positif yang terdeteksi memiliki hasil salah
- True Negative: Data negatif yang terdeteksi memiliki hasil benar
- False Negative: Data negatif yang terdeteksi memiliki hasil salah

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

Gambar 2.49 Contoh Confusion Matrix

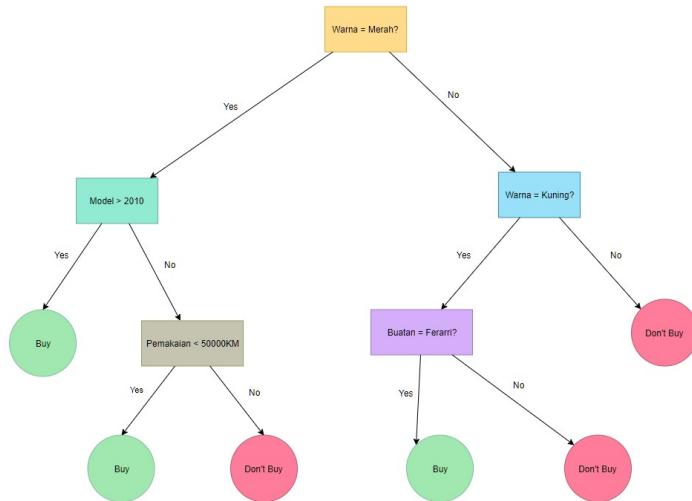
2.3.1.5 Bagaimana K-fold cross validation bekerja

1. Total instance dibagi menjadi N bagian.
2. Fold yang pertama adalah bagian pertama menjadi data uji (testing data) dan sisanya menjadi training data.
3. Lalu hitung akurasi berdasarkan porsi data tersebut dengan menggunakan persamaan.
4. Fold yang kedua adalah bagian kedua, yang menjadi data uji(testing data) dan sisanya training data.
5. Kemudian hitung akurasi berdasarkan porsi data tersebut.
6. Dan seterusnya hingga habis mencapai fold ke-K.
7. Terakhir hitung rata-rata akurasi K buah.



Gambar 2.50 Contoh K-Fold Cross Validation

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



Gambar 2.51 Contoh Decision Tree membeli mobil

2.3.1.7 Information Gain dan Entropi

- Information Gain

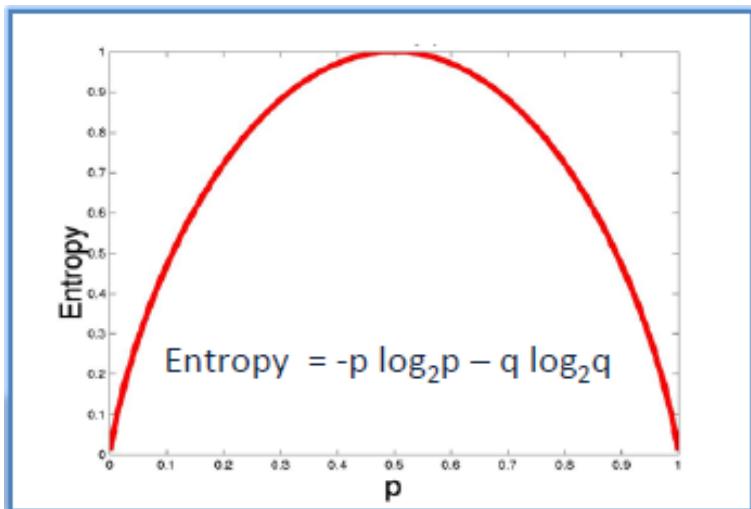
Information Gain merupakan total data yang didapat dari data - data acak yang data tersebut akan digunakan untuk analisis data lainnya. Information Gain ini digunakan pada decision tree sebagai label setiap aksi - aksi yang perlu dinilai validasinya.



Gambar 2.52 Information Gain

- Entropi

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



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

Gambar 2.53 Entropi

2.3.2 Praktek

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

```

1 # In [0]
2 1174066 % 3 #Hasilnya 1 maka akan menggunakan nama Kota

```

2.3.2.1 Nomor 1

```

1 import pandas as pd #Import library pandas menggantinya nama yang
    akan dipanggil jadi pd
2 tokyo = pd.read_csv('dataset/student-mat.csv', sep=';') #Membuat
    variable tokyo yang isinya memanggil fungsi membaca file csv
3 len(tokyo) #Menghitung jumlah data yang ada pada csv yang tadi
    sudah dibaca

```

```
[2]: %%file config.yaml
# Created on Sun Mar 6 12:58:02 2020
# Author: Arin
# -----
# $179465 S: (empty) I want about management name data
# -----
# Import gender as pd
# take x = read_csv('dataset/student-mat.csv', sep=';')
# df
```

Gambar 2.54 Nomor 1

2.3.2.2 Nomor 2

```
1 # In [2]
2 # generate binary label (pass/fail) based on G1+G2+G3 (test
3 #   grades, each 0-20 pts); threshold for passing is sum>=30
4 tokyo['pass'] = tokyo.apply(lambda row: 1 if (row['G1']+row['G2']
5 #   )+row['G3']) >= 35 else 0, axis=1) #Membuat label binary (
6 #   pass/fail) berdasarkan G1+G2+G3 (testgrade, semuanya 0-20
7 #   point); Batas untuk pass adalah sum>=30
8 tokyo = tokyo.drop(['G1', 'G2', 'G3'], axis=1) #Meghilangkan data
9 #   G1 G2 dan G3
10 tokyo.head() #Menampilkan data
```

```
[4]:  
# Load the data from student-mat.csv  
df = pd.read_csv('student-mat.csv', sep=';')  
  
tokyo['mat'] = tokyo['mat'].apply(lambda row: 1 if (row[0] == 'G') | (row[1] == 'G') | (row[2] == 'G') | (row[3] == 'G') else 0, axis=1)  
tokyo['drugs'] = tokyo['drugs'].apply(lambda row: 1 if (row[0] == 'yes') | (row[1] == 'yes') | (row[2] == 'yes') | (row[3] == 'yes') else 0, axis=1)  
tokyo['absent'] = tokyo['absent'].apply(lambda row: 1 if (row[0] == 'yes') | (row[1] == 'yes') | (row[2] == 'yes') | (row[3] == 'yes') else 0, axis=1)  
  
Variable explorer: No editor: Help  
Python console  
In [1]:  
# Load the data from student-mat.csv  
tokyo = pd.read_csv('student-mat.csv', sep=',')  
  
Out[1]: 395  
  
In [2]: tokyo['mat'] = 1 if (tokyo['G1'] == 'G') | (tokyo['G2'] == 'G') | (tokyo['G3'] == 'G') | (tokyo['G4'] == 'G') else 0, axis=1  
tokyo['drugs'] = 1 if (tokyo['D1'] == 'yes') | (tokyo['D2'] == 'yes') | (tokyo['D3'] == 'yes') | (tokyo['D4'] == 'yes') else 0, axis=1  
tokyo['absent'] = 1 if (tokyo['AS1'] == 'yes') | (tokyo['AS2'] == 'yes') | (tokyo['AS3'] == 'yes') | (tokyo['AS4'] == 'yes') else 0, axis=1  
  
tokyo.head(10).to_csv('test_0-9558828095.csv', line_terminator='\r\n')  
tokyo.to_csv('test_0-9558828095.csv', line_terminator='\r\n')  
  
NameError: name 'AS4' is not defined  
  
In [3]:  
tokyo['mat'] = tokyo['mat'].apply(lambda row: 1 if (row[0] == 'G') | (row[1] == 'G') | (row[2] == 'G') | (row[3] == 'G') else 0, axis=1)  
tokyo['drugs'] = tokyo['drugs'].apply(lambda row: 1 if (row[0] == 'yes') | (row[1] == 'yes') | (row[2] == 'yes') | (row[3] == 'yes') else 0, axis=1)  
tokyo['absent'] = tokyo['absent'].apply(lambda row: 1 if (row[0] == 'yes') | (row[1] == 'yes') | (row[2] == 'yes') | (row[3] == 'yes') else 0, axis=1)  
  
Out[3]:  
# Load the data from student-mat.csv  
tokyo = pd.read_csv('student-mat.csv', sep=',')  
tokyo['mat'] = tokyo['mat'].apply(lambda row: 1 if (row[0] == 'G') | (row[1] == 'G') | (row[2] == 'G') | (row[3] == 'G') else 0, axis=1)  
tokyo['drugs'] = tokyo['drugs'].apply(lambda row: 1 if (row[0] == 'yes') | (row[1] == 'yes') | (row[2] == 'yes') | (row[3] == 'yes') else 0, axis=1)  
tokyo['absent'] = tokyo['absent'].apply(lambda row: 1 if (row[0] == 'yes') | (row[1] == 'yes') | (row[2] == 'yes') | (row[3] == 'yes') else 0, axis=1)  
  
[4 rows x 31 columns]  
In [4]:
```

Gambar 2.55 Nomor 2

2.3.2.3 Nomor 3

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

```
# In [1]:
# Import library
import pandas as pd
# Read dataset
tokyo = pd.read_csv('tokyo.csv')
# Select columns
columns=['sex', 'school', 'address', 'families', 'parents', 'age', 'relatives',
          'nursery', 'higher', 'internet', 'romantic']
# Head
tokyo.head()

# Out[1]:
In [1]: tokyo = pd.read_csv('tokyo.csv', columns=['sex', 'school', 'address', 'families', 'parents', 'age', 'relatives',
          'nursery', 'higher', 'internet', 'romantic'])
Out[1]:
   sex school address families parents age relatives nursery higher internet romantic
0    M   Pedu   Pudu ... Internet_no romanic_no
1    L    Lekir    L ... Internet_no romanic_no
2    M    Lekir    L ... Internet_no romanic_no
3    M    Lekir    L ... Internet_no romanic_no
4    M    Lekir    L ... Internet_no romanic_no
... ... ... ... ... ...
[5 rows x 17 columns]
```

Gambar 2.56 Nomor 3

2.3.2.4 Nomor 4

```
# In [4]:
# shuffle rows
tokyo = tokyo.sample(frac=1) #Mengambil data sample dari tokyo
# split training and testing data
tokyo_train = tokyo[:500] #Membagi data untuk training
tokyo_test = tokyo[500:] #Membagi data untuk test
# drop pass column
tokyo_train_att = tokyo_train.drop(['pass'], axis=1) #Meghapus data yang telah pass dan memasukkannya
tokyo_train_pass = tokyo_train['pass'] #Mengambil data yang pass saja
# drop pass column
tokyo_test_att = tokyo_test.drop(['pass'], axis=1) #Meghapus data yang telah pass dan memasukkannya
tokyo_test_pass = tokyo_test['pass'] #Mengambil data yang pass saja
# drop pass column
tokyo_att = tokyo.drop(['pass'], axis=1) #Meghapus data yang telah pass dan memasukkannya
tokyo_pass = tokyo['pass'] #Mengambil data yang pass saja
# number of passing students in whole dataset:
import numpy as np #Mengimport library numpy sebagai np
print("Passing: %d out of %d (%.2f%%)" % (np.sum(tokyo_pass), len(tokyo_pass), 100*float(np.sum(tokyo_pass)) / len(tokyo_pass)))
#Menampilkan data
```

```
# In [2]:
# Import library
import pandas as pd
# Read dataset
tokyo = pd.read_csv('tokyo.csv')
# Select columns
columns=['sex', 'school', 'address', 'families', 'parents', 'age', 'relatives',
          'nursery', 'higher', 'internet', 'romantic']
# Head
tokyo.head()

# Out[2]:
In [2]: tokyo = pd.read_csv('tokyo.csv')
Out[2]:
   sex school address families parents age relatives nursery higher internet romantic
0    M   Pedu   Pudu ... Internet_no romanic_no
1    L    Lekir    L ... Internet_no romanic_no
2    M    Lekir    L ... Internet_no romanic_no
3    M    Lekir    L ... Internet_no romanic_no
4    M    Lekir    L ... Internet_no romanic_no
... ... ... ... ... ...
[5 rows x 17 columns]
```

Gambar 2.57 Nomor 4

2.3.2.5 Nomor 5

```
# In [5]:
# fit a decision tree
from sklearn import tree #import Decision tree dari library
sklearn
```

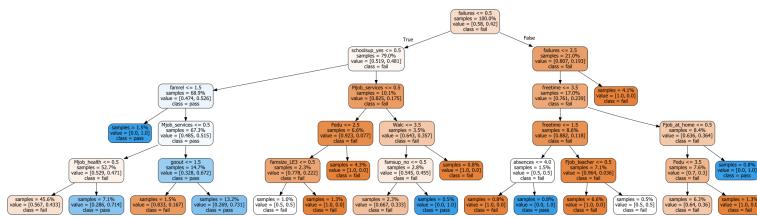
```
4 kyoto = tree.DecisionTreeClassifier(criterion="entropy",
      max_depth=5) #Membuat decision tree dengan maximal depthnya 5
5 kyoto = kyoto.fit(tokyo_train_att, tokyo_train_pass) #Memasukkan
          data yang akan dijadikan decision treenya
```

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

Gambar 2.58 Nomor 5

2.3.2.6 Nomor 6

```
1 # In [6]:  
2 # visualize tree  
3 import graphviz #Mengimport Library Graphviz untuk  
     memvisualisasikan decision tree  
4 dot_data = tree.export_graphviz(kyoto, out_file=None, label="all"  
     , impurity=False, proportion=True,  
     feature_names=list(  
         tokyo_train_att), class_names=["fail", "pass"],  
     filled=True, rounded=True) #  
     Mendefinisikan dot_data yang isikan akan berisikan data yang  
     akan dijadikan gambar  
7 graph = graphviz.Source(dot_data) #Memasukkan data tadi menjadi  
     sebuah graph  
8 graph #Menampilkan graph menggunakan graphviz
```



Gambar 2.59 Nomor 6

2.3.2.7 Nomor 7

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

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

Gambar 2.60 Nomor 7

2.3.2.8 Nomor 8

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

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

Gambar 2.61 Nomor 8

2.3.2.9 Nomor 9

```
1 # In [9]:
2 from sklearn.model_selection import cross_val_score #Mengimport
  fungsi cross_val_score dari library sklearn
3 nagoya = cross_val_score(kyoto, tokyo_att, tokyo_pass, cv=5) #
  Mendefinisikan nagoya yang isinya pembagian data menjadi 5
4 # show average score and +/- two standard deviations away (
  covering 95% of scores)
5 print("Accuracy: %.2f (+/- %.2f)" % (nagoya.mean(), nagoya.std()
  () * 2)) #Menampilkan data nilai dan +/- dari dua standar
  deviasi
```

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

Gambar 2.62 Nomor 9

2.3.2.10 Nomor 10

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

```
In [70]: for max_depth in range(1, 20):
...:     kyoto = tree.DecisionTreeClassifier(criterion="entropy", max_depth=max_depth)
...:     ...:
...:     nagoya = cross_val_score(kyoto, tokyo_att, tokyo_pass, cv=5)
...:     ...:
...:     print("Max depth: %d, Accuracy: %0.2f (+/- %0.2f)" % (max_depth, nagoya.mean(), nagoya.std() * 2))
Max depth: 1, Accuracy: 0.56 (+/- 0.01)
Max depth: 2, Accuracy: 0.56 (+/- 0.08)
Max depth: 3, Accuracy: 0.56 (+/- 0.12)
Max depth: 4, Accuracy: 0.57 (+/- 0.08)
Max depth: 5, Accuracy: 0.55 (+/- 0.09)
Max depth: 6, Accuracy: 0.57 (+/- 0.18)
Max depth: 7, Accuracy: 0.57 (+/- 0.15)
Max depth: 8, Accuracy: 0.54 (+/- 0.18)
Max depth: 9, Accuracy: 0.54 (+/- 0.11)
Max depth: 10, Accuracy: 0.58 (+/- 0.11)
Max depth: 11, Accuracy: 0.56 (+/- 0.10)
Max depth: 12, Accuracy: 0.60 (+/- 0.08)
Max depth: 13, Accuracy: 0.59 (+/- 0.05)
Max depth: 14, Accuracy: 0.58 (+/- 0.05)
Max depth: 15, Accuracy: 0.57 (+/- 0.07)
Max depth: 16, Accuracy: 0.58 (+/- 0.06)
Max depth: 17, Accuracy: 0.60 (+/- 0.11)
Max depth: 18, Accuracy: 0.58 (+/- 0.08)
Max depth: 19, Accuracy: 0.58 (+/- 0.06)
```

Gambar 2.63 Nomor 10

2.3.2.11 Nomor 11

```
1 # In[11]:
2 depth_acc = np.empty((19,3), float) #Membuat array baru
3 i = 0 #Membuat variable berisikan 0
4 for max_depth in range(1, 20): #Perulangan untuk memasukkan data
5     kyoto = tree.DecisionTreeClassifier(criterion="entropy",
6     max_depth=max_depth)#Membuat decision Tree
7     nagoya = cross_val_score(kyoto, tokyo_att, tokyo_pass, cv=5)
8     #Mendefinisikan nagoya yang isinya pembagian data menjadi 5
9     depth_acc[i,0] = max_depth #Memasukkan data max_depth ke
10    array depth_acc
11    depth_acc[i,1] = nagoya.mean() #Memasukkan data rata-rata
12    dari nagoya ke array depth_acc
13    depth_acc[i,2] = nagoya.std() * 2 #Memasukkan data akar 2
14    dari nagoya ke array depth_acc
15    i += 1
16
17 depth_acc
```

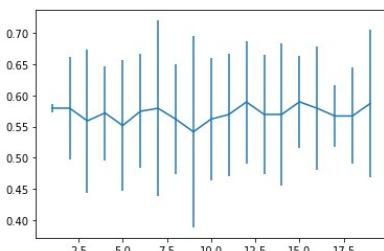
```
In [71]: depth_acc = np.empty((19,3), float)
...: i = 0
...: for max_depth in range(1, 20):
...:     kyoto = tree.DecisionTreeClassifier(criterion="entropy", max_depth=max_depth)
...:     nagoya = cross_val_score(kyoto, tokyo_att, tokyo_pass, cv=5)
...:     depth_acc[i,0] = max_depth
...:     depth_acc[i,1] = nagoya.mean()
...:     depth_acc[i,2] = nagoya.std() * 2
...:     i += 1
...:
...:
...: depth_acc
Out[71]:
array([[1.00000000e+00, 5.79751704e-01, 6.30768599e-03],
       [2.00000000e+00, 5.79654333e-01, 8.25005697e-02],
       [3.00000000e+00, 5.54241318e-01, 1.21808510e-01],
       [4.00000000e+00, 5.71964460e-01, 8.72318860e-02],
       [5.00000000e+00, 5.46678027e-01, 9.66616528e-02],
       [6.00000000e+00, 5.69561819e-01, 1.08113451e-01],
       [7.00000000e+00, 5.67030996e-01, 1.40406275e-01],
       [8.00000000e+00, 5.47030996e-01, 1.22447311e-01],
       [9.00000000e+00, 5.51966082e-01, 6.68884125e-02],
       [1.00000000e+01, 5.87314184e-01, 7.12478298e-02],
       [1.10000000e+01, 5.77124310e-01, 7.61119153e-02],
       [1.20000000e+01, 5.89719247e-01, 4.34452239e-02],
       [1.30000000e+01, 5.97569783e-01, 3.87484760e-02],
       [1.40000000e+01, 5.77026939e-01, 7.68881184e-02],
       [1.50000000e+01, 5.97347452e-01, 5.69404888e-02],
       [1.60000000e+01, 5.74720058e-01, 1.29000478e-01],
       [1.70000000e+01, 5.82282538e-01, 5.71762018e-02],
       [1.80000000e+01, 5.74720870e-01, 3.56163418e-02],
       [1.90000000e+01, 5.77250893e-01, 8.75345835e-02]])
```

Gambar 2.64 Nomor 11

2.3.2.12 Nomor 12

```
1 # In [12]:
2 import matplotlib.pyplot as plt #Menimport fungsi pyplot dari
   library matplotlib sebagai plt
3 fig, ax = plt.subplots() #Membuat plot baru
4 ax.errorbar(depth_acc[:,0], depth_acc[:,1], yerr=depth_acc[:,2])
   #Mengisikan data plot
5 plt.show() #Menampilkan plot
```

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



Gambar 2.65 Nomor 12

2.3.3 Penanganan Error

2.3.3.1 Error

1. ModuleNotFoundError

Traceback (most recent call last):

```
File "<ipython-input-25-af85b140ad99>", line 1, in <module>
    import graphviz
```

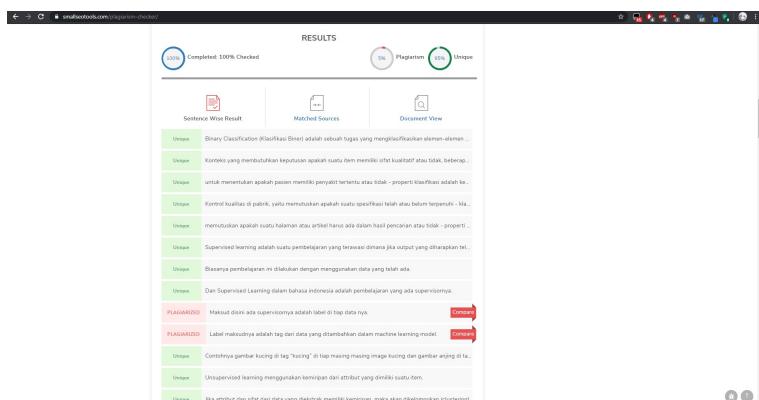
```
ModuleNotFoundError: No module named 'graphviz'
```

Gambar 2.66 ModuleNotFoundError

2.3.3.2 Solusi

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

2.3.4 Bukti Tidak Plagiat



Gambar 2.67 Bukti Tidak Plagiat

2.3.5 Link Youtube:

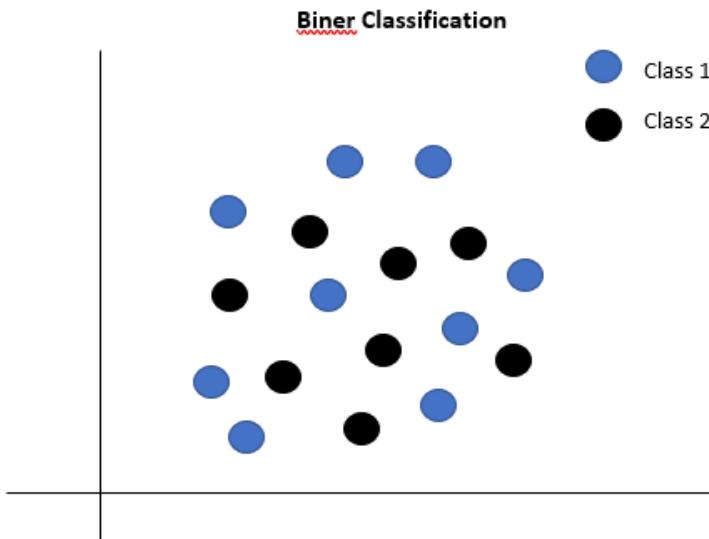
<https://youtu.be/lwhaqVixHu8>

2.4 1174087 - Ilham Muhammad Ariq

Chapter 2 - Membangun model prediksi

2.4.1 Teori

2.4.1.1 *Jelaskan Apa Itu Binary Classification dilengkapi ilustrasi gambar sendiri.*
Klasifikasi biner bertujuan untuk mengklasifikasikan elemen-elemen dari himpunan yang diberikan ke dalam dua kelompok (memprediksi kelompok mana yang masing-masing dimiliki) berdasarkan aturan klasifikasi.

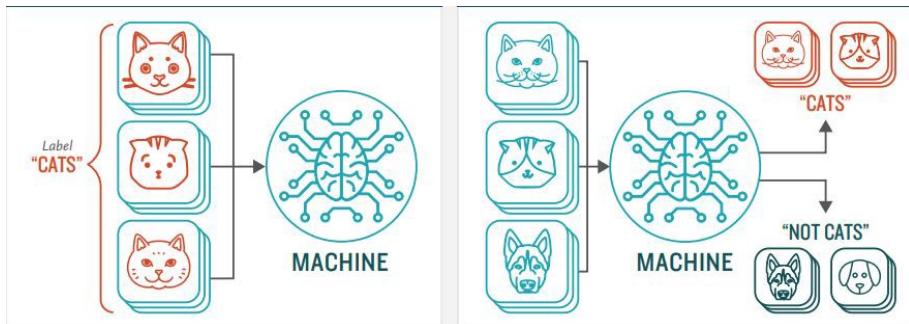


Gambar 2.68 Binary Classification

2.4.1.2 *Jelaskan Apa itu supervised learning , unsupervised learning dan clustering dengan ilustrasi gambar sendiri.*

1. supervised learning

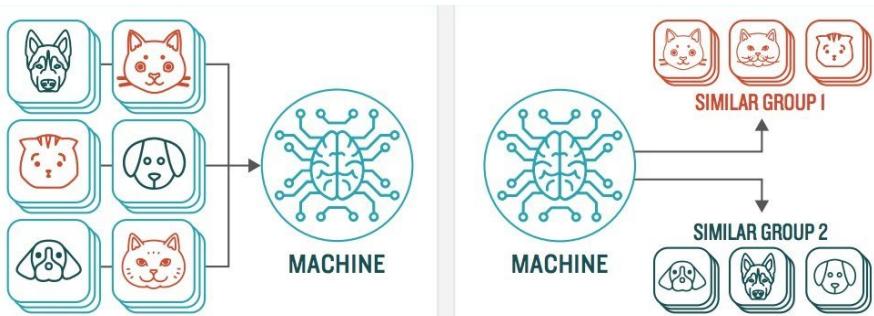
Supervised learning adalah suatu pembelajaran yang terawasi dimana output yang diharapkan telah diketahui sebelumnya. Biasanya pembelajaran ini dilakukan dengan menggunakan data yang telah ada. Dan Supervised Learning dalam bahasa indonesia adalah pembelajaran yang ada supervisornya. Maksudnya ada supervisornya adalah label di tiap data nya.



Gambar 2.69 Supervised Learning

2. unsupervised learning

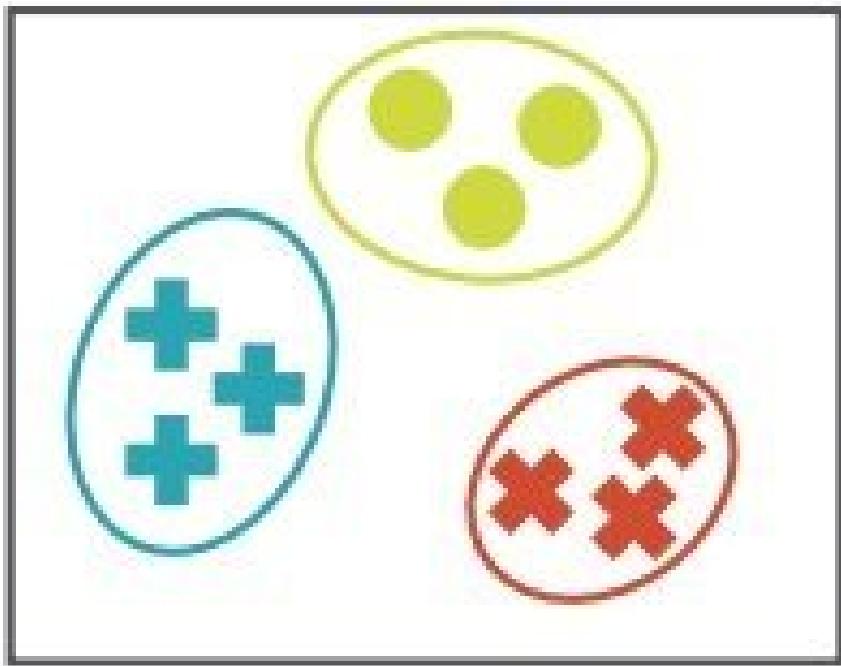
Unsupervised learning berbeda dengan supervised learning. Unsupervised learning memiliki keunggulan dari supervised learning. Supervised learning memiliki label sebagai dasar prediksi untuk membuat clasification dan regression algorithm yang memungkinkan. Tetapi dalam realitanya, data real banyak yang tidak memiliki label. Jadi unsupervised learning menggunakan ke samaan dari attribut attribut yang dimiliki untuk mencari kemiripan, dan kemudian dikelompok kelompokan (clustering). Sehingga hal ini akan menimbulkan kelompok kelompok (cluster).



Gambar 2.70 Unsupervised Learning

3. Clustering

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



Gambar 2.71 gambaran clustering

2.4.1.3 *Jelaskan apa itu evaluasi dan akurasi dan disertai ilustrasi contoh dengan gambar sendiri*

Evaluasi adalah tentang bagaimana kita dapat mengevaluasi seberapa baik model bekerja dengan mengukur akurasinya. Dan akurasi akan didefinisikan sebagai persentase kasus yang diklasifikasikan dengan benar. Kita dapat menganalisis kesalahan yang dibuat oleh model, atau tingkat kebingungannya, menggunakan matriks kebingungan.

Hasil Prediksi "Mangga"		Hasil Prediksi "Jambu"	
True "Mangga"	24	Hasil Prediksi "Jambu"	1
True "Jambu"	3		22

Gambar 2.72 Evaluasi dan Akurasi

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

Confusion Matrix suatu metode yang biasanya digunakan untuk melakukan perhitungan akurasi pada konsep data mining atau Sistem Pendukung Keputusan. Pada pengukuran kinerja menggunakan confusion matrix, terdapat 4

(empat) istilah sebagai representasi hasil proses klasifikasi. Keempat istilah tersebut adalah True Positive (TP), True Negative (TN), False Positive (FP) dan False Negative (FN). Nilai True Negative (TN) merupakan jumlah data negatif yang terdeteksi dengan benar, sedangkan False Positive (FP) merupakan data negatif namun terdeteksi sebagai data positif. Sementara itu, True Positive (TP) merupakan data positif yang terdeteksi benar. False Negative (FN) merupakan kebalikan dari True Positive, sehingga data positif, namun terdeteksi sebagai data negatif.

		True Values	
		True	False
Prediction	True	TP Correct result	FP Unexpected result
	False	FN Missing result	TN Correct absence of result

Gambar 2.73 Confusion Matrix

2.4.1.5 Jelaskan bagaimana K-fold cross validation bekerja dengan gambar ilustrasi contoh buatan sendiri.

Cara Kerja k-fold cross validation:

- Total instance dibagi menjadi N bagian.
- Fold yang pertama adalah bagian pertama menjadi data uji (testing data) dan sisanya menjadi training data.
- Lalu hitung akurasi berdasarkan porsi data tersebut dengan menggunakan persamaan.
- Fold yang kedua adalah bagian kedua, yang menjadi data uji(testing data)dan sisanya training data.
- Kemudian hitung akurasi berdasarkan porsi data tersebut.
- Dan seterusnya hingga habis mencapai fold ke-K.
- Terakhir hitung rata-rata akurasi K buah.

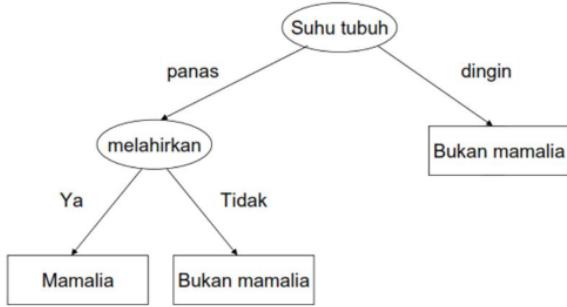


Gambar 2.74 K-Fold Validation

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

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

Contoh Pohon Keputusan : Klasifikasi Vertebrata



Gambar 2.75 Decision Tree Hewan Vertebrata

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

1. information Gain

Information Gain merupakan total data yang didapat dari data - data acak yang data tersebut akan digunakan untuk analisis data lainnya.

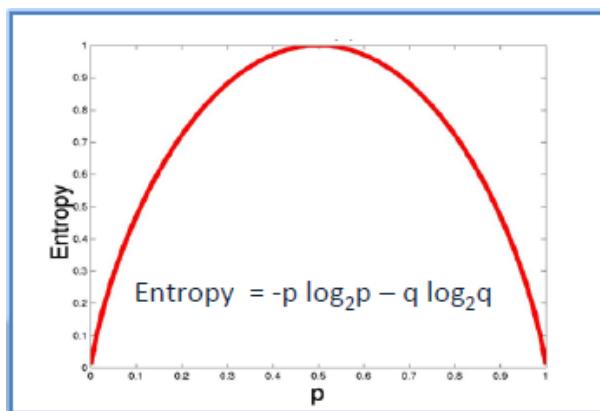
Information Gain ini digunakan pada decision tree sebagai label setiap aksi - aksi yang perlu dinilai validitasnya.



Gambar 2.76 information gain

2. Entropi

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



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

Gambar 2.77 Entropi

2.4.2 Praktek

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

```
1
2 # In [0]
```

2.4.2.1 Nomor 1

```
1 # In [1]
2 import pandas as pd #import library pandas dan sebagai pd
3 palembang = pd.read_csv('E:/Kecerdasan Buatan/chapter2/KB3C/src
   /1174087/2/dataset/student-mat.csv', sep=';') #Membuat
   variable palembang yang isinya memanggil fungsi membaca file
   csv

In [7]: import pandas as pd #import library pandas dan sebagai pd
...: palembang = pd.read_csv('E:/Kecerdasan Buatan/chapter2/KB3C/src/1174087/2/
...: dataset/student-mat.csv', sep=';') #Membuat variable palembang yang isinya memanggil
...: fungsi membaca file csv
...: ...: len(palembang) #Menghitung jumlah data yang ada pada csv yang sudah dibaca
Out[7]: 395
```

Gambar 2.78 Nomor 1

2.4.2.2 Nomor 2

```
1
2 # In [2]
3 # generate binary label (pass/fail) based on G1+G2+G3 ( test
   grades, each 0–20 pts); threshold for passing is sum>=30
4 palembang['pass'] = palembang.apply(lambda row: 1 if (row['G1']+
   row['G2']+row['G3']) >= 35 else 0, axis=1) #Membuat label
   binary (pass/fail) berdasarkan G1+G2+G3 (testgrade, semuanya
   0–20 point); Batas untuk pass adalah sum>=30
5 palembang = palembang.drop(['G1', 'G2', 'G3'], axis=1) #
   Meghilangkan data G1 G2 dan G3
```

	school	sex	age	address	famsize	...	Dalc	Walc	health	absences	pass
0	GP	F	18	U	GT3	...	1	1	3	6	0
1	GP	F	17	U	GT3	...	1	1	3	4	0
2	GP	F	15	U	LE3	...	2	3	3	10	0
3	GP	F	15	U	GT3	...	1	1	5	2	1
4	GP	F	16	U	GT3	...	1	2	5	4	0

[5 rows x 31 columns]

Gambar 2.79 Nomor 2

2.4.2.3 Nomor 3

```

1
2 # In [3]:
3 # use one-hot encoding on categorical columns
4 palembang = pd.get_dummies(palembang, columns=['sex', 'school', 'address', 'famsize', 'Pstatus', 'Mjob', 'Fjob', 'reason', 'guardian', 'schoolsup', 'famsup', 'paid', 'activities', 'nursery', 'higher', 'internet', 'romantic'])
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
```

	age	Medu	Fedu	...	internet_yes	romantic_no	romantic_yes
0	18	4	4	...	0	1	0
1	17	1	1	...	1	1	0
2	15	1	1	...	1	1	0
3	15	4	2	...	1	0	1
4	16	3	3	...	0	1	0

[5 rows x 57 columns]

Gambar 2.80 Nomor 3

2.4.2.4 Nomor 4

```

1
2 # In [4]:
3 # shuffle rows
4 palembang = palembang.sample(frac=1) #Mengambil data sample dari
5 # palembang
6 # split training and testing data
7 palembang_train = palembang[:500] #Membagi data untuk training
8 palembang_test = palembang[500:] #Membagi data untuk test
9
10 palembang_train_att = palembang_train.drop(['pass'], axis=1) #
11 # Meghapus data yang telah pass dan memasukkannya
12 palembang_train_pass = palembang_train['pass'] #Mengambil data
13 # yang pass saja
14
15 palembang_test_att = palembang_test.drop(['pass'], axis=1) #
16 # Meghapus data yang telah pass dan memasukkannya
17 palembang_test_pass = palembang_test['pass'] #Mengambil data yang
18 # pass saja
19
20 # number of passing students in whole dataset:
21 import numpy as np #Mengimport library numpy sebagai np
22
```

```

...: import numpy as np #Mengimport Library numpy sebagai np
...: print("Passing: %d out of %d (%.2f%%)" % (np.sum(palembang_pass), len(palembang_pass),
100*float(np.sum(palembang_pass)) / len(palembang_pass))) #lenampulkan data
Passing: 166 out of 395 (42.03%)

```

Gambar 2.81 Nomor 4

2.4.2.5 Nomor 5

```

1 # In [5]:
2 # fit a decision tree
3 from sklearn import tree #import Decision tree dari library
4         sklearn
5 muaraenim = tree.DecisionTreeClassifier(criterion="entropy",
6         max_depth=5) #Membuat decision tree dengan maximal depthnya 5

```

In [16]: `from sklearn import tree #import Decision tree dari library sklearn`
`.... muaraenim = tree.DecisionTreeClassifier(criterion="entropy", max_depth=5) #Membuat decision tree dengan maximal depthnya 5`
`.... muaraenim = muaraenim.fit(palembang_train_att, palembang_train_pass) #masukkan data yang akan dijadikan decision treenya`

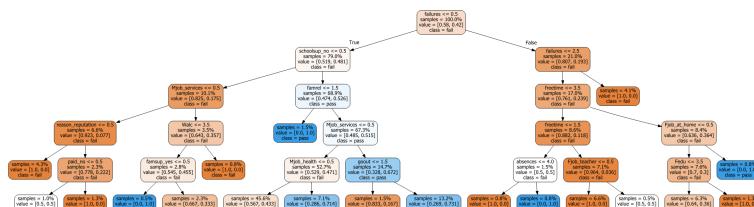
Gambar 2.82 Nomor 5

2.4.2.6 Nomor 6

```

1 # In [6]:
2 # visualize tree
3 import graphviz #Mengimport Library Graphviz untuk memvisualisasikan decision tree
4 dot_data = tree.export_graphviz(muaraenim, out_file=None, label="all",
5         impurity=False, proportion=True,
6         feature_names=list(
7             palembang_train_att), class_names=["fail", "pass"],
8         filled=True, rounded=True) #
9 Mendefinisikan dot_data yang isikan akan berisikan data yang akan dijadikan gambar
10 graph = graphviz.Source(dot_data) #Masukkan data tadi menjadi sebuah graph
11 graph #Menampilkan graph menggunakan graphviz

```



Gambar 2.83 Nomor 6

2.4.2.7 Nomor 7

```

1 # save tree
2 tree.export_graphviz(muaraenim, out_file="student-performance.dot",
3     , label="all", impurity=False, proportion=True,
4         feature_names=list(palembang_train_att),
5         class_names=["fail", "pass"],
6             filled=True, rounded=True) #Digunakan untuk
7     mengexport graph tree tadi yang telah kita buat

```

```

In [13]: tree.export_graphviz(muaraenim, out_file="student-performance.dot", label="all",
imprtity=False, proportion=True,
...: feature_names=list(palembang_train_att), class_names=["fail", "pass"],
...: filled=True, rounded=True) #Digunakan untuk mengexport graph tree tadi
yang telah kita buat

```

Gambar 2.84 Nomor 7

2.4.2.8 Nomor 8

```

1 muaraenim.score(palembang_test_att, palembang_test_pass) # Menghitung prediksi nilai yang akan datang dimasa depan

```

```
ValueError: Found array with 0 sample(s) (shape=(0, 56)) while a minimum of 1 is required.
```

Gambar 2.85 Nomor 8

Error dikarenakan tidak ada sample data ditemukan

2.4.2.9 Nomor 9

```

1 from sklearn.model_selection import cross_val_score #Mengimport
    fungsi cross_val_score dari library sklearn
2 prabumulih = cross_val_score(muaraenim, palembang_att,
    palembang_pass, cv=5) #Mendefinisikan prabumulih yang isinya
        pembagian data menjadi 5
3 # show average score and +/- two standard deviations away (
    covering 95% of scores)
4 print("Accuracy: %.2f (+/- %.2f)" % (prabumulih.mean(),
    prabumulih.std() * 2)) #Menampilkan data nilai dan +/- dari
        dua standar deviasi

```

```

In [16]: from sklearn.model_selection import cross_val_score #Mengimport fungsi cross_val_score dari
library sklearn
...: prabumulih = cross_val_score(muaraenim, palembang_att, palembang_pass, cv=5)
#Mendefinisikan prabumulih yang isinya pembagian data menjadi 5
...: # show average score and +/- two standard deviations away (covering 95% of scores)
...: print("Accuracy: %.2f (+/- %.2f)" % (prabumulih.mean(), prabumulih.std() * 2))
#Menampilkan data nilai dan +/- dari dua standar deviasi
Accuracy: 0.57 (+/- 0.07)

```

Gambar 2.86 Nomor 9

2.4.2.10 Nomor 10

```

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

```

```

Max depth: 9, Accuracy: 0.59 (+/- 0.09)
Max depth: 10, Accuracy: 0.57 (+/- 0.09)
Max depth: 11, Accuracy: 0.59 (+/- 0.09)
Max depth: 12, Accuracy: 0.60 (+/- 0.08)
Max depth: 13, Accuracy: 0.58 (+/- 0.08)
Max depth: 14, Accuracy: 0.60 (+/- 0.08)
Max depth: 15, Accuracy: 0.59 (+/- 0.07)
Max depth: 16, Accuracy: 0.62 (+/- 0.06)
Max depth: 17, Accuracy: 0.63 (+/- 0.07)
Max depth: 18, Accuracy: 0.61 (+/- 0.06)
Max depth: 19, Accuracy: 0.60 (+/- 0.06)

```

Gambar 2.87 Nomor 10

2.4.2.11 Nomor 11

```

1 depth_acc = np.empty((19,3), float) #Membuat array baru
2 i = 0 #Membuat variable berisikan 0
3 for max_depth in range(1, 20): #Perulangan untuk memasukkan data
4     muaraenim = tree.DecisionTreeClassifier(criterion="entropy",
5         max_depth=max_depth) #Membuat decision Tree
6     prabumulih = cross_val_score(muaraenim, palembang_att,
7         palembang_pass, cv=5) #Mendefinisikan prabumulih yang isinya
8         pembagian data menjadi 5
9     depth_acc[i,0] = max_depth #Memasukkan data max_depth ke
10    array depth_acc
11    depth_acc[i,1] = prabumulih.mean() #Memasukkan data rata-rata
12    dari prabumulih ke array depth_acc
13    depth_acc[i,2] = prabumulih.std() * 2 #Memasukkan data akar 2
14    dari prabumulih ke array depth_acc
15    i += 1
16
17 depth_acc

```

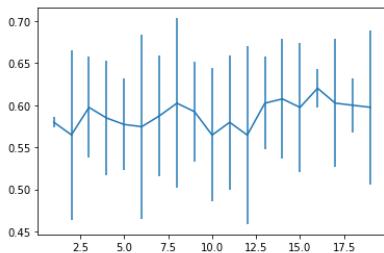
```
[9.00000000e+00, 5.92346641e-01, 5.98837850e-02],
[1.00000000e+01, 5.64466894e-01, 7.92362856e-02],
[1.10000000e+01, 5.79690847e-01, 7.97871700e-02],
[1.20000000e+01, 5.64531808e-01, 1.05870335e-01],
[1.30000000e+01, 6.02410743e-01, 5.51545214e-02],
[1.40000000e+01, 6.07506491e-01, 7.13718594e-02],
[1.50000000e+01, 5.97413178e-01, 7.64863591e-02],
[1.60000000e+01, 6.20229633e-01, 2.33434488e-02],
[1.70000000e+01, 6.02508114e-01, 7.63071783e-02],
[1.80000000e+01, 5.99943200e-01, 3.24634690e-02],
[1.90000000e+01, 5.97541383e-01, 9.13409892e-02]])
```

Gambar 2.88 Nomor 11

2.4.2.12 Nomor 12

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

```
In [22]: import matplotlib.pyplot as plt #Menimpor fungsi pyplot dari library matplotlib sebagai plt
...: fig, lampung = plt.subplots() #Membuat plot baru
...: lampung.errorbar(depth_acc[:,0], depth_acc[:,1], yerr=depth_acc[:,2]) #Mengisikan data plot
...: plt.show() #Menampilkan plot
```



Gambar 2.89 Nomor 12

2.4.3 Penanganan Error

2.4.3.1 Error

1. ModuleNotFoundError

```
File "<ipython-input-17-e9072bd6b1ea>", line 1, in <module>
    import graphviz #Mengimport Library Grapthviz untuk memvisualisasikan decision tree
ModuleNotFoundError: No module named 'graphviz'
```

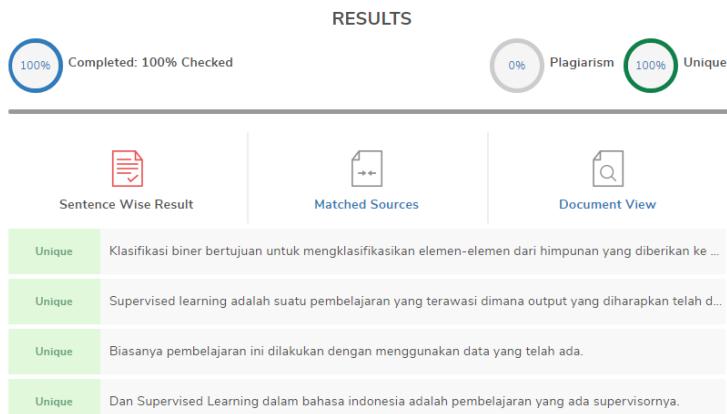
Gambar 2.90 ModuleNotFoundError

2.4.3.2 Solusi

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

```
1 conda install graphviz
```

2.4.4 Bukti Tidak Plagiat



Gambar 2.91 Bukti Tidak Plagiat

2.4.5 Link Youtube:

<https://www.youtube.com/watch?v=014uRF2MkH4>

2.5 1174054 - Aulyardha Anindita

2.5.1 Teori

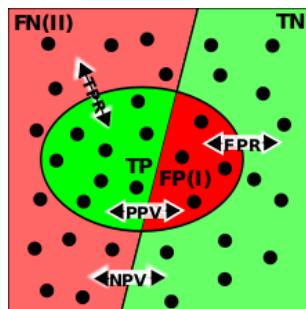
1. Binary Classification

Binary Classification adalah suatu tugas mengklafisikan himpunan yang

didalamnya terdapat elemen-elemen yang dimasukkan ke dalam kelompok berdasarkan aturan klasifikasi. Beberapa karakteristik tersebut contohnya tes medis digunakan untuk mengetahui suatu pasien memiliki penyakit tertentu atau tidak. dan properti klasifikasi tersebut adalah keberadaan penyakit dari pasien.

Binary Classification digunakan untuk tujuan praktis dalam banyak masalah klasifikasi biner, dan kedua kelompok tersebut tidak simetris daripada akurasi secara keseluruhan, proporsi relatif dari berbagai macam kesalahan yang menarik. Contohnya, dalam pengujian medis tadi, false positif maksudnya mendeteksi penyakit ketika ada sedangkan untuk false negatif artinya tidak mendeteksi penyakit ketika ada.

Ada banyak metrik yang bisa digunakan dalam mengukur kinerja klasifikasi dan prediksi. misalnya dapat dilihat pada gambar berikut:



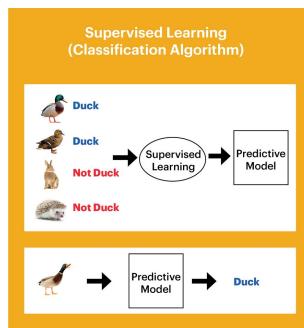
Gambar 2.92 Binary Classification

Bagian kiri dan kanan masing-masing memiliki instance yang sebenarnya ada dan tidak memiliki kondisi. Sedangkan bentuk oval tersebut berisi instance yang diklasifikasikan atau diprediksi sebagai positif atau negatif.

2. Supervised Learning, Unsupervised Learning, dan Clustering

▪ Supervised Learning

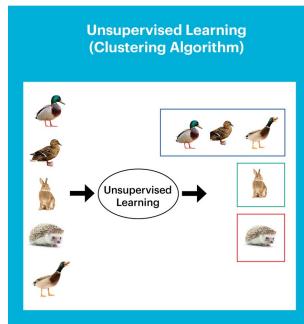
Dalam Supervised Learning, suatu program komputer diberikan dataset pelatihan yang kemudian diberi label dengan nilai output yang sesuai, dan fungsi tersebut akan ditentukan berdasarkan pada dataset. Fungsi atau algoritma tersebut kemudian akan digunakan untuk mengklasifikasikan data baru untuk memprediksi nilai-nilai output yang sesuai dengan asumsi bahwa data baru sesuai dengan aturan dan fungsi yang digunakan. Berikut adalah contoh supervised learning:



Gambar 2.93 Supervised Learning

- **Unsupervised Learning**

Dalam Unsupervised Learning, dataset pelatihan tidak memiliki label nilai output yang sesuai, karena tidak ada jawaban benar untuk dipelajari, tujuan algoritma ini adalah untuk mengungkap pola-pola menarik yang dapat ditemukan dalam data, dan data baru akan membantu untuk mengonfirmasi atau membantalkan pola-pola yang ditemukannya. Berikut adalah contoh Unsupervised Learning:

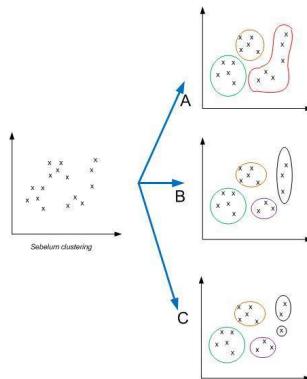


Gambar 2.94 Unsupervised Learning

- **Clustering**

Clustering adalah suatu teknik yang masuk kedalam kelompok Unsupervised Learning yang merupakan teknik dimana mesin akan bekerja atau belajar sendiri tanpa diajari bagaimana cara memecahkan masalahnya. Contohnya, Kita memiliki sebuah data, yaitu data pelanggan yang berisi jenis kelamin, besarnya penghasilan dan besarnya pembelian produk. Maka dengan algoritma Clustering kita dapat mengetahui pelanggan kita akan dikelompokkan kedalam beberapa kluster dengan sendirinya. Misalnya ada pelanggan yang pelit, pelanggan yang royal dan lain sebagainya. Contohnya, Bisa kita lihat bagaimana

sebuah teknik clustering bisa mengelompokkan data ke dalam beberapa kluster.



Gambar 2.95 Clustering

3. Evaluasi dan Akurasi

Evaluasi merupakan suatu cara atau teknik dalam mengevaluasi seberapa baik model bekerja dengan mengukur akurasinya. Sedangkan akurasi adalah suatu persentase kasus yang diklasifikasikan dengan benar. Kita bisa menganalisis suatu kesalahan yang dibuat dengan model atau tingkat confusion dengan menggunakan matriks confusion. Berikut adalah contoh klasifikasi biner yang menunjukkan berapa kali model telah membuat prediksi yang benar dari objek.

	Predicted "apple"	Predicted "orange"
True "apple"	20	5
True "orange"	3	22

Gambar 2.96 Evaluasi

Dalam tabel diatas, baris True Apple dan True Orange mengacu pada suatu kasus dimana objek itu sebenarnya sebuah apel atau sebenarnya jeruk. Kolom merujuk pada prediksi yang dibuat oleh model. Kita melihat bahwa ada 20 apel yang diprediksi dengan benar, sementara ada 5 apel yang salah diidentifikasi sebagai jeruk. Sehingga, matriks confusion harus memiliki semua nol, kecuali untuk diagonal sehingga kita dapat menghitung akurasi dengan menambahkan angka secara diagonal seperti pada gambar berikut :

$$\text{Accuracy} = (20 + 22) / (20 + 5 + 3 + 22) = 84\%$$

Gambar 2.97 Akurasi

4. Cara Membuat dan Membaca Confusion Matrix

Confusion Matrix adalah suatu matrix yang memberikan informasi perbandingan hasil klasifikasi yang dilakukan pada sistem atau model dengan hasil klasifikasi sebenarnya. Confusion Matrix berbentuk tabel matriks yang menggambarkan suatu kinerja model klasifikasi dari serangkaian data uji yang nilai sebenarnya diketahui.

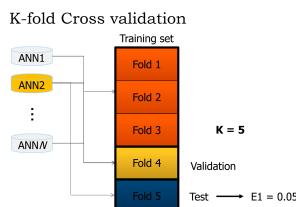
Berikut cara membuat dan membaca confusion matrix : pertama, tentukan terlebih dahulu pokok permasalahan dan atributnya. kedua, buatlah pohon keputusan dan data testingnya. ketiga, carilah nilai a,b,c dan d. Selanjutnya, cari nilai recall, precesion, accuracy serta seeror rate. Berikut contoh Confusion Matrix :

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

Gambar 2.98 Confusion Matrix

5. Cara Kerja K-fold Cross Validation

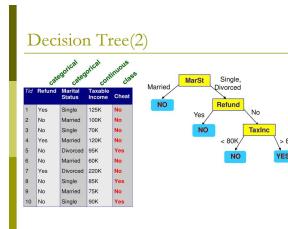
- Pertama, total instance bagi menjadi N bagian
- Fold pertama merupakan bagian peertama yang menjadi data uji atau testing data dan sisanya menjadi training data
- Kemudian, hitung akurasi dari porsi data dengan menggunakan persamaan
- Fol kedua, adalah bagian kedua dengan menjadi data uji atau testing data dan sisanya merupakan training data
- Lalu, hitung akurasi dari porsi data tersebut
- Lakukan hal tersebut, sampai habis mencapai fold ke-K
- Setelah itu, hitung rata-rata akurasi K



Gambar 2.99 K-fold Cross Validation

6. Decision Tree

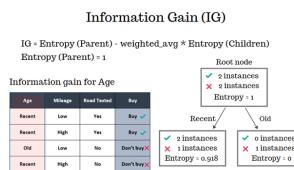
Decision Tree adalah salah satu model prediksi dengan menggunakan struktur pohon atau struktur yang berhierarki. Konsepnya adalah mengubah data menjadi decision tree dan beberapa aturan keputusan. Decision Tree memiliki manfaat yaitu membrekdown proses pengambilan keputusan yang kompleks menjadi lebih simpel sehingga pengambil keputusan akan lebih mudah dipahami.



Gambar 2.100 Decision Tree

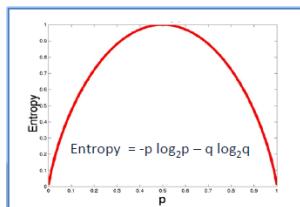
7. Information Gain dan Entropi

Information Gain adalah suatu teknik yang didasarkan pada penurunan entropi setelah dataset dibagi pada atribut. Membangun keputusan adalah menemukan atribut yang mengembalikan perolehan informasi tertinggi.



Gambar 2.101 Information Gain

Entropi adalah suatu ukuran acak dalam informasi yang sedang diproses. Semakin tinggi suatu entropi, semakin sulit menarik kesimpulan dari informasi tersebut. Decision tree dibangun dari atas kebawah dan melibatkan partisi data kedalam himpunan bagian yang berisi instance dengan nilai yang sama. Algoritma ID3 menggunakan entropi untuk menghitung suatu homogenitas sampel.



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

Gambar 2.102 Entropy

2.5.2 Praktek

1. Nomor 1

```
1 # load dataset (menggunakan student-mat.csv)
2 import pandas as pd # mengimport library pandas sebagai pd
3 makassar = pd.read_csv('D:/Mata Kuliah/Tingkat 3/Semester 6/
4 Kecerdasan Buatan/Chapter 2/dataset/student-mat.csv', sep=
5 ';' ) #variabel makassar berfungsi untuk membaca atau read
6 file student-mat.csv
7 len(makassar) #mengetahui jumlah baris pada data yang
8 dipanggil
```

Hasilnya :

```
In [44]: import pandas as pd # mengimport library pandas sebagai pd  
...: makanan = pd.read_csv('D:/Data Kuliah/Fingkat 5/Semester 6/Kecerdasan  
Buat dan Chapter 2/dataset/student-mat.csv', sep=',') # variabel makanan berfungsi  
untuk menampung atau read file student-mat.csv  
...:  
len(makanan) # mengetahui jumlah baris pada data yang dipanggil
```

Gambar 2.103 Hasil Nomor 1

2. Nomor 2

```

1 # generate binary label (pass/fail) based on G1+G2+G3 (test
  grades, each 0-20 pts); threshold for passing is sum>=30
2 makassar[ 'pass' ] = makassar.apply(lambda row: 1 if (row['G1']
  ]+row['G2']+row[ 'G3' ]) >= 35 else 0, axis=1) #
  mendeklarasikan pass/fail nya data berdasarkan G1+G2+G3.
3 makassar = makassar.drop([ 'G1', 'G2', 'G3' ], axis=1) #untuk
  mengetahui baris G1+G2+G3 ditambahkan, dan hasilnya sama
  dengan 35 maka axisnya 1.
4 makassar.head() #memanggil variabel kucing dimana ketentuan
  head ini digunakan untuk mengembalikan baris n atas 5
  secara default dari frame atau seri data

```

Hasilnya :

```
In [45]: makassar['pass'] = makassar.apply(lambda row: 1 if (row['G1'] > row['G2']) & (row['G2'] >= 35 else 0, axis=1) #mengembalikan variabel pass=1 jika data berdasarkan ketentuan
...: makassar = makassar.drop(['G1', 'G2', 'G3'], axis=1) #untuk menghilangkan baris yang tidak diperlukan
...: makassar.head() #menggil variabel hingga dimana ketentuan head tel dipenuhi untuk mengelakkan baris n atas 5 secara default dari frame atau seri data
Out[45]:
schl sex age address famsize Pstatus Mjob Fjob reason guardian schoolsup famsup paid activities nursery higher internet romantic
0 GP F 18 U GTG ... 1 1 3 6 0
1 GP F 19 U L1S ... 1 1 3 6 0
2 GP F 15 U L1S ... 2 3 3 10 0
3 GP F 19 U GTG ... 1 2 5 2 1
4 GP F 16 U GTG ... 1 2 5 4 0
[5 rows x 31 columns]
```

Gambar 2.104 Hasil Nomor 2

3. Nomor 3

```
1 # use one-hot encoding on categorical columns
2 makassar = pd.get_dummies(makassar, columns=['sex', 'school', 'address',
3 'famsize', 'Pstatus', 'Mjob', 'Fjob',
4 'reason', 'guardian', 'schoolsup', 'famsup', 'paid', 'activities',
5 'nursery', 'higher', 'internet', 'romantic']) #variabel makassar dikonversi menjadi bentuk yang lebih baik dalam prediksi dan memanggil seluruh atribut
makassar.head() #memanggil variabel makassar dengan ketentuan head ini digunakan untuk mengembalikan baris n atas 5 secara default dari frame atau seri data
```

Hasilnya :

```
In [46]: makassar = pd.get_dummies(makassar, columns=['sex', 'school', 'address',
'famsize', 'Pstatus', 'Mjob', 'Fjob',
'reason', 'guardian', 'schoolsup',
'famsup', 'paid', 'activities',
'nursery', 'higher', 'internet',
'romantic']) #variabel makassar dikonversi menjadi bentuk yang lebih baik dalam prediksi
...: makassar.head() #memanggil variabel makassar dengan ketentuan head tel dipenuhi untuk mengelakkan baris n atas 5 secara default dari frame atau seri data
Out[46]:
age Redu Fedu ... internet_yes romantic_no romantic_yes
0 18 4 4 ... 0 1 0
1 17 3 3 ... 1 1 0
2 15 1 1 ... 1 1 0
3 19 3 3 ... 1 0 1
4 16 3 3 ... 0 1 0
[5 rows x 57 columns]
```

Gambar 2.105 Hasil Nomor 3

4. Nomor 4

```
1 # shuffle rows
2 makassar = makassar.sample(frac=1) #mengembalikan variabel makassar menjadi sampel acak dengan frac=1
# split training and testing data
4 makassar_train = makassar[:500] #membuat variabel baru makassar_train
5 makassar_test = makassar[500:] #membuat variabel baru makassar_test yang sisa dari train
6
7 makassar_train_att = makassar_train.drop(['pass'], axis=1) # membuat variabel baru dengan ketentuan dari makassar_train
8 makassar_train_pass = makassar_train['pass'] #membuat variabel baru dengan ketentuan dari makassar_train
9
10 makassar_test_att = makassar_test.drop(['pass'], axis=1) # membuat variabel baru dengan ketentuan dari makassar_test
```

```
11 makassar_test_pass = makassar_test[ 'pass' ] #membuat variabel  
12     baru dengan ketentuan dari makassar_test  
13 makassar_att = makassar.drop([ 'pass' ], axis=1) #membuat  
14     variabel makassar_att sebagai salinan dari makassar  
15 makassar_pass = makassar[ 'pass' ] #membuat variabel  
16     makassar_pass sebagai salinan dari makassar  
17  
18 # number of passing students in whole dataset:  
19 import numpy as np #mengimport module numpy sebagai np y  
20 print("Passing: %d out of %d (%.2f%%)" % (np.sum(  
21         makassar_pass), len(makassar_pass), 100*float(np.sum(  
22         makassar_pass)) / len(makassar_pass))) #untuk  
23     mengembalikan nilai passing dari pelajar dari keseluruhan  
24     dataset dengan cara print.
```

Hasilnya :

```

.... makassar_train_pass = makassar_train.drop(['pass'], axis=1) membuat variabel
.... makassar_train_pass = makassar_train['pass'] membuat variabel baru dengan ketentuan dari nilai tersebut

.... makassar_test_pass = makassar_test.drop(['pass'], axis=1) membuat variabel
.... makassar_test_pass = makassar_test['pass'] membuat variabel baru dengan ketentuan dari nilai tersebut

.... makassar_1st = makassar.drop(['pass'], axis=1) membuat variabel
.... makassar_1st = makassar['pass'] membuat variabel baru dengan ketentuan dari nilai tersebut

.... makassar_2nd = makassar.drop(['pass'], axis=1) membuat variabel
.... makassar_2nd = makassar['pass'] membuat variabel baru dengan ketentuan dari nilai tersebut

.... # number of passing students in whole dataset
.... import numpy as np
.... np.sum(makassar['pass']) # (np.sum(makassar_pass), len(makassar_pass))
.... len(makassar)*100/float(np.sum(makassar_pass)) / len(makassar_pass)) diitung persentase jumlah siswa yang lulus
```

Gambar 2.106 Hasil Nomor 4

5. Nomor 5

```

1 # fit a decision tree
2 from sklearn import tree #import tree dari library sklearn
3 bone = tree.DecisionTreeClassifier(criterion="entropy",
4         max_depth=5) #membuat variabel bone sebagai decisiontree ,
5         dengan criterion fungsi mengukur kualitas split
6 bone = bone.fit(makassar_train_att , makassar_train_pass) # 
7         training varibael bone dengan data dari variabel makassar.

```

Hasilnya :

```
In [48]: from sklearn import tree #import tree dari Library sklearn
.... bone = tree.DecisionTreeClassifier(criterion="entropy", max_depth=5)
#membuat variabel bone sebagai decisiontree, dengan criterion fungsi mengukur
#kualitas split
.... bone = bone.fit(makassar_train_att, makassar_train_pass) #training
variabel bone dengan data dari variabel makassar.
```

Gambar 2.107 Hasil Nomor 5

6. Nomor 6

```
1 # visualize tree
2 import os
3 os.environ["PATH"] += os.pathsep + 'C:/Program Files (x86)/
4 Graphviz2.38/bin/'
```

```

4 import graphviz #import library graphviz sebagai perangkat
    lunak visualisasi grafik open source
5 dot_data = tree.export_graphviz(bone, out_file=None, label="all",
        impurity=False, proportion=True,
        feature_names=list(
            makassar_train_att), class_names=["fail", "pass"],
        filled=True, rounded=True) #
    mengambil data untuk diterjemahkan ke grafik
8 graph = graphviz.Source(dot_data) #membuat variabel graph
    sebagai grafik yang di ambil dari dot_data
9 graph #memanggil graph

```

Hasilnya :



Gambar 2.108 Hasil Nomor 6

7. Nomor 7

```

1 # save tree
2 tree.export_graphviz(bone, out_file="student-performance.dot",
    , label="all", impurity=False, proportion=True,
    feature_names=list(makassar_train_att),
    class_names=["fail", "pass"],
    filled=True, rounded=True) #save tree
    sebagai export graphviz ke file student-performance.dot

```

Hasilnya :

```

In [50]: tree.export_graphviz(bone, out_file="student-performance.dot",
    label="all", impurity=False, proportion=True,
    feature_names=list(makassar_train_att),
    class_names=["fail", "pass"],
    filled=True, rounded=True) #save tree sebagai export
graphviz file student-performance.dot

```

Gambar 2.109 Hasil Nomor 7

8. Nomor 8

```

1
2 bone.score(makassar_att, makassar_pass) #score juga disebut
    prediksi dengan diberi beberapa data input baru

```

Hasilnya :

```

In [51]: bone.score(makassar_att, makassar_pass) #score juga disebut prediksi
    dengan diberi beberapa data input baru
Out[51]: 0.7863291139240596

```

Gambar 2.110 Hasil Nomor 8

9. Nomor 9

```

1  from sklearn.model_selection import cross_val_score #import
2      class cross_val_score dari sklearn
3  scores = cross_val_score(bone, makassar_att, makassar_pass,
4      cv=5) #mengevaluasi score dengan validasi silang
5  # show average score and +/- two standard deviations away (
6      covering 95% of scores)
7  print("Accuracy: %0.2f (+/- %0.2f)" % (scores.mean(), scores.
8      std() * 2)) #print akurasi

```

Hasilnya :

```

In [52]: from sklearn.model_selection import cross_val_score #import class
cross_val_score
...+1 scores = cross_val_score(bone, makassar_att, makassar_pass, cv=5)
mengevaluasi score dengan validasi silang
...+1 # show average score and +/- two standard deviations away (covering 95%
of scores)
...+1 print("Accuracy: %0.2f (+/- %0.2f)" % (scores.mean(), scores.std() * 2))
...+1
Accuracy: 0.62 (+/- 0.18)

```

Gambar 2.111 Hasil Nomor 9

10. Nomor 10

```

1  for max_depth in range(1, 20):
2      bone = tree.DecisionTreeClassifier(criterion="entropy",
3          max_depth=max_depth)
4      scores = cross_val_score(bone, makassar_att,
5          makassar_pass, cv=5)
6      print("Max depth: %d, Accuracy: %0.2f (+/- %0.2f)" % (
7          max_depth, scores.mean(), scores.std() * 2))
8
9  #Disini ini menunjukkan seberapa dalam di tree itu.
10 Semakin dalam tree , semakin banyak perpecahan yang
11 dimilikinya dan menangkap lebih banyak informasi tentang
12 data .

```

Hasilnya :

```

...+1 scores = cross_val_score(bone, makassar_att, makassar_pass, cv=5)
...+1 print("Max depth %d, Accuracy: %0.2f (+/- %0.2f)" % (max_depth,
...+1
tree.DecisionTreeClassifier(criterion="entropy", max_depth=max_depth)
Max depth: 1, Accuracy: 0.58 (+/- 0.01)
Max depth: 2, Accuracy: 0.59 (+/- 0.14)
Max depth: 3, Accuracy: 0.62 (+/- 0.07)
Max depth: 4, Accuracy: 0.59 (+/- 0.07)
Max depth: 5, Accuracy: 0.63 (+/- 0.09)
Max depth: 6, Accuracy: 0.64 (+/- 0.12)
Max depth: 7, Accuracy: 0.64 (+/- 0.12)
Max depth: 8, Accuracy: 0.64 (+/- 0.12)
Max depth: 9, Accuracy: 0.62 (+/- 0.08)
Max depth: 10, Accuracy: 0.62 (+/- 0.07)
Max depth: 11, Accuracy: 0.62 (+/- 0.07)
Max depth: 12, Accuracy: 0.62 (+/- 0.07)
Max depth: 13, Accuracy: 0.62 (+/- 0.07)
Max depth: 14, Accuracy: 0.62 (+/- 0.08)
Max depth: 15, Accuracy: 0.63 (+/- 0.08)
Max depth: 16, Accuracy: 0.63 (+/- 0.08)
Max depth: 17, Accuracy: 0.62 (+/- 0.08)
Max depth: 18, Accuracy: 0.62 (+/- 0.08)
Max depth: 19, Accuracy: 0.61 (+/- 0.08)

```

Gambar 2.112 Hasil Nomor 10

11. Nomor 11

```

1 depth_acc = np.empty((19,3), float) #Dengan 19 sebagai bentuk
2         array kosong, 3 sebagai output data-type
3 bulukumba = 0 #variabel bulukumba sebagai array 0
4 for max_depth in range(1, 20): #perulangan dengan max_depth
5     bone = tree.DecisionTreeClassifier(criterion="entropy",
6             max_depth=max_depth) #variabel ular untuk decision tree
7             dengan ketentuan entropy
8     scores = cross_val_score(bone, makassar_att,
9             makassar_pass, cv=5) #scores diambil dari data
10            cross_val_score
11     depth_acc[bulukumba,0] = max_depth #mengembalikan array
12             dengan ketentuan 0 dan max_depth
13     depth_acc[bulukumba,1] = scores.mean() #mengembalikan
14             array dengan ketentuan 1 dan scores.mean
15     depth_acc[bulukumba,2] = scores.std() * 2 #mengembalikan
16             array dengan ketentuan 2 dan scores.std, std berarti
17             menghitung standar deviasi
18     bulukumba += 1
19
20 depth_acc #Depth acc akan membuat array kosong dengan
21         mengembalikan array baru dengan bentuk dan tipe yang
22         diberikan

```

Hasilnya :

```

Out[54]: array([[1.00000000e+00, 5.79751704e-01, 6.30768599e-01],
 [1.00000000e+00, 5.46955271e-01, 6.23554871e-02],
 [1.00000000e+00, 5.92572217e-01, 7.55790870e-02],
 [1.00000000e+00, 6.15516788e-01, 9.89162865e-02],
 [1.00000000e+00, 6.15516788e-01, 9.89162865e-02],
 [1.00000000e+00, 6.39739280e-01, 1.82023515e-01],
 [1.00000000e+00, 6.85229533e-01, 5.56592246e-02],
 [1.00000000e+01, 6.85229533e-01, 5.56592246e-02],
 [1.00000000e+01, 6.85229533e-01, 5.56592246e-02],
 [1.00000000e+01, 6.35516784e-01, 4.08697001e-02],
 [1.00000000e+01, 6.17985232e-01, 7.85221184e-02],
 [1.00000000e+01, 6.36644271e-01, 6.82219467e-02],
 [1.00000000e+01, 6.25516788e-01, 6.23559880e-02],
 [1.00000000e+01, 6.25516788e-01, 6.23559880e-02],
 [1.00000000e+01, 5.97763713e-01, 7.80811645e-02],
 [1.00000000e+01, 6.257079562e-01, 1.15790540e-01]])

```

Gambar 2.113 Hasil Nomor 11

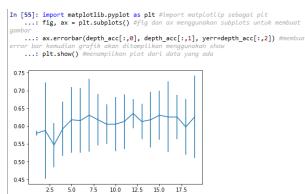
12. Nomor 12

```

1 import matplotlib.pyplot as plt #import matplotlib sebagai
2         plt
3 fig, ax = plt.subplots() #fig dan ax menggunakan subplots
4         untuk membuat gambar
5 ax.errorbar(depth_acc[:,0], depth_acc[:,1], yerr=depth_acc
6             [:,2]) #membuat error bar kemudian grafik akan ditampilkan
7             menggunakan show
8 plt.show() #menampilkan plot dari data yang ada

```

Hasilnya :

**Gambar 2.114** Hasil Nomor 12

2.5.3 Penanganan Error

1. ScreenShoot Error

```
File "cipython-input-7-f751199220fa", line 1, in <module>
    import graphviz #import library graphviz sebagai perangkat
    luncur visualisasi grafik open source
ModuleNotFoundError: No module named 'graphviz'
```

Gambar 2.115 Module Not Found Error

```
FileNotFoundException: File b'dataset/student-mat.csv' does not exist
```

Gambar 2.116 File Not Found Error

```
File "C:\ProgramData\Anaconda\lib\site-packages\graphviz
\backend.py", line 162, in run
    raise ExecutableNotFoundError(cmd)
ExecutableNotFoundError: failed to execute ['dot', '-Tsvg'], make sure
the Graphviz executables are on your system's PATH
Out[8]: <graphviz.files.Source at 0x2595e5d8788>
```

Gambar 2.117 Executable Not Found

2. Tuliskan Kode Error dan Jenis Error

- Module Not Found Error
- File Not Found Error
- Executable Not Found

3. Cara Penanganan Error

- Module Not Found Error

Dengan memperbaiki penulisan atau kesalahan dalam penulisan kode atau melakukan install package atau modul yang belum terinstal

- File Not Found Error

Dengan memperbaiki directory file. Sesuaikan sama tempat penyimpanan di laptop

- Executable Not Found

Dengan menginstal aplikasi graphviz di windows dan menambahkan directory diatas kode program

2.5.4 Bukti Tidak Plagiat

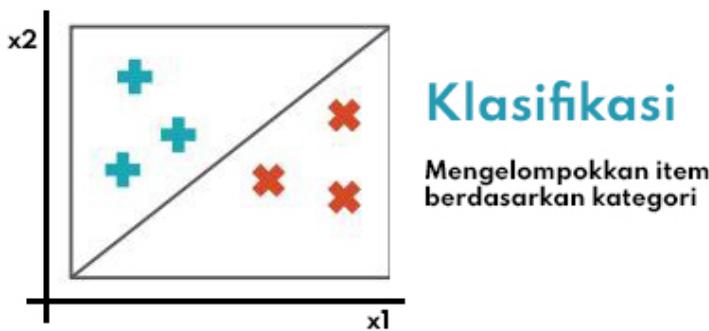


Gambar 2.118 Bukti Tidak Plagiat

2.5.5 Link Youtube

2.6 Arrizak Furqona Gifary (1174070)

2.6.1 Teori



Gambar 2.119 Binary Classification

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

1. Tes medis
untuk menentukan apakah pasien memiliki penyakit tertentu atau tidak - properti klasifikasi adalah keberadaan penyakit.
2. Metode uji "lulus atau gagal"

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

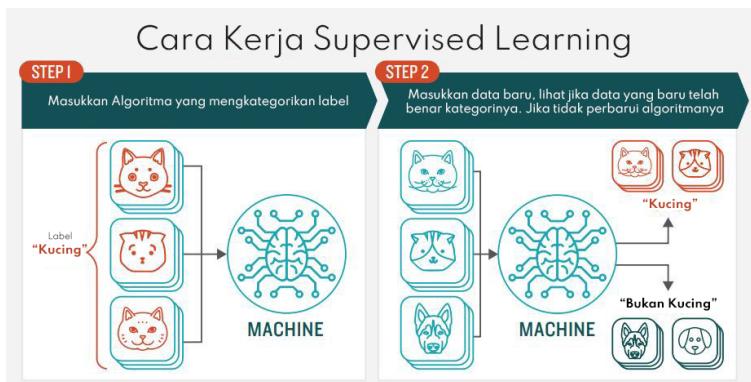
3. Pengambilan informasi

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

2.6.1.2 *Supervised Learning , Unsupervised Learning dan Clustering*

1. Supervised Learning

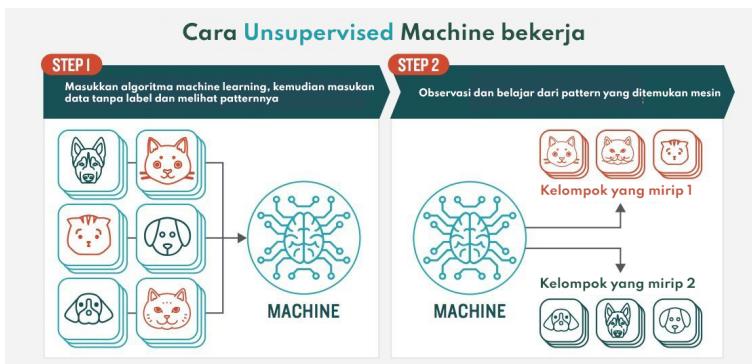
Supervised learning adalah suatu pembelajaran yang terawasi dimana jika output yang diharapkan telah diketahui sebelumnya. Biasanya pembelajaran ini dilakukan dengan menggunakan data yang telah ada. Dan Supervised Learning dalam bahasa indonesia adalah pembelajaran yang ada supervisornya. Maksud disini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



Gambar 2.120 Supervised Learning

2. Unsupervised Learning

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



Gambar 2.121 Unsupervised Learning

3. Clustering

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



Gambar 2.122 Clustering

2.6.1.3 Evaluasi dan Akurasi Evaluasi adalah tentang bagaimana kita dapat mengevaluasi seberapa baik model bekerja dengan mengukur tingkat akurasinya. Dan akurasi akan didefinisikan sebagai persentase kasus yang diklasifikasikan dengan benar. Kita dapat menganalisis kesalahan yang dibuat oleh model, atau tingkat kebingungannya, menggunakan matriks kebingungan(confusion matrix). Matriks kebingungan mengacu pada kebingungan dalam model, tetapi matriks kebingungan ini bisa menjadi sedikit sulit untuk dipahami ketika mereka menjadi sangat besar.

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

Gambar 2.123 Evaluasi

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

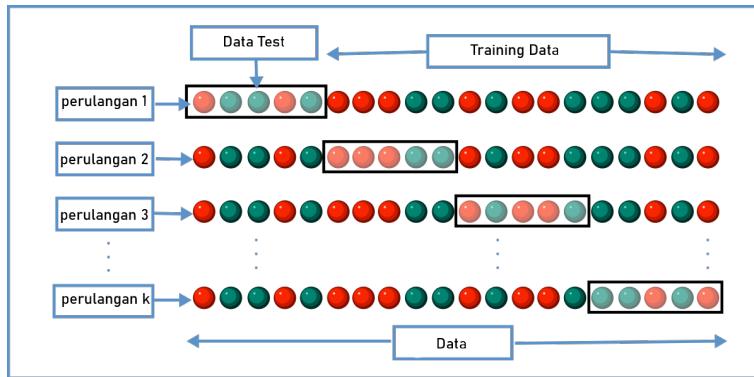
- True Positive: Data positif yang terdeteksi memiliki hasil benar
- False Positive: Data Positif yang terdeteksi memiliki hasil salah
- True Negative: Data negatif yang terdeteksi memiliki hasil benar
- False Negative: Data negatif yang terdeteksi memiliki hasil salah

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

Gambar 2.124 Contoh Confusion Matrix

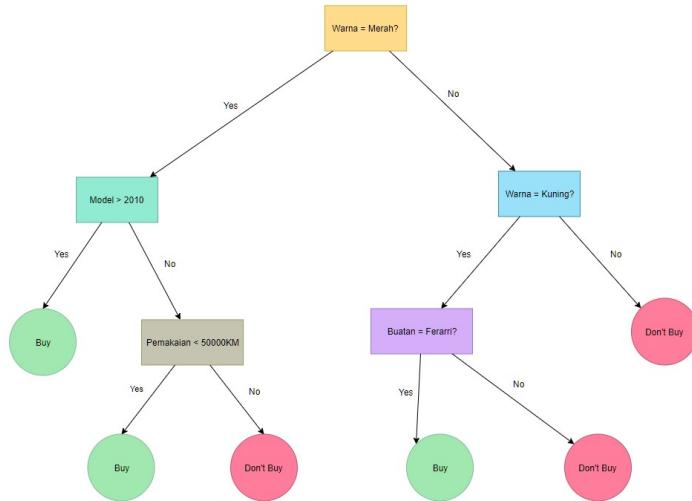
2.6.1.5 Bagaimana K-fold cross validation bekerja

1. Total instance dibagi menjadi N bagian.
2. Fold yang pertama adalah bagian pertama menjadi data uji (testing data) dan sisanya menjadi training data.
3. Lalu hitung akurasi berdasarkan porsi data tersebut dengan menggunakan persamaan.
4. Fold yang kedua adalah bagian kedua, yang menjadi data uji(testing data) dan sisanya training data.
5. Kemudian hitung akurasi berdasarkan porsi data tersebut.
6. Dan seterusnya hingga habis mencapai fold ke-K.
7. Terakhir hitung rata-rata akurasi K buah.



Gambar 2.125 Contoh K-Fold Cross Validation

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



Gambar 2.126 Contoh Decision Tree membeli mobil

2.6.1.7 Information Gain dan Entropi

- Information Gain

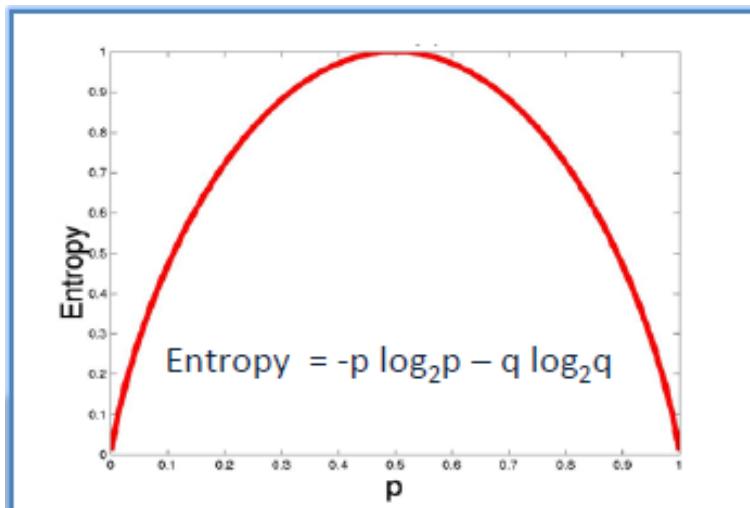
Information Gain merupakan total data yang didapat dari data - data acak yang data tersebut akan digunakan untuk analisis data lainnya. Information Gain ini digunakan pada decision tree sebagai label setiap aksi - aksi yang perlu dinilai validasinya.



Gambar 2.127 Information Gain

- Entropi

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



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

Gambar 2.128 Entropi

2.6.2 Praktek

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

```

1 # In [0]
2 1174070 % 3 #Hasilnya 1 maka akan menggunakan nama Kota

```

2.6.2.1 Nomor 1

```

1 import pandas as pd #Import library pandas menggantinya nama yang
    akan dipanggil jadi pd
2 tokyo = pd.read_csv('dataset/student-mat.csv', sep=';') #Membuat
    variabel tokyo yang isinya memanggil fungsi membaca file csv
3 len(tokyo) #Menghitung jumlah data yang ada pada csv yang tadi
    sudah dibaca

```

Gambar 2.129 Nomor 1

2.6.2.2 Nomor 2

```
1 # In [2]
2 # generate binary label (pass/fail) based on G1+G2+G3 (test
3 #   grades, each 0-20 pts); threshold for passing is sum>=30
4 tokyo['pass'] = tokyo.apply(lambda row: 1 if (row['G1']+row['G2']
5 #   )+row['G3']) >= 35 else 0, axis=1) #Membuat label binary (
6 #   pass/fail) berdasarkan G1+G2+G3 (testgrade, semuanya 0-20
7 #   point); Batas untuk pass adalah sum>=30
8 tokyo = tokyo.drop(['G1', 'G2', 'G3'], axis=1) #Meghilangkan data
9 #   G1 G2 dan G3
10 tokyo.head() #Menampilkan data
```

```
[1]: token = tokens[0].apply(lambda row: 1 if (row['G1']>=90)&(row['G2']>=90)&(row['G3']>=90) else 0, axis=1)
token.head()

Variable explorer File editor Help
Python console
In [1]: token = tokens[0].apply(lambda row: 1 if (row['G1']>=90)&(row['G2']>=90)&(row['G3']>=90) else 0, axis=1)
token.head()

Out[1]: 0    0
1    1
2    0
3    0
4    0
5    0
6    0
7    0
8    0
9    0
10   0
11   0
12   0
13   0
14   0
15   0
16   0
17   0
18   0
19   0
20   0
21   0
22   0
23   0
24   0
25   0
26   0
27   0
28   0
29   0
30   0
31   0
32   0
33   0
34   0
35   0
36   0
37   0
38   0
39   0
40   0
41   0
42   0
43   0
44   0
45   0
46   0
47   0
48   0
49   0
50   0
51   0
52   0
53   0
54   0
55   0
56   0
57   0
58   0
59   0
60   0
61   0
62   0
63   0
64   0
65   0
66   0
67   0
68   0
69   0
70   0
71   0
72   0
73   0
74   0
75   0
76   0
77   0
78   0
79   0
80   0
81   0
82   0
83   0
84   0
85   0
86   0
87   0
88   0
89   0
90   0
91   0
92   0
93   0
94   0
95   0
96   0
97   0
98   0
99   0
100  0
101  0
102  0
103  0
104  0
105  0
106  0
107  0
108  0
109  0
110  0
111  0
112  0
113  0
114  0
115  0
116  0
117  0
118  0
119  0
120  0
121  0
122  0
123  0
124  0
125  0
126  0
127  0
128  0
129  0
130  0
131  0
132  0
133  0
134  0
135  0
136  0
137  0
138  0
139  0
140  0
141  0
142  0
143  0
144  0
145  0
146  0
147  0
148  0
149  0
150  0
151  0
152  0
153  0
154  0
155  0
156  0
157  0
158  0
159  0
160  0
161  0
162  0
163  0
164  0
165  0
166  0
167  0
168  0
169  0
170  0
171  0
172  0
173  0
174  0
175  0
176  0
177  0
178  0
179  0
180  0
181  0
182  0
183  0
184  0
185  0
186  0
187  0
188  0
189  0
190  0
191  0
192  0
193  0
194  0
195  0
196  0
197  0
198  0
199  0
200  0
201  0
202  0
203  0
204  0
205  0
206  0
207  0
208  0
209  0
210  0
211  0
212  0
213  0
214  0
215  0
216  0
217  0
218  0
219  0
220  0
221  0
222  0
223  0
224  0
225  0
226  0
227  0
228  0
229  0
230  0
231  0
232  0
233  0
234  0
235  0
236  0
237  0
238  0
239  0
240  0
241  0
242  0
243  0
244  0
245  0
246  0
247  0
248  0
249  0
250  0
251  0
252  0
253  0
254  0
255  0
256  0
257  0
258  0
259  0
260  0
261  0
262  0
263  0
264  0
265  0
266  0
267  0
268  0
269  0
270  0
271  0
272  0
273  0
274  0
275  0
276  0
277  0
278  0
279  0
280  0
281  0
282  0
283  0
284  0
285  0
286  0
287  0
288  0
289  0
290  0
291  0
292  0
293  0
294  0
295  0
296  0
297  0
298  0
299  0
300  0
301  0
302  0
303  0
304  0
305  0
306  0
307  0
308  0
309  0
310  0
311  0
312  0
313  0
314  0
315  0
316  0
317  0
318  0
319  0
320  0
321  0
322  0
323  0
324  0
325  0
326  0
327  0
328  0
329  0
330  0
331  0
332  0
333  0
334  0
335  0
336  0
337  0
338  0
339  0
340  0
341  0
342  0
343  0
344  0
345  0
346  0
347  0
348  0
349  0
350  0
351  0
352  0
353  0
354  0
355  0
356  0
357  0
358  0
359  0
360  0
361  0
362  0
363  0
364  0
365  0
366  0
367  0
368  0
369  0
370  0
371  0
372  0
373  0
374  0
375  0
376  0
377  0
378  0
379  0
380  0
381  0
382  0
383  0
384  0
385  0
386  0
387  0
388  0
389  0
390  0
391  0
392  0
393  0
394  0
395  0
396  0
397  0
398  0
399  0
400  0
401  0
402  0
403  0
404  0
405  0
406  0
407  0
408  0
409  0
410  0
411  0
412  0
413  0
414  0
415  0
416  0
417  0
418  0
419  0
420  0
421  0
422  0
423  0
424  0
425  0
426  0
427  0
428  0
429  0
430  0
431  0
432  0
433  0
434  0
435  0
436  0
437  0
438  0
439  0
440  0
441  0
442  0
443  0
444  0
445  0
446  0
447  0
448  0
449  0
450  0
451  0
452  0
453  0
454  0
455  0
456  0
457  0
458  0
459  0
460  0
461  0
462  0
463  0
464  0
465  0
466  0
467  0
468  0
469  0
470  0
471  0
472  0
473  0
474  0
475  0
476  0
477  0
478  0
479  0
480  0
481  0
482  0
483  0
484  0
485  0
486  0
487  0
488  0
489  0
490  0
491  0
492  0
493  0
494  0
495  0
496  0
497  0
498  0
499  0
500  0
501  0
502  0
503  0
504  0
505  0
506  0
507  0
508  0
509  0
510  0
511  0
512  0
513  0
514  0
515  0
516  0
517  0
518  0
519  0
520  0
521  0
522  0
523  0
524  0
525  0
526  0
527  0
528  0
529  0
530  0
531  0
532  0
533  0
534  0
535  0
536  0
537  0
538  0
539  0
540  0
541  0
542  0
543  0
544  0
545  0
546  0
547  0
548  0
549  0
550  0
551  0
552  0
553  0
554  0
555  0
556  0
557  0
558  0
559  0
560  0
561  0
562  0
563  0
564  0
565  0
566  0
567  0
568  0
569  0
570  0
571  0
572  0
573  0
574  0
575  0
576  0
577  0
578  0
579  0
580  0
581  0
582  0
583  0
584  0
585  0
586  0
587  0
588  0
589  0
590  0
591  0
592  0
593  0
594  0
595  0
596  0
597  0
598  0
599  0
600  0
601  0
602  0
603  0
604  0
605  0
606  0
607  0
608  0
609  0
610  0
611  0
612  0
613  0
614  0
615  0
616  0
617  0
618  0
619  0
620  0
621  0
622  0
623  0
624  0
625  0
626  0
627  0
628  0
629  0
630  0
631  0
632  0
633  0
634  0
635  0
636  0
637  0
638  0
639  0
640  0
641  0
642  0
643  0
644  0
645  0
646  0
647  0
648  0
649  0
650  0
651  0
652  0
653  0
654  0
655  0
656  0
657  0
658  0
659  0
660  0
661  0
662  0
663  0
664  0
665  0
666  0
667  0
668  0
669  0
670  0
671  0
672  0
673  0
674  0
675  0
676  0
677  0
678  0
679  0
680  0
681  0
682  0
683  0
684  0
685  0
686  0
687  0
688  0
689  0
690  0
691  0
692  0
693  0
694  0
695  0
696  0
697  0
698  0
699  0
700  0
701  0
702  0
703  0
704  0
705  0
706  0
707  0
708  0
709  0
710  0
711  0
712  0
713  0
714  0
715  0
716  0
717  0
718  0
719  0
720  0
721  0
722  0
723  0
724  0
725  0
726  0
727  0
728  0
729  0
730  0
731  0
732  0
733  0
734  0
735  0
736  0
737  0
738  0
739  0
740  0
741  0
742  0
743  0
744  0
745  0
746  0
747  0
748  0
749  0
750  0
751  0
752  0
753  0
754  0
755  0
756  0
757  0
758  0
759  0
760  0
761  0
762  0
763  0
764  0
765  0
766  0
767  0
768  0
769  0
770  0
771  0
772  0
773  0
774  0
775  0
776  0
777  0
778  0
779  0
780  0
781  0
782  0
783  0
784  0
785  0
786  0
787  0
788  0
789  0
790  0
791  0
792  0
793  0
794  0
795  0
796  0
797  0
798  0
799  0
800  0
801  0
802  0
803  0
804  0
805  0
806  0
807  0
808  0
809  0
810  0
811  0
812  0
813  0
814  0
815  0
816  0
817  0
818  0
819  0
820  0
821  0
822  0
823  0
824  0
825  0
826  0
827  0
828  0
829  0
830  0
831  0
832  0
833  0
834  0
835  0
836  0
837  0
838  0
839  0
840  0
841  0
842  0
843  0
844  0
845  0
846  0
847  0
848  0
849  0
850  0
851  0
852  0
853  0
854  0
855  0
856  0
857  0
858  0
859  0
860  0
861  0
862  0
863  0
864  0
865  0
866  0
867  0
868  0
869  0
870  0
871  0
872  0
873  0
874  0
875  0
876  0
877  0
878  0
879  0
880  0
881  0
882  0
883  0
884  0
885  0
886  0
887  0
888  0
889  0
890  0
891  0
892  0
893  0
894  0
895  0
896  0
897  0
898  0
899  0
900  0
901  0
902  0
903  0
904  0
905  0
906  0
907  0
908  0
909  0
910  0
911  0
912  0
913  0
914  0
915  0
916  0
917  0
918  0
919  0
920  0
921  0
922  0
923  0
924  0
925  0
926  0
927  0
928  0
929  0
930  0
931  0
932  0
933  0
934  0
935  0
936  0
937  0
938  0
939  0
940  0
941  0
942  0
943  0
944  0
945  0
946  0
947  0
948  0
949  0
950  0
951  0
952  0
953  0
954  0
955  0
956  0
957  0
958  0
959  0
960  0
961  0
962  0
963  0
964  0
965  0
966  0
967  0
968  0
969  0
970  0
971  0
972  0
973  0
974  0
975  0
976  0
977  0
978  0
979  0
980  0
981  0
982  0
983  0
984  0
985  0
986  0
987  0
988  0
989  0
990  0
991  0
992  0
993  0
994  0
995  0
996  0
997  0
998  0
999  0
1000 0
```

Gambar 2.130 Nomor 2

2.6.2.3 Nomor 3

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

```
tokyo = pd.get_dummies(tokyo, columns=['sex', 'school', 'address', 'family', 'status', 'high', 'job',
                                         'reason', 'parent1', 'schoolsup', 'famrel', 'medu', 'activities',
                                         'varray', 'higher', 'internet', 'romantic'])

tokyo.head(10)
```

```
In [15]: tokyo = pd.get_dummies(tokyo, columns=['sex', 'school', 'address', 'family', 'status', 'high', 'job',
...                                         'reason', 'parent1', 'schoolsup', 'famrel', 'medu', 'activities',
...                                         'varray', 'higher', 'internet', 'romantic'])

Out[15]:
```

```
In [15]: tokyo
```

#	id	Fedu	Medu	Internet_yes	romantic_no	romantic_yes
0	1	4	4	1	0	0
1	2	4	4	1	0	0
2	3	1	3	1	1	0
3	4	1	3	1	1	0
4	5	1	3	1	1	0
5	6	1	3	1	1	0
6	7	1	3	1	1	0
7	8	1	3	1	1	0
8	9	1	3	1	1	0
9	10	1	3	1	1	0

```
In [16]: tokyo.shape
```

```
In [16]:
```

Gambar 2.131 Nomor 3

2.6.2.4 Nomor 4

```
1 # In [4]:  
2 # shuffle rows  
3 tokyo = tokyo.sample(frac=1) #Mengambil data sample dari tokyo  
4 # split training and testing data  
5 tokyo_train = tokyo[:500] #Membagi data untuk training  
tokyo_test = tokyo[500:] #Membagi data untuk test  
6  
7 tokyo_train_att = tokyo_train.drop(['pass'], axis=1) #Meghapus  
    data yang telah pass dan memasukkannya  
8 tokyo_train_pass = tokyo_train['pass'] #Mengambil data yang pass  
    saja  
9  
10 tokyo_test_att = tokyo_test.drop(['pass'], axis=1) #Meghapus data  
    yang telah pass dan memasukkannya  
11 tokyo_test_pass = tokyo_test['pass'] #Mengambil data yang pass  
    saja  
12  
13 tokyo_att = tokyo.drop(['pass'], axis=1) #Meghapus data yang  
    telah pass dan memasukkannya  
14 tokyo_pass = tokyo['pass'] #Mengambil data yang pass saja  
15  
16  
17 # number of passing students in whole dataset:  
18 import numpy as np #Mengimport library numpy sebagai np  
19 print("Passing: %d out of %d (%.2f%%)" % (np.sum(tokyo_pass), len  
    (tokyo_pass), 100*float(np.sum(tokyo_pass)) / len(tokyo_pass)  
)) #Menampilkan data
```

Gambar 2.132 Nomor 4

2.6.2.5 Nomor 5

```
1 # In [5]:  
2 # fit a decision tree  
3 from sklearn import tree #import Decision tree dari library  
    sklearn
```

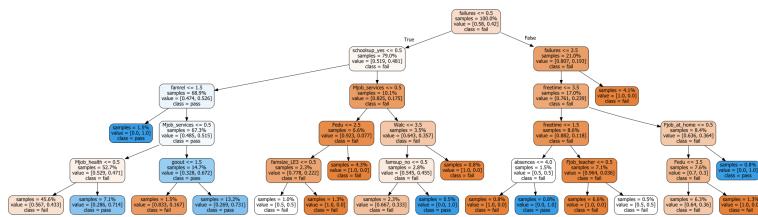
```
4 kyoto = tree.DecisionTreeClassifier(criterion="entropy",
      max_depth=5) #Membuat decision tree dengan maximal depthnya 5
5 kyoto = kyoto.fit(tokyo_train_att, tokyo_train_pass) #Memasukkan
      data yang akan dijadikan decision treenya
```

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

Gambar 2.133 Nomor 5

2.6.2.6 Nomor 6

```
1 # In [6]:  
2 # visualize tree  
3 import graphviz #Mengimport Library Graphviz untuk  
     memvisualisasikan decision tree  
4 dot_data = tree.export_graphviz(kyoto, out_file=None, label="all"  
     , impurity=False, proportion=True,  
     feature_names=list(  
         tokyo_train_att), class_names=["fail", "pass"],  
     filled=True, rounded=True) #  
     Mendefinisikan dot_data yang isikan akan berisikan data yang  
     akan dijadikan gambar  
7 graph = graphviz.Source(dot_data) #Memasukkan data tadi menjadi  
     sebuah graph  
8 graph #Menampilkan graph menggunakan graphviz
```



Gambar 2.134 Nomor 6

2.6.2.7 Nomor 7

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

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

Gambar 2.135 Nomor 7

2.6.2.8 Nomor 8

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

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

Gambar 2.136 Nomor 8

2.6.2.9 Nomor 9

```
1 # In [9]:
2 from sklearn.model_selection import cross_val_score #Mengimport
  fungsi cross_val_score dari library sklearn
3 nagoya = cross_val_score(kyoto, tokyo_att, tokyo_pass, cv=5) #
  Mendefinisikan nagoya yang isinya pembagian data menjadi 5
4 # show average score and +/- two standard deviations away (
  covering 95% of scores)
5 print("Accuracy: %.2f (+/- %.2f)" % (nagoya.mean(), nagoya.std()
  () * 2)) #Menampilkan data nilai dan +/- dari dua standar
  deviasi
```

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

Gambar 2.137 Nomor 9

2.6.2.10 Nomor 10

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

```
In [70]: for max_depth in range(1, 20):
...:     kyoto = tree.DecisionTreeClassifier(criterion="entropy", max_depth=max_depth)
...:     nagoya = cross_val_score(kyoto, tokyo_att, tokyo_pass, cv=5)
...:     print("Max depth: %d, Accuracy: %0.2f (+/- %0.2f)" % (max_depth, nagoya.mean(), nagoya.std() * 2))
Max depth: 1, Accuracy: 0.50 (+/- 0.01)
Max depth: 2, Accuracy: 0.50 (+/- 0.08)
Max depth: 3, Accuracy: 0.50 (+/- 0.12)
Max depth: 4, Accuracy: 0.57 (+/- 0.08)
Max depth: 5, Accuracy: 0.55 (+/- 0.09)
Max depth: 6, Accuracy: 0.57 (+/- 0.18)
Max depth: 7, Accuracy: 0.57 (+/- 0.15)
Max depth: 8, Accuracy: 0.54 (+/- 0.18)
Max depth: 9, Accuracy: 0.54 (+/- 0.11)
Max depth: 10, Accuracy: 0.58 (+/- 0.11)
Max depth: 11, Accuracy: 0.56 (+/- 0.10)
Max depth: 12, Accuracy: 0.60 (+/- 0.08)
Max depth: 13, Accuracy: 0.59 (+/- 0.05)
Max depth: 14, Accuracy: 0.58 (+/- 0.05)
Max depth: 15, Accuracy: 0.57 (+/- 0.07)
Max depth: 16, Accuracy: 0.58 (+/- 0.06)
Max depth: 17, Accuracy: 0.60 (+/- 0.11)
Max depth: 18, Accuracy: 0.58 (+/- 0.08)
Max depth: 19, Accuracy: 0.58 (+/- 0.06)
```

Gambar 2.138 Nomor 10

2.6.2.11 Nomor 11

```
1 # In[11]:
2 depth_acc = np.empty((19,3), float) #Membuat array baru
3 i = 0 #Membuat variable berisikan 0
4 for max_depth in range(1, 20): #Perulangan untuk memasukkan data
5     kyoto = tree.DecisionTreeClassifier(criterion="entropy",
6         max_depth=max_depth)#Membuat decision Tree
7     nagoya = cross_val_score(kyoto, tokyo_att, tokyo_pass, cv=5)
8     #Mendefinisikan nagoya yang isinya pembagian data menjadi 5
9     depth_acc[i,0] = max_depth #Memasukkan data max_depth ke
10    array depth_acc
11    depth_acc[i,1] = nagoya.mean() #Memasukkan data rata-rata
12    dari nagoya ke array depth_acc
13    depth_acc[i,2] = nagoya.std() * 2 #Memasukkan data akar 2
14    dari nagoya ke array depth_acc
15    i += 1
16
17 depth_acc
```

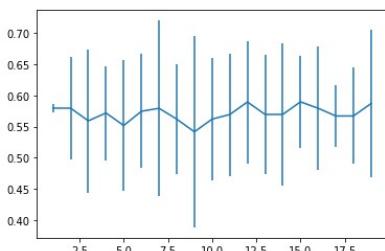
```
In [71]: depth_acc = np.empty((19,3), float)
...: i = 0
...: for max_depth in range(1, 20):
...:     kyoto = tree.DecisionTreeClassifier(criterion="entropy", max_depth=max_depth)
...:     nagoya = cross_val_score(kyoto, tokyo_att, tokyo_pass, cv=5)
...:     depth_acc[i,0] = max_depth
...:     depth_acc[i,1] = nagoya.mean()
...:     depth_acc[i,2] = nagoya.std() * 2
...:     i += 1
...:
...:
...: depth_acc
Out[71]:
array([[1.0000000e+00, 5.79751704e-01, 6.30768599e-03],
       [2.0000000e+00, 5.79654333e-01, 8.25005697e-02],
       [3.0000000e+00, 5.54241318e-01, 1.21808510e-01],
       [4.0000000e+00, 5.71964460e-01, 8.72318860e-02],
       [5.0000000e+00, 5.46678027e-01, 9.66616528e-02],
       [6.0000000e+00, 5.69561819e-01, 1.08113451e-01],
       [7.0000000e+00, 5.67030996e-01, 1.40406275e-01],
       [8.0000000e+00, 5.47030996e-01, 1.22447311e-01],
       [9.0000000e+00, 5.51966082e-01, 6.6884125e-02],
       [1.0000000e+01, 5.87314318e-01, 7.12478298e-02],
       [1.1000000e+01, 5.77124310e-01, 7.61119153e-02],
       [1.2000000e+01, 5.89719247e-01, 4.34452239e-02],
       [1.3000000e+01, 5.97569783e-01, 3.87484760e-02],
       [1.4000000e+01, 5.77026939e-01, 7.68881184e-02],
       [1.5000000e+01, 5.97347452e-01, 5.69404888e-02],
       [1.6000000e+01, 5.74720058e-01, 1.29000478e-01],
       [1.7000000e+01, 5.82282538e-01, 5.71762018e-02],
       [1.8000000e+01, 5.74720870e-01, 3.56163418e-02],
       [1.9000000e+01, 5.77250893e-01, 8.75345835e-02]])
```

Gambar 2.139 Nomor 11

2.6.2.12 Nomor 12

```
1 # In [12]:
2 import matplotlib.pyplot as plt #Menimport fungsi pyplot dari
   library matplotlib sebagai plt
3 fig, ax = plt.subplots() #Membuat plot baru
4 ax.errorbar(depth_acc[:,0], depth_acc[:,1], yerr=depth_acc[:,2]) #Mengisikan data plot
5 plt.show() #Menampilkan plot
```

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



Gambar 2.140 Nomor 12

2.6.3 Penanganan Error

2.6.3.1 Error

1. ModuleNotFoundError

Traceback (most recent call last):

```
File "<ipython-input-25-af85b140ad99>", line 1, in <module>
    import graphviz
```

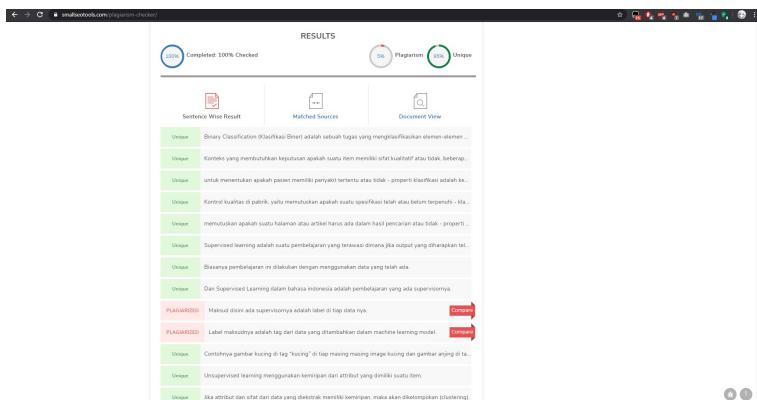
```
ModuleNotFoundError: No module named 'graphviz'
```

Gambar 2.141 ModuleNotFoundError

2.6.3.2 Solusi

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

2.6.4 Bukti Tidak Plagiat



Gambar 2.142 Bukti Tidak Plagiat

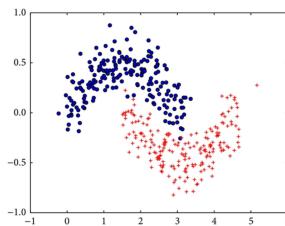
2.6.5 Link Youtube:

https://youtu.be/_rB-Z2xMMdk

2.7 Nurul Izza Hamka — 1174062

2.7.1 Teori

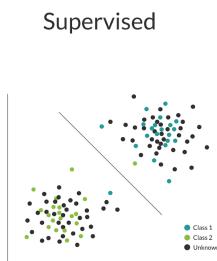
1. Apa itu Binary Classification dilengkapi Ilustrasi Buatan sendiri Binary Classification adalah mengklasifikasikan sebuah elemen-elemen dari suatu himpunan yang di masukkan kedalam dua kelompok dengan memprediksi kelompok mana yang masing-masing dimiliki berdasarkan aturan klasifikasi. Konteks yang memerlukan keputusan apakah suatu item memiliki sifat kualitatif atau tidak, beberapa karakteristik tertentu, atau beberapa klasifikasi biner khas



Gambar 2.143 Gambar Binary Classification

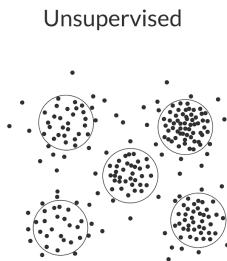
2. Apa itu supervised Learning, Unsupervised Learning, dan Clustering dengan ilustrasi sendiri

Pengertian dalam konteks AI, supervised learning adalah sistem dimana sebuah input dan output data yang kita inginkan sudah tersedia. Input dan output data ini diberi label untuk klasifikasi dasar pembelajaran untuk pemrosesan data yang akan datang. Supervised learning ini menyediakan algoritma untuk pembelajaran dengan jumlah diketahui untuk mendukung sebuah penilaian yang akan datang.



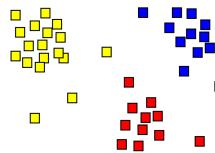
Gambar 2.144 Gambar Supervised Learning

Unsupervised Learning tidak memiliki data latih, sehingga dari data yang telah ada kita kelompokkan menjadi dua ataupun tiga bagian dst. Unsupervised learning ini merupakan pelatihan algoritma kecerdasan buatan menggunakan infomasi yang tidak diklasifikasikan atau diberi label dan memungkinkan algoritma untuk bertindak atas informasi tersebut tanpa bimbingan.



Gambar 2.145 Gambar Unsupervised Learning

Clustering adalah pengelompokan objek dengan sedemikian rupa sehingga objek berada dalam kelompok yang sama (disebut kluster) lebih mirip satu sama lain dibandingkan kelompok lain.



Gambar 2.146 Gambar Clustering

3. Apa itu evaluasi dan akurasi dari buku dan disertai contoh dengan gambar sendiri.

Evaluasi adalah bagaimana kita dapat melakukan evaluasi tentang seberapa baik model bekerja dengan mengukur tingkat akurasinya.

Akurasi ini didefinisikan sebagai sebuah persentase kasus yang diklasifikasikan dengan benar. Analisis dapat dilakukan menggunakan sebuah matriks kebingungan apabila model yang dibuat terdapat kesalaha atau tingkat kebingungannya.

4. Bagaimana cara membuat dan membaca cunfusion matrix, buat cunfusion matrix buatan sendiri

Confusion matrix adalah untuk memberikan informasi sebuah oerbandingan hasil klasifikasi yang di lakukan oleh sebuah sistem atau model dengan hasil klasifikasi yang nyata.

Cara membuat dan membaca cunfusion matrix:

1. Tentukan pokok permasalahan dan atributanya, misal gaji dan listik.
2. Buat pohon keputusan.
3. Lalu data testingnya.
4. Lalu mencari nilai a, b, c, dan d.
5. Selanjutnya mencari nilai recall, precision, accuracy, serta dan error rate.

Confusion Matrix

		Actually Positive (1)	Actually Negative (0)
		True Positives (TPs)	False Positives (FPs)
Predicted Positive (1)	Predicted Positive (1)	True Positives (TPs)	False Positives (FPs)
	Predicted Negative (0)	False Negatives (FNs)	True Negatives (TNs)

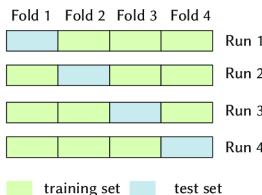
Gambar 2.147 Gambar Cunfusion Matrix

5. Bagaimana K-fold cross validation bekerja, disertai gambar ilustrasi contoh buatan sendiri

K-fold cross validation adalah salah satu metode yang mengevaluasi kinerja classifier, metode ini dapat kita gunakan jika jumlah data terbatas atau jumlah instance tidak banyak.

Cara kerja K-fold Cross validation adalah :

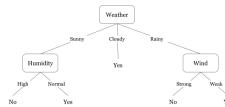
1. Total instance dibagi menjadi N bagian.
2. Fold ke 1 adalah bagian pertama menjadi data uji (testing data) dan sisanya menjadi training data.
3. Fold yang kedua adalah ketika ke 2 menjadi data uji dan sisanya menjadi data latih
4. Hitung akurasi berdasarkan porsi data tersebut
5. Begitupun dengan selanjutnya hingga mencapai fold ke-K, dan hitung rata-rata akurasi dari K buah akurasi tersebut.



Gambar 2.148 Gambar K-Fold Cross Validation

6. Apa itu desicion tree, buat gambar ilustrasi contoh buatan sendiri

Decision tree adalah sebuah metode pembelajaran non-parametrik yang digunakan untuk klasifikasi dan juga regresi. Tujuannya adalah untuk membuat sebuah model yang dapat memprediksi sebuah nilai variable target dengan memahami aturan sederhana dari fitur data.



Gambar 2.149 Gambar Decision Tree

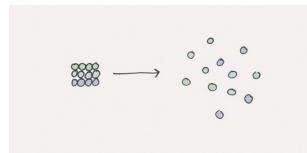
7. Apa itu Information gain dan entropi, gambar ilustrasi contoh buatan sendiri.

Information gain adalah penurunan entropi setelah dataset dibagi pada sebuah atribut. Dalam membangun decision tree semua tentang menemukan atribut yang mengembalikan pribahan informasi tertinggi.

$$\begin{aligned} IG(T, a) &= H(T) - \sum_{v \in vals(a)} P_a(v)H(S_a(v)) \\ &= H(T) - \mathbb{E}_{P_a}[H(S_a(v))] \\ &= H(T) - H(T|a). \end{aligned}$$

Gambar 2.150 Gambar Information Gain

Entropi merupakan ukuran keacakan dalam sebuah informasi yang sedang diproses. Semakin tinggi sebuah entropi maka semakin sulit menarik sebuah kesimpulan dari informasi tersebut.



Gambar 2.151 Gambar Binary Classification

2.7.2 Scikit-learn

1. Nomor 1

```

1
2 # load dataset (menggunakan student-mat.csv)
3 import pandas as pd #mengimport library pandas sebagai pd
4 burger = pd.read_csv('D:/LEC/IT SMT VI/ARTIFICIAL
    INTELLIGENCE/Chapter 2/Src/dataset/student-mat.csv', sep=','
    ;') #variabel median berfungsi untuk read file student-mat.
    csv
5 len(burger) #mengetahui jumlah baris pada data yang dipanggil

```

In [2]: In[2]: len(burger) Out[2]: 35

Gambar 2.152 Gambar Hasil No 1

2. Nomor 2

```

1
2 # generate binary label (pass/fail) based on G1+G2+G3 ( test
    grades , each 0–20 pts); threshold for passing is sum>=30
3 burger[ 'pass' ] = burger.apply(lambda row: 1 if (row[ 'G1' ]+row
    [ 'G2' ]+row[ 'G3' ]) >= 35 else 0, axis=1) #mendeklarasikan
    pass/fail nya data berdasarkan G1+G2+G3.
4 burger= burger.drop([ 'G1' , 'G2' , 'G3' ], axis=1) #untuk
    mengetahui baris G1+G2+G3 ditambahkan , dan hasilnya sama
    dengan 35 maka axisnya 1.
5 burger.head() #memanggil variabel burger dengan ketentuan
    head ini digunakan untuk mengembalikan baris n atas 5
    secara default dari frame atau seri data

```

In [2]: burger['pass'] = burger.apply(lambda row: 1 if (row['G1']+row['G2']+row['G3']) >= 35 else 0, axis=1)

In [2]: burger.drop(['G1', 'G2', 'G3'], axis=1, inplace=True)

In [2]: burger

Gambar 2.153 Gambar Hasil No 2

3. Nomor 3

```

1
2 # use one-hot encoding on categorical columns
3 burger = pd.get_dummies(burger, columns=[ 'sex' , 'school' ,
    'address' , 'famsize' , 'Pstatus' , 'Mjob' , 'Fjob' ,
    'reason' , 'guardian' , 'schoolsup' , 'famsup' , 'paid' ,
    'activities' , 'nursery' , 'higher' , 'internet
    , 'romantic' ]) #variabel burger dikonversi menjadi
    bentuk yang lebih baik dalam prediksi dan memanggil
    seluruh atribut
6 burger.head() #memanggil variabel burger dengan ketentuan
    head ini digunakan untuk mengembalikan baris n atas 5
    secara default dari frame atau seri data

```

```
In [10]: burger = pd.get_dummies(burger, columns=['User', 'Address', 'Address_2', 'Feature', 'Home', 'Type',
       ...], drop_first=True)
        burger['Interact'] = burger['User'].str.cat(burger['Address'], sep='|')
        burger['Interact'] = burger['Interact'].str.cat(burger['Address_2'], sep='|')
        burger['Interact'] = burger['Interact'].str.cat(burger['Feature'], sep='|')
        burger['Interact'] = burger['Interact'].str.cat(burger['Home'], sep='|')
        burger['Interact'] = burger['Interact'].str.cat(burger['Type'], sep='|')

Out[10]:
```

User	Address	Address_2	Feature	Home	Type	Interact
1	1	1	1	1	1	1 1 1 1 1
2	1	1	1	1	1	1 1 1 1 1
3	1	1	1	1	1	1 1 1 1 1
4	1	1	1	1	1	1 1 1 1 1
5	1	1	1	1	1	1 1 1 1 1
6	1	1	1	1	1	1 1 1 1 1
7	1	1	1	1	1	1 1 1 1 1
8	1	1	1	1	1	1 1 1 1 1
9	1	1	1	1	1	1 1 1 1 1
10	1	1	1	1	1	1 1 1 1 1
11	1	1	1	1	1	1 1 1 1 1
12	1	1	1	1	1	1 1 1 1 1
13	1	1	1	1	1	1 1 1 1 1
14	1	1	1	1	1	1 1 1 1 1
15	1	1	1	1	1	1 1 1 1 1
16	1	1	1	1	1	1 1 1 1 1
17	1	1	1	1	1	1 1 1 1 1
18	1	1	1	1	1	1 1 1 1 1
19	1	1	1	1	1	1 1 1 1 1

Gambar 2.154 Gambar Hasil No 3

4. Nomor 4

```
1
2 # shuffle rows
3 burger = burger.sample(frac=1) #mengembalikan variabel burger
       menjadi sampel acak dengan frac=1
4 # split training and testing data
5 burger_train = burger[:500] #membuat variabel baru
       burger_train
6 burger_test = burger[500:] #membuat variabel baru burger_test
       yang sisanya dari train
7
8 burger_train_att = burger_train.drop(['pass'], axis=1) #
       membuat variabel baru dengan ketentuan dari burger_train
9 burger_train_pass = burger_train['pass'] #membuat variabel baru
       dengan ketentuan dari burger_train
10
11 burger_test_att = burger_test.drop(['pass'], axis=1) #membuat
       variabel baru dengan ketentuan dari burger-test
12 burger_test_pass = burger_test['pass'] #membuat variabel baru
       dengan ketentuan dari burger-test
13
14 burger_att = burger.drop(['pass'], axis=1) #membuat variabel
       burger_att sebagai salinan dari burger
15 burger_pass = burger['pass'] #membuat variabel burger-pass
       sebagai salinan dari burger
16
17 # number of passing students in whole dataset:
18 import numpy as np #mengimport module numpy sebagai np y
19 print("Passing: %d out of %d (%.2f%%)" % (np.sum(burger_pass),
       , len(burger_pass), 100*float(np.sum(burger_pass)) / len(
       burger_pass))) #untuk mengembalikan nilai passing dari
       pelajar dari keseluruhan dataset dengan cara print.
```

```
In [11]: burger = burger.sample(frac=1) #mengembalikan variabel burger menjadi sampel acak dengan frac=1
        burger_train = burger[:500] #membuat variabel data burger_train
        burger_train_att = burger_train.drop(['pass'], axis=1) #membuat variabel data burger_train_att
        burger_train_pass = burger_train['pass'] #membuat variabel data burger_train_pass
        burger_test = burger[500:] #membuat variabel data burger-test
        burger_test_att = burger_test.drop(['pass'], axis=1) #membuat variabel data burger-test_att
        burger_test_pass = burger_test['pass'] #membuat variabel data burger-test_pass
        burger_att = burger.drop(['pass'], axis=1) #membuat variabel burger_att sebagai salinan dari burger
        burger_pass = burger['pass'] #membuat variabel burger-pass sebagai salinan dari burger
        burger_att = burger_att.dropna() #menghapus baris yang kosong pada data burger_att
        burger_pass = burger_pass.dropna() #menghapus baris yang kosong pada data burger_pass
        burger_att = burger_att.reset_index() #memulihkan index pada data burger_att
        burger_pass = burger_pass.reset_index() #memulihkan index pada data burger_pass
        burger_att['Index'] = burger_att['Index'].apply(lambda x: str(x))
        burger_pass['Index'] = burger_pass['Index'].apply(lambda x: str(x))

Out[11]:
```

Gambar 2.155 Gambar Hasil No 4

5. Nomor 5

```
1
2 # fit a decision tree
```

```

3 from sklearn import tree #import tree dari library sklearn
4 pizza = tree.DecisionTreeClassifier(criterion="entropy",
5                                     max_depth=5) #membuat variabel pizza sebagai decisiontree ,
6                                     dengan criterion fungsi mengukur kualitas split
7 pizza = pizza.fit(burger_train_att , burger_train_pass) # training varibael pizza dengan data dari variabel burger.

```

In [3]: from sklearn import tree import time from joblib import load
... pizza = tree.DecisionTreeClassifier(criterion="entropy", max_depth=5)
... pizza = pizza.fit(burger_train_att , burger_train_pass) #training varibael pizza dengan data dari variabel burger.

Gambar 2.156 Gambar Hasil No 5

6. Nomor 6

```

1
2 # visualize tree
3 import os
4 os.environ["PATH"] += os.pathsep + 'C:/Program Files (x86)/
5 Graphviz2.38/bin/'
6 import graphviz #import library graphviz sebagai perangkat
7 lunak visualisasi grafik open source
8 dot_data = tree.export_graphviz(pizza , out_file=None, label="all",
9                                impurity=False , proportion=True,
10                               feature_names=list(
11                                   burger_train_att), class_names=["fail", "pass"],
12                               filled=True, rounded=True) #
13                               mengambil data untuk diterjemahkan ke grafik
14 graph = graphviz.Source(dot_data) #membuat variabel graph
15 sebagai grafik yang di ambil dari dot_data
16 graph #memanggil graph

```

In [4]: import os
... os.environ["PATH"] += os.pathsep + 'C:/Program Files (x86)/Graphviz2.38/bin'
... dot_data = tree.export_graphviz(pizza , out_file=None, label="all",
... impurity=False , proportion=True,
... feature_names=list(burger_train_att), class_names=["fail", "pass"],
... filled=True, rounded=True)
... graph = graphviz.Source(dot_data) #membuat variabel graph sebagai grafik yang di ambil dari dot_data

Gambar 2.157 Gambar Hasil No 6



Gambar 2.158 Gambar Hasil No 6

7. Nomor 7

```

1
2 # save tree
3 tree.export_graphviz(pizza , out_file="student-performance.dot"
4                      , label="all" , impurity=False , proportion=True,

```

```

4         feature_names=list(burger_train_att),
5         class_names=[“ fail”, “ pass”],
6             filled=True, rounded=True) #save tree
7 sebagai export graphviz ke file student-performance.dot

```

```
In [5]: tree.export_graphviz(tree, out_file='student-performance.dot', feature_names=feature_names,
...                         class_names=class_names, filled=True, rounded=True)
```

Gambar 2.159 Gambar Hasil No 7

8. Nomor 8

```

1 #pizza.score(median_test_att, median_test_pass)
2 pizza.score(burger_att, burger_pass) #score juga disebut
3 prediksi dengan diberi beberapa data input baru

```

```
In [6]: pizza.score(burger_att, burger_pass) #score juga disebut prediksi dengan beberapa data input baru
Out[6]: 0.70120115248598
```

Gambar 2.160 Gambar Hasil No 8

9. Nomor 9

```

1 from sklearn.model_selection import cross_val_score #import
2         class cross_val_score dari sklearn
3 scores = cross_val_score(pizza, burger_att, burger_pass, cv
4 =5) #mengevaluasi score dengan validasi silang
5 # show average score and +/- two standard deviations away (
6     covering 95% of scores)
7 print("Accuracy: %0.2f (+/- %0.2f)" % (scores.mean(), scores.
8     std() * 2)) #print akurasi

```

```
In [7]: from sklearn.model_selection import cross_val_score
cross_val_score(pizza, burger_att, burger_pass, cv=5)
# mengevaluasi score dengan validasi silang
# show average score and +/- two standard deviations away (covering 95% of scores)
Accuracy: 0.54 (+/- 0.07)
```

Gambar 2.161 Gambar Hasil No 9

10. Nomor 10

```

1 for max_depth in range(1, 20):
2     pizza = tree.DecisionTreeClassifier(criterion="entropy",
3         max_depth=max_depth)
4     scores = cross_val_score(pizza, burger_att, burger_pass,
5     cv=5)
6     print("Max depth: %d, Accuracy: %0.2f (+/- %0.2f)" % (
7         max_depth, scores.mean(), scores.std() * 2))

```

6 #Disini ini menunjukkan seberapa dalam di tree itu.
Semakin dalam tree , semakin banyak perpecahan yang
dimilikinya dan menangkap lebih banyak informasi tentang
data .

```
[4]: [x for x in depth if x[0] == 'Accuracy' and x[1] == 'Burgers' and x[2] == 'Score']
```

Gambar 2.162 Gambar Hasil No 10

11. Nomor 11

```
1 depth_acc = np.empty((19,3), float) #Dengan 19 sebagai bentuk
2         array kosong, 3 sebagai output data-type
3 kentang = 0 #variabel kentang sebagai array 0
4 for max_depth in range(1, 20): #perulangan dengan max_depth
5     pizza = tree.DecisionTreeClassifier(criterion="entropy",
6                                         max_depth=max_depth) #variabel pizza untuk decision tree
7     dengan ketentuan entropy
8     scores = cross_val_score(pizza, burger_att, burger_pass,
9                                cv=5) #scores diambil dari data cross_val_score
10    depth_acc[kentang,0] = max_depth #mengembalikan array
11    dengan ketentuan 0 dan max_depth
12    depth_acc[kentang,1] = scores.mean() #mengembalikan array
13    dengan ketentuan 1 dan scores.mean
14    depth_acc[kentang,2] = scores.std() * 2 #mengembalikan
15    array dengan ketentuan 2 dan scores.std, std berarti
16    menghitung standar deviasi
17    kentang += 1
18
19 depth_acc #Depth acc akan membuat array kosong dengan
20         mengembalikan array baru dengan bentuk dan tipe yang
21         diberikan
```

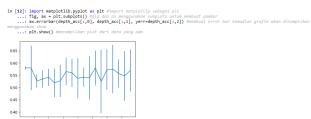
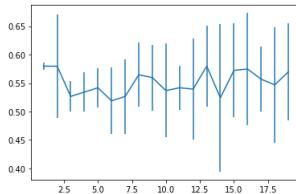
Gambar 2.163 Gambar Hasil No 11

12. Nomor 12

```

2 import matplotlib.pyplot as plt #import matplotlib sebagai
     plt
3 fig , ax = plt.subplots() #fig dan ax menggunakan subplots
     untuk membuat gambar
4 ax.errorbar(depth_acc[:,0] , depth_acc[:,1] , yerr=depth_acc
     [:,2]) #membuat error bar kemudian grafik akan ditampilkan
     menggunakan show
5 plt.show() #menampilkan plot dari data yang ada

```

**Gambar 2.164** Gambar Hasil No 12**Gambar 2.165** Gambar Hasil No 12

2.7.3 Penanganan error

1. Hasil Screenshoot Error

```

File "C:\Python\input-7-f751109209fa", line 1, in <module>
    import graphviz #import library graphviz sebagai perangkat
    lunak visualisasi grafik open source
ModuleNotFoundError: No module named 'graphviz'

```

Gambar 2.166 Gambar Module Not Found Error

2. Cara penanganannya adalah dengan memperbaiki penulisannya yang salah dalam penulisan kode atau melakukan penginstallan package atau modul yang belum terinstal.

```
FileNotFoundException: File b'dataset/student-mat.csv' does not exist
```

Gambar 2.167 Gambar Error Bagian 2

3. Cara penanganannya adalah memperbaiki directory file. Dengan menyesuaikan tempat penyimpanan di laptop.

```
File "C:\ProgramData\Anaconda3\lib\site-packages\graphviz\\  
backend.py", line 162, in run  
    raise ExecutableNotFoundError(cmd)  
  
ExecutableNotFoundError: failed to execute ['dot', '-Tsvg'], make sure  
the Graphviz executables are on your systems' PATH  
  
[Out[8]: graphviz.files.Source at 0x259e5d0780]
```

Gambar 2.168 Gambar Error Bagian 3

4. Cara penanganannya lakukan penginstallan graphviz di windows dan menambahkan directory diatas pada kode program.

2.7.4 Bukti Tidak Plagiat

- #### 1. Bukti Tidak Melakukan Plagiat



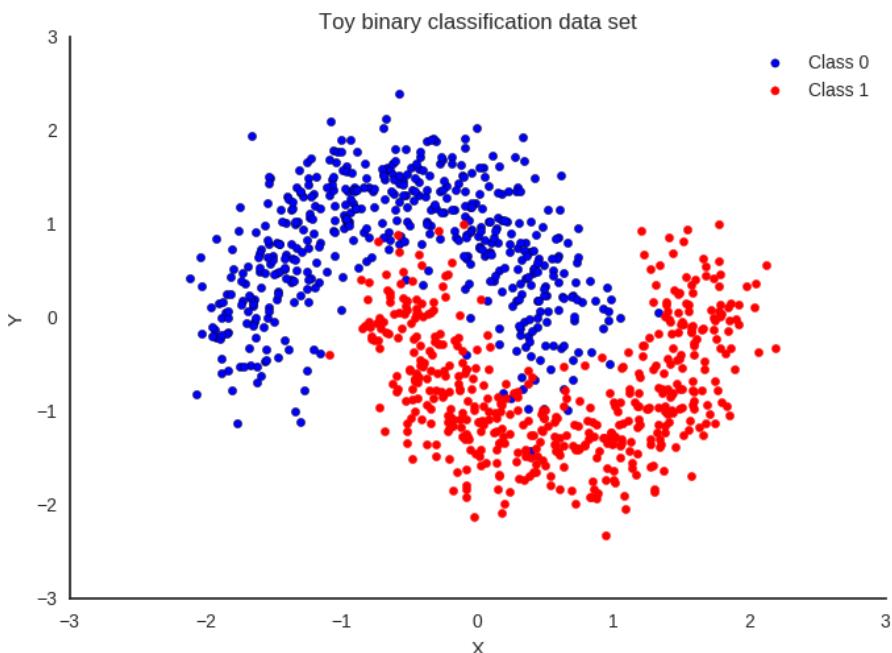
Gambar 2.169 Bukti Tidak Plagiat

2.7.5 Link Youtube

2.8 1174084 - Muhammad Reza Syachrani

2.8.1 Teori

1. Jelaskan Apa Itu Binary Classification dilengkapi ilustrasi gambar



Gambar 2.170 gambaran binary classification

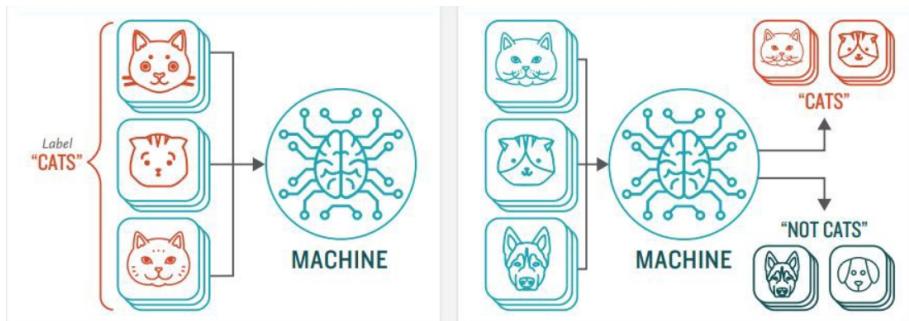
Klasifikasi biner atau binomial adalah tugas mengklasifikasikan elemen-elemen dari himpunan yang diberikan ke dalam dua kelompok (memprediksi kelompok mana yang masing-masing dimiliki) berdasarkan aturan klasifikasi.

2. Jelaskan Apa itu supervised learning , unsupervised learning dan clustering dengan ilustrasi gambar

- (a) supervised learning

Dalam bahasa indonesia Supervised Learning adalah pembelajaran yang ada supervisornya. Maksud ada supervisornya adalah label di tiap data nya. Label adalah tag dari data yang ditambahkan dalam machine learning model. Contohnya gambar kucing di tag “kucing” di tiap masing masing image kucing dan gambar anjing di tag “anjing” di tiap masing gambar anjing. Machine learning kategori berupa clasification (“anjing”, “kucing”, “beruang”, dsb) dan regression (berat badan, tinggi badan dsb). Supervised learning digunakan untuk memprediksi pola yang sudah ada contoh data yang lengkap, jadi pola yang terbentuk adalah hasil pembelajaran data lengkap tersebut. Jika kita memasukan data baru, setelah melakukan ETL (Extract Transform Load) maka kita mendapat info feature dari sample

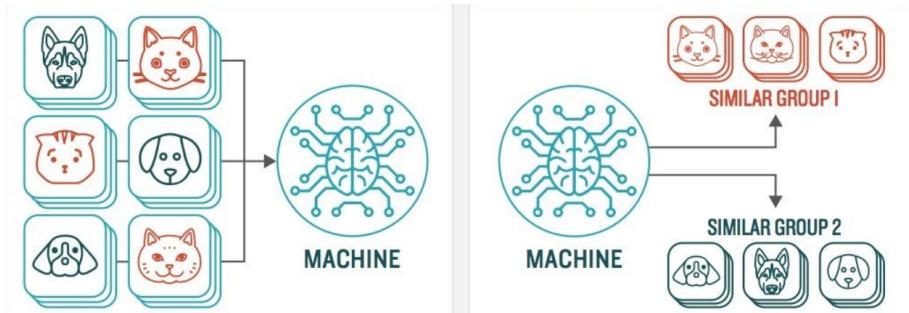
baru tersebut. Kemudian feature tersebut di compare dengan pattern classification dari model yang didapat dari label data. Setiap label di compare hingga selesai, dan yang memiliki percentage lebih banyak akan diambil sebagai prediksi akhir.



Gambar 2.171 gambaran cara kerja supervised

(b) unsupervised learning

Unsupervised learning tidak menggunakan label seperti supervised learning dalam memprediksi target features / variable. Melainkan menggunakan kesamaan dari attribut-attribut yang dimiliki. Apabila attribut dan sifat-sifat dari data-data feature yang diekstrak memiliki kemiripan-miripan, maka akan dikelompokkan-kelompokan (clustering). Sehingga hal ini akan menimbulkan kelompok-kelompok (cluster). Jumlah cluster bisa unlimited. Dari kelompok itu model melabelkan, dan jika data baru mau di prediksi, maka akan dicocokkan dengan kelompok yang mirip dengan featurenya.

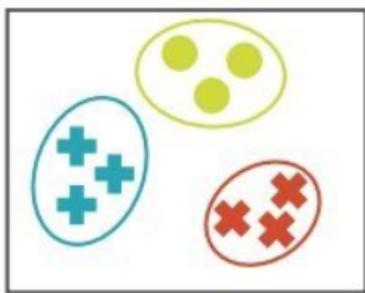


Gambar 2.172 gambaran cara kerja unsupervised

(c) Clustering

Clustering adalah sebuah metode untuk membedakan data-data menjadi kumpulan dari group yang isinya merupakan data yang serupa

setiap grupnya. Basisnya dapat berupa kesamaan atau perbedaan dari setiap grup tersebut.



CLUSTERING

Identifying similarities in groups

For Example: Are there patterns in the data to indicate certain patients will respond better to this treatment than others?

Gambar 2.173 gambaran clustering

- Jelaskan apa itu evaluasi dan akurasi dan disertai ilustrasi contoh dengan gambar

Evaluasi adalah tentang bagaimana kita dapat mengevaluasi seberapa baik model bekerja dengan mengukur akurasinya. Dan akurasi akan didefinisikan sebagai persentase kasus yang diklasifikasikan dengan benar. Kita dapat menganalisis kesalahan yang dibuat oleh model, atau tingkat kebingungannya, menggunakan matriks kebingungan (*confusion matrix*). Matriks kebingungan mengacu pada kebingungan dalam model, tetapi matriks kebingungan ini bisa menjadi sedikit sulit untuk dipahami ketika mereka menjadi sangat besar.

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

Gambar 2.174 contoh confusion matrix

- Jelaskan bagaimana cara membuat Confusion Matrix, Buat confusion matrix

Confusion matrix juga sering disebut error matrix. Pada umumnya confusion matrix memberikan informasi perbandingan hasil klasifikasi yang dilakukan oleh sistem (model) dengan hasil klasifikasi sebenarnya. Confusion matrix berbentuk tabel matriks yang menggambarkan kinerja model klasifikasi pada serangkaian data uji yang nilai sebenarnya diketahui. Untuk menggunakan Confusion Matrix, ada 4 istilah sebagai hasil proses dari klasifikasi. Diantaranya adalah :

- True Positive : Data positif yang terdeteksi memiliki hasil benar

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

Confusion Matrix

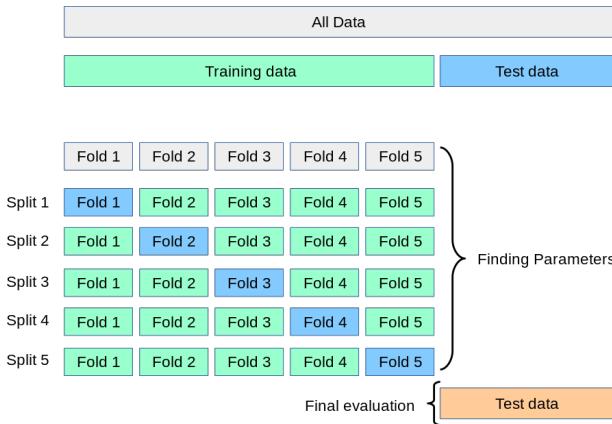
	Actually Positive (1)	Actually Negative (0)
Predicted Positive (1)	True Positives (TPs)	False Positives (FPs)
Predicted Negative (0)	False Negatives (FNs)	True Negatives (TNs)

Gambar 2.175 contoh confusion matrix

5. Jelaskan bagaimana K-fold cross validation bekerja dengan gambar ilustrasi contoh

Cara kerja k-fold validation:

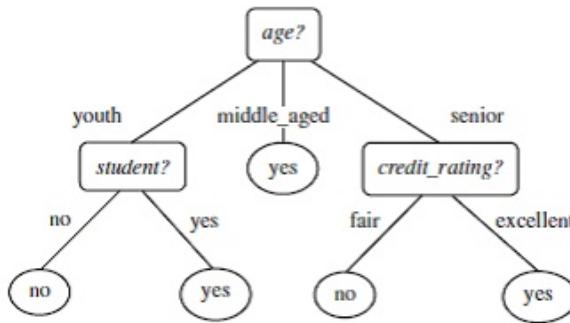
- Total instance dibagi menjadi N bagian.
- Fold yang pertama adalah bagian pertama menjadi data uji (testing data) dan sisanya menjadi training data.
- Lalu hitung akurasi berdasarkan porsi data tersebut dengan menggunakan persamaan.
- Fold yang kedua adalah bagian kedua, yang menjadi data uji(testing data)dan sisanya training data.
- Kemudian hitung akurasi berdasarkan porsi data tersebut.
- Dan seterusnya hingga habis mencapai fold ke-K.
- Terakhir hitung rata-rata akurasi K buah.



Gambar 2.176 contoh K-Fold Validation

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

Decision tree merupakan model prediksi menggunakan struktur pohon atau struktur berhirarki. Hasil dari setiap struktur biasanya menggunakan jawaban (True dan False) atau cabang lain yang akan menjadi pohon selanjutnya. Setiap keputusan diantaranya akan membandingkan kondisi yang diberikan kepada struktur untuk dibandingkan kondisi apa saja yang sudah didapat pada sistem tersebut.



Gambar 2.177 Decision Tree

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

- (a) information Gain

Information Gain adalah teknik seleksi fitur dengan cara memakai metode scoring untuk nominal ataupun pembobotan atribut kontinu yang didiskretkan menggunakan maksimal entropy. Suatu en-

tropy digunakan untuk mendefinisikan nilai Information Gain. Entropy menggambarkan banyaknya informasi yang dibutuhkan untuk mengkodekan suatu kelas

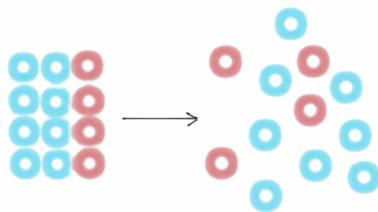
$$Gain(T, X) = Entropy(T) - Entropy(T, X)$$

$$\begin{aligned} G(\text{PlayGolf}, \text{Outlook}) &= E(\text{PlayGolf}) - E(\text{PlayGolf}, \text{Outlook}) \\ &= 0.940 - 0.693 = 0.247 \end{aligned}$$

Gambar 2.178 information gain

(b) Entropi

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



Gambar 2.179 penggambaran entropi

2.8.2 Praktek

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

```
2 1174084 % 3
3 #Hasilnya = 1 maka mengambil variable dengan nama kota
```

1. No. 1

```
1 # In[1]:
2 # load dataset (menggunakan student-mat)
3 import pandas as pd #Import library pandas menggantinya nama
4         yang akan dipanggil jadi pd
5 mataram = pd.read_csv('D:/ Git Kecerdasan Buatan/KB3C/src
6 /1174084/2/dataset/student-mat.csv', sep=';') #Membuat
7         variable tokyo yang isinya memanggil fungsi membaca file
8         csv
9 len(mataram) #Menghitung jumlah data yang ada pada csv yang
10        tadi sudah dibaca
```

```
In [13]: import pandas as pd
...: mataram = pd.read_csv('D:/New folder/KB3C/src/1174084/2/dataset/student-
mat.csv', sep=';')
...: len(mataram)
Out[13]: 395
```

Gambar 2.180 Loading Dataset

2. No. 2

```
1 # In[2]:
2 # generate binary label (pass/fail) based on G1+G2+G3 (test
3         grades, each 0–20 pts); threshold for passing is sum>=30
4 mataram['pass'] = mataram.apply(lambda row: 1 if (row['G1']+
5         row['G2']+row['G3']) >= 35 else 0, axis=1) #Membuat label
6         binary (pass/fail) berdasarkan G1+G2+G3 (testgrade,
7         semuanya 0–20 point); Batas untuk pass adalah sum>=30
8 mataram = mataram.drop(['G1', 'G2', 'G3'], axis=1) #
9         Meghilangkan data G1 G2 dan G3
10 mataram.head() #Menampilkan data
```

```
In [14]: mataram['pass'] = mataram.apply(lambda row: 1 if (row['G1']+row['G2']+row['G3'])
...: >= 35 else 0, axis=1)
...: mataram = mataram.drop(['G1', 'G2', 'G3'], axis=1)
...: mataram.head()
Out[14]:
   school sex age address famsize ... Dalc Walc health absences pass
0      GP   F  18       U    GT3 ...   1     1     3      6    0
1      GP   F  17       U    GT3 ...   1     1     3      4    0
2      GP   F  15       U    LE3 ...   2     3     3     10    0
3      GP   F  15       U    GT3 ...   1     1     5      2    1
4      GP   F  16       U    GT3 ...   1     2     5      4    0
[5 rows x 31 columns]
```

Gambar 2.181 Generate Binary Label

3. No. 3

```

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

```

```

In [15]: mataram = pd.get_dummies(mataram, columns=['sex', 'school', 'address',
...: 'famsize', 'Pstatus', 'Mjob', 'Fjob',
...: ...,
...: 'paid', 'activities',
...: ...,
...: mataram.head())
Out[15]:
   age  Medu  Fedu ...  internet_yes  romantic_no  romantic_yes
0    18      4     4 ...          0           1            0
1    17      1     1 ...          1           1            0
2    15      1     1 ...          1           1            0
3    15      4     2 ...          1           0            1
4    16      3     3 ...          0           1            0
[5 rows x 57 columns]

```

Gambar 2.182 One-hot Encoding

4. No. 4

```

1 # In [4]:
2 # shuffle rows
3 mataram = mataram.sample(frac=1) #Mengambil data sample dari
4 mataram
5 # split training and testing data
6 mataram_train = mataram[:500] #Membagi data untuk training
7 mataram_test = mataram[500:] #Membagi data untuk test
8 mataram_train_att = mataram_train.drop(['pass'], axis=1) #Meghapus data yang telah pass dan memasukkannya
9 mataram_train_pass = mataram_train['pass'] #Mengambil data yang pass saja
10 mataram_test_att = mataram_test.drop(['pass'], axis=1) #Meghapus data yang telah pass dan memasukkannya
11 mataram_test_pass = mataram_test['pass'] #Mengambil data yang pass saja
12 mataram_att = mataram.drop(['pass'], axis=1) #Meghapus data yang telah pass dan memasukkannya
13 mataram_pass = mataram['pass'] #Mengambil data yang pass saja
14 # number of passing students in whole dataset:
15 import numpy as np #Mengimport library numpy sebagai np
16 print("Passing: %d out of %d (%.2f%%)" % (np.sum(mataram_pass),
17 ), len(mataram_pass), 100*float(np.sum(mataram_pass)) / len(mataram_pass))) #Menampilkan data

```

```
In [16]: mataram = mataram.sample(frac=1)
...: # split training and testing data
...: mataram_train = mataram[:500]
...: mataram_test = mataram[500:]
...:
...: mataram_train_att = mataram_train.drop(['pass'], axis=1)
...: mataram_train_pass = mataram_train['pass']
...:
...: mataram_test_att = mataram_test.drop(['pass'], axis=1)
...: mataram_test_pass = mataram_test['pass']
...:
...: mataram_att = mataram.drop(['pass'], axis=1)
...: mataram_pass = mataram['pass']
...:
...: # number of passing students in whole dataset:
...: import numpy as np
...: print("Passing: %d out of %d (%.2f%%)" % (np.sum(mataram_pass),
len(mataram_pass), 100*float(np.sum(mataram_pass)) / len(mataram_pass)))
Passing: 166 out of 395 (42.03%)
```

Gambar 2.183 Shuffle Rows

5. No. 5

```
1 # In [5]:
2 # fit a decision tree
3 from sklearn import tree #import Decision tree dari library
sklearn
4 bandung = tree.DecisionTreeClassifier(criterion="entropy",
max_depth=5) #Membuat decition tree dengan maximal
depthnya 5
5 bandung = bandung.fit(mataram_train_att, mataram_train_pass)##
Memasukkan data yang akan dijadikan decition treenya
```

```
In [17]: from sklearn import tree
...: bandung = tree.DecisionTreeClassifier(criterion="entropy", max_depth=5)
...: bandung = bandung.fit(mataram_train_att, mataram_train_pass)
```

Gambar 2.184 Fit Decision tree

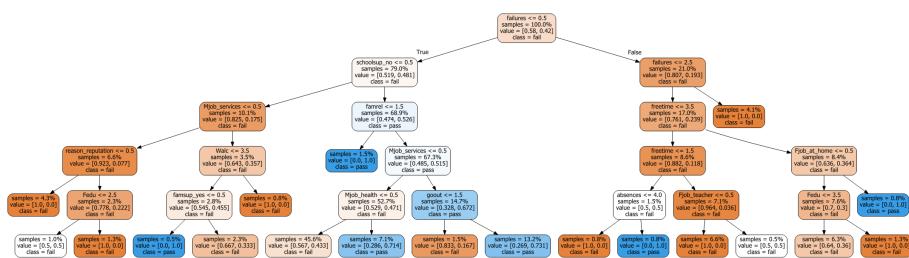
6. No. 6

```
1 # In [6]:
2 # visualize tree
3 import os
4 os.environ["PATH"] += os.pathsep + 'C:/ Users/rezas/Anaconda3/
Library/bin/graphviz/'
5 import graphviz #Mengimport Library Graphviz untuk
memvisualisasikan decision tree
6 malang = tree.export_graphviz(bandung, out_file=None, label="all",
impurity=False, proportion=True,
feature_names=list(
mataram_train_att), class_names=["fail", "pass"],
filled=True, rounded=True) #
Mendefinisikan dot_data yang isikan akan berisikan data
yang akan dijadikan gambar
```

```

9 jogja = graphviz.Source(malang) #Memasukkan data tadi menjadi sebuah jogja
10 jogja #Menampilkan jogja menggunakan graphviz

```



Gambar 2.185 Visualize tree

7. No. 7

```

1 # In[7]:
2 # save tree
3 tree.export_graphviz(bandung, out_file="1174084.dot", label="all",
4                      impurity=False, proportion=True,
5                      feature_names=list(mataram_train_att),
6                      class_names=["fail", "pass"],
7                      filled=True, rounded=True) #Digunakan
     untuk mengexport graph tree tadi yang telah kita buat

```

```

In [20]: tree.export_graphviz(bandung, out_file="student-
         performance.dot", label="all", impurity=False, proportion=True,
         ...:                                     feature_names=list(mataram_train_att),
         ...:                                     class_names=["fail", "pass"],
         ...:                                     filled=True, rounded=True)

```

Gambar 2.186 menyimpan(save) tree

8. No. 8

```

1 # In[8]:
2
3 bandung.score(mataram_test_att, mataram_test_pass) # Menghitung prediksi nilai yang akan datang dimasa depan

```

9. No. 9

```

1 # In[9]:
2 from sklearn.model_selection import cross_val_score #
     Mengimport fungsi cross_val_score dari library sklearn
3 makassar = cross_val_score(bandung, mataram_att, mataram_pass
     , cv=5) #Mendefinisikan nagoya yang isinya pembagian data
     menjadi 5

```

```

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

```

```

In [22]: from sklearn.model_selection import cross_val_score
...: makasar = cross_val_score(bandung, mataram_att,
mataram_pass, cv=5)
...: # show average score and +/- two standard deviations away
(covering 95% of scores)
...: print("Accuracy: %0.2f (+/- %0.2f)" % (makasar.mean(),
makasar.std() * 2))
Accuracy: 0.55 (+/- 0.10)

```

Gambar 2.187 Cross Val Score

10. No. 10

```

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

```

```
In [23]: for max_depth in range(1, 20):
    ...
    bandung =
tree.DecisionTreeClassifier(criterion="entropy",
max_depth=max_depth)
    ...
    makasar = cross_val_score(bandung, mataram_att,
mataram_pass, cv=5)
    ...
    print("Max depth: %d, Accuracy: %0.2f (+/- %0.2f)" %
(max_depth, makasar.mean(), makasar.std() * 2))
Max depth: 1, Accuracy: 0.58 (+/- 0.01)
Max depth: 2, Accuracy: 0.58 (+/- 0.10)
Max depth: 3, Accuracy: 0.56 (+/- 0.07)
Max depth: 4, Accuracy: 0.55 (+/- 0.11)
Max depth: 5, Accuracy: 0.55 (+/- 0.09)
Max depth: 6, Accuracy: 0.58 (+/- 0.11)
Max depth: 7, Accuracy: 0.56 (+/- 0.12)
Max depth: 8, Accuracy: 0.56 (+/- 0.08)
Max depth: 9, Accuracy: 0.56 (+/- 0.06)
Max depth: 10, Accuracy: 0.58 (+/- 0.09)
Max depth: 11, Accuracy: 0.53 (+/- 0.04)
Max depth: 12, Accuracy: 0.54 (+/- 0.05)
Max depth: 13, Accuracy: 0.57 (+/- 0.04)
Max depth: 14, Accuracy: 0.55 (+/- 0.07)
Max depth: 15, Accuracy: 0.57 (+/- 0.07)
Max depth: 16, Accuracy: 0.57 (+/- 0.02)
Max depth: 17, Accuracy: 0.57 (+/- 0.05)
Max depth: 18, Accuracy: 0.56 (+/- 0.10)
Max depth: 19, Accuracy: 0.56 (+/- 0.04)
```

Gambar 2.188 Max Depth

11. No. 11

```
1 # In[11]:
2 depth_acc = np.empty((19,3), float) #Membuat array baru
3 i = 0 #Membuat variable berisikan 0
4 for max_depth in range(1, 20): #Perulangan untuk memasukkan
    data
    bandung = tree.DecisionTreeClassifier(criterion="entropy"
        , max_depth=max_depth) #Membuat decision Tree
    makassar = cross_val_score(bandung, mataram_att,
        mataram_pass, cv=5) #Mendefinisikan nagoya yang isinya
        pembagian data menjadi 5
    depth_acc[i,0] = max_depth#Memasukkan data max_depth ke
    array depth_acc
    depth_acc[i,1] = makassar.mean() #Memasukkan data rata-
        rata dari nagoya ke array depth_acc
    depth_acc[i,2] = makassar.std() * 2 #Memasukkan data akar
        2 dari nagoya ke array depth_acc
    i += 1
11 depth_acc
```

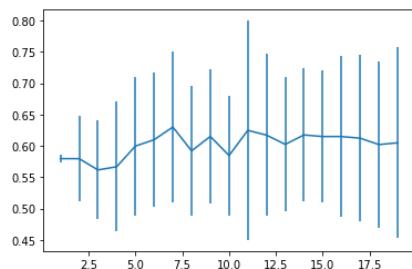
```
In [25]: depth_acc = np.empty((19,3), float)
...: i = 0
...: for max_depth in range(1, 20):
...:     bandung = tree.DecisionTreeClassifier(criterion="entropy",
max_depth=max_depth)
...:     makasar = cross_val_score(bandung, mataram_att, mataram_pass,
cv=5)
...:     depth_acc[i,0] = max_depth
...:     depth_acc[i,1] = makasar.mean()
...:     depth_acc[i,2] = makasar.std() * 2
...:     i += 1
...:
...: depth_acc
Out[25]:
array([[1.00000000e+00, 5.79751704e-01, 6.30768599e-03],
[2.00000000e+00, 5.79750893e-01, 9.75940982e-02],
[3.00000000e+00, 5.62222493e-01, 7.49317143e-02],
[4.00000000e+00, 5.51935248e-01, 1.06141842e-01],
[5.00000000e+00, 5.52064265e-01, 9.79558598e-02],
[6.00000000e+00, 5.72282538e-01, 8.64707915e-02],
[7.00000000e+00, 5.59656767e-01, 1.15237294e-01],
[8.00000000e+00, 5.41837877e-01, 8.49788036e-02],
[9.00000000e+00, 5.67186790e-01, 7.09553477e-02],
[1.00000000e+01, 5.84845829e-01, 6.13459259e-02],
[1.10000000e+01, 5.36838689e-01, 8.22803695e-02],
[1.20000000e+01, 5.69847452e-01, 7.22967003e-02],
[1.30000000e+01, 5.62027751e-01, 4.03465703e-02],
[1.40000000e+01, 5.44465271e-01, 7.01962406e-02],
[1.50000000e+01, 5.61901980e-01, 5.54653992e-02],
[1.60000000e+01, 5.69464460e-01, 4.20869065e-02],
[1.70000000e+01, 5.69336254e-01, 8.26363302e-02],
[1.80000000e+01, 5.57027751e-01, 8.58325793e-02],
[1.90000000e+01, 5.59496917e-01, 2.84251709e-02]])
```

Gambar 2.189 Depth in Range

12. No. 12

```
1 # In[12]:
2 import matplotlib.pyplot as plt #Menimport fungsi pyplot dari
   library matplotlib sebagai plt
3
4 solo, denpasar = plt.subplots() #Membuat plot baru
5 denpasar.errorbar(depth_acc[:,0], depth_acc[:,1], yerr=
   depth_acc[:,2]) #Mengisikan data plot
6 plt.show() #Menampilkan plot
```

```
In [13]: import matplotlib.pyplot as plt
...: solo, denpasar = plt.subplots()
...: denpasar.errorbar(depth_acc[:,0], depth_acc[:,1],
yerr=depth_acc[:,2])
...: plt.show()
```



Gambar 2.190 Matplotlib

2.8.3 Penanganan Error

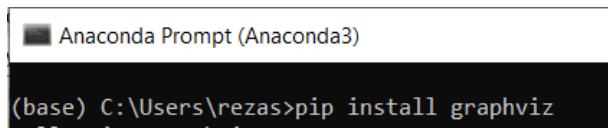
1. Error

```
File "<ipython-input-8-d311926e56ab>", line 3, in <module>
    import graphviz

ModuleNotFoundError: No module named 'graphviz'
```

Gambar 2.191 No module named graphviz

2. Solusi:



Gambar 2.192 Solusi dengan anaconda prompt

2.8.4 Bukti Tidak Plagiat



Gambar 2.193 plagirism

2.8.5 Link Video Youtube

<https://youtu.be/7gIPgpQA1LA>

2.9 1174073 - Ainul Filiani

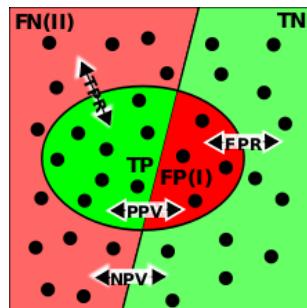
2.9.1 Teori

1. Binary Classification

masalah klasifikasi biner praktis, kedua kelompok tidak simetris - dari pada akurasi keseluruhan, proporsi relatif dari berbagai jenis kesalahan yang menarik. Misalnya, dalam pengujian medis, false positive (mendeteksi penyakit ketika tidak ada) dianggap berbeda daripada false negative.

Binary Classification digunakan untuk tujuan praktis dalam banyak masalah klasifikasi biner, dan kedua kelompok tersebut tidak simetris daripada akurasi secara keseluruhan, proporsi relatif dari berbagai macam kesalahan yang menarik. Contohnya, dalam pengujian medis tadi, false positif maksudnya mendeteksi penyakit ketika ada sedangkan untuk false negatif artinya tidak mendeteksi penyakit ketika ada.

Ada banyak metrik yang bisa digunakan dalam mengukur kinerja klasifikasi dan prediksi. misalnya dapat dilihat pada gambar berikut:



Gambar 2.194 Binary Classification

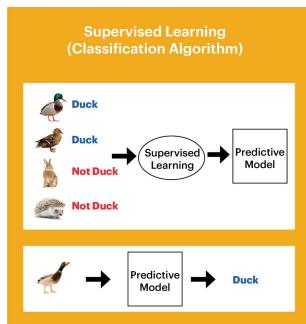
Bagian kiri dan kanan masing-masing memiliki instance yang sebenarnya ada dan tidak memiliki kondisi. Sedangkan bentuk oval tersebut berisi instance yang diklasifikasikan atau diprediksi sebagai positif atau negatif.

2. Supervised Learning, Unsupervised Learning, dan Clustering

- Supervised Learning

Dalam Supervised Learning, suatu program komputer diberikan dataset

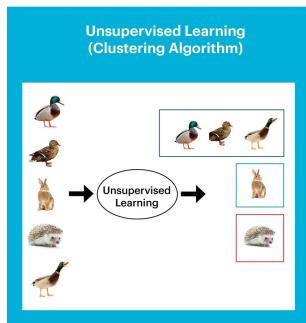
pelatihan yang kemudian diberi label dengan nilai output yang sesuai, dan fungsi tersebut akan ditentukan berdasarkan pada dataset. Fungsi atau algoritma tersebut kemudian akan digunakan untuk mengklasifikasi data baru untuk memprediksi nilai-nilai output yang sesuai dengan asumsi bahwa data baru sesuai dengan aturan dan fungsi yang digunakan. Berikut adalah contoh supervised learning:



Gambar 2.195 Supervised Learning

▪ Unsupervised Learning

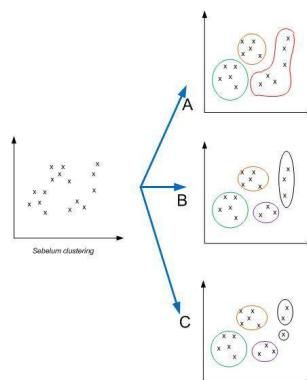
Unsupervised learning adalah istilah yang digunakan untuk pembelajaran bahasa Ibrani, yang terkait dengan pembelajaran tanpa guru, juga dikenal sebagai organisasi mandiri dan metode pemodelan kepadatan probabilitas input. Analisis cluster sebagai cabang pembelajaran mesin yang mengelompokkan data yang belum diberi label, diklasifikasikan atau dikategorikan. Alih-alih menanggapi umpan balik, analisis klaster mengidentifikasi kesamaan dalam data dan bereaksi berdasarkan ada tidaknya kesamaan di setiap potongan data baru. Berikut adalah contoh Unsupervised Learning:



Gambar 2.196 Unsupervised Learning

- Clustering

Cluster analysis or clustering adalah tugas pengelompokan sekumpulan objek sedemikian rupa sehingga objek dalam kelompok yang sama (disebut klaster) lebih mirip (dalam beberapa hal) satu sama lain daripada pada kelompok lain (kluster). Ini adalah tugas utama penambangan data eksplorasi, dan teknik umumuntukanalisisdatastastistik,yangdigunakandibanyakbidang,termasuk pembelajaranmesin,pengetahuanbioinformatika, kompresi data, dan grafik komputer. Analisis Cluster sendiri bukan merupakan salah satu algoritma spesifik, tetapi tugas umum yang harus dipecahkan. Ini dapat dicapai dengan berbagai algoritma yang berbeda secara signifikan dalam pemahaman mereka tentang apa yang merupakan sebuah cluster dan bagaimana cara menemukannya secara efisien. Gagasan populer mengenai cluster termasuk kelompok dengan jarak kecil antara anggota cluster, area padat ruang data, interval atau distribusi statistik tertentu. Clustering karena itu dapat dirumuskan sebagai masalah optimasi multi-objektif. Algoritma pengelompokan dan pengaturan parameter yang sesuai (termasuk parameter fungsi jarak yang akan digunakan, ambang kepadatan atau jumlah cluster yang diharapkan) tergantung pada set data individual dan penggunaan hasil yang dimaksudkan. Analisis kluster bukan merupakan tugas otomatis, tetapi proses berulang penemuan pengetahuan atau optimasi multi-objektif interaktif yang melibatkan percobaan dan kegagalan. Seringkali diperlukan untuk memodifikasi praproses data dan parameter model hingga hasilnya mencapai properti yang diinginkan. Contohnya, Bisa kita lihat bagaimana sebuah teknik clustering bisa mengelompokkan data ke dalam beberapa kluster.



Gambar 2.197 Clustering

3. Evaluasi dan Akurasi

Evaluasi merupakan suatu cara atau teknik dalam mengevaluasi seberapa baik model bekerja dengan mengukur akurasinya. Sedangkan akurasi

adalah suatu persentase kasus yang diklasifikasikan dengan benar. Kita bisa menganalisis suatu kesalahan yang dibuat dengan model atau tingkat confusion dengan menggunakan matriks confusion. Berikut adalah contoh klasifikasi biner yang menunjukkan berapa kali model telah membuat prediksi yang benar dari objek.

	Predicted "apple"	Predicted "orange"
True "apple"	20	5
True "orange"	3	22

Gambar 2.198 Evaluasi

Dalam tabel diatas, baris True Apple dan True Orange mengacu pada suatu kasus dimana objek itu sebenarnya sebuah apel atau sebenarnya jeruk. Kolom merujuk pada prediksi yang dibuat oleh model. Kita melihat bahwa ada 20 apel yang diprediksi dengan benar, sementara ada 5 apel yang salah diidentifikasi sebagai jeruk. Sehingga, matriks confusion harus memiliki semua nol, kecuali untuk diagonal sehingga kita dapat menghitung akurasi dengan menambahkan angka secara diagonal seperti pada gambar berikut :

$$\text{Accuracy} = (20 + 22) / (20 + 5 + 3 + 22) = 84$$

Gambar 2.199 Akurasi

4. Cara Membuat dan Membaca Confusion Matrix

Confusion Matrix adalah suatu matrix yang memberikan informasi perbandingan hasil klasifikasi yang dilakukan pada sistem atau model dengan hasil klasifikasi sebenarnya. Confusion Matrix berbentuk tabel matriks yang menggambarkan suatu kinerja model klasifikasi dari serangkaian data uji yang nilai sebenarnya diketahui.

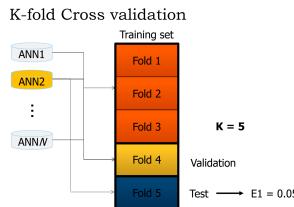
Berikut cara membuat dan membaca confusion matrix : pertama, tentukan terlebih dahulu pokok permasalahan dan atributnya. kedua, buatlah pohon keputusan dan data testingnya. ketiga, carilah nilai a,b,c dan d. Selanjutnya, cari nilai recall, precesion, accuracy serta seeror rate. Berikut contoh Confusion Matrix :

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

Gambar 2.200 Confusion Matrix

5. Cara Kerja K-fold Cross Validation

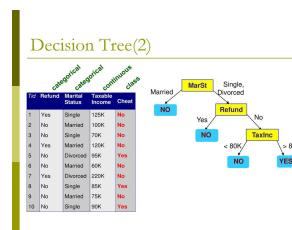
- Pertama, total instance bagi menjadi N bagian
- Fold pertama merupakan bagian pertama yang menjadi data uji atau testing data dan sisanya menjadi training data
- Kemudian, hitung akurasi dari porsi data dengan menggunakan persamaan
- Fold kedua, adalah bagian kedua dengan menjadi data uji atau testing data dan sisanya merupakan training data
- Lalu, hitung akurasi dari porsi data tersebut
- Lakukan hal tersebut, sampai habis mencapai fold ke-K
- Setelah itu, hitung rata-rata akurasi K



Gambar 2.201 K-fold Cross Validation

6. Decision Tree

Decision Tree adalah salah satu model prediksi dengan menggunakan struktur pohon atau struktur yang berhierarki. Konsepnya adalah mengubah data menjadi decision tree dan beberapa aturan keputusan. Decision Tree memiliki manfaat yaitu membrekdown porsesi pengambilan keputusan yang kompleks menjadi lebih simpel sehingga pengambil keputusan akan lebih mudah dipahami dan diinterpretasikan solusi dari suatu permasalahan.

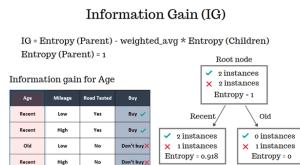


Gambar 2.202 Decision Tree

7. Information Gain dan Entropi

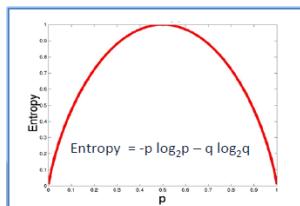
Information Gain adalah suatu teknik yang didasarkan pada penurunan

entropi setelah dataset dibagi pada atribut. Membangun keputusan adalah menemukan atribut yang mengembalikan perolehan informasi tertinggi.



Gambar 2.203 Information Gain

Entropi adalah suatu ukuran acak dalam informasi yang sedang diproses. Semakin tinggi suatu entropi, semakin sulit menarik kesimpulan dari informasi tersebut. Decision tree dibangun dari atas kebawah dan melibatkan partisi data kedalam himpunan bagian yang berisi instance dengan nilai yang sama. Algoritma ID3 menggunakan entropi untuk menghitung suatu homogenitas sampel.



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

Gambar 2.204 Entropy

2.9.2 Praktek

1. Nomor 1

```

1 # load dataset (menggunakan student-mat.csv)
2 import pandas as pd # mengimport library pandas sebagai pd
3 mangga = pd.read_csv('D:/kecerdasan buatan/Chapter 2/dataset/
                      student-mat.csv', sep=';') #variabel mangga berfungsi
                      untuk membaca atau read file student-mat.csv
4 len(mangga) #mengetahui jumlah baris pada data yang dipanggil

```

Hasilnya :

Gambar 2.205 Hasil Nomor 1

2. Nomor 2

```

1 # generate binary label (pass/fail) based on G1+G2+G3 ( test
  #   grades , each 0–20 pts); threshold for passing is sum>=30
2 mangga['pass'] = mangga.apply(lambda row: 1 if (row['G1']+row
  ['G2']+row['G3']) >= 35 else 0, axis=1) #mendeklarasikan
  # pass/fail nya data berdasarkan G1+G2+G3.
3 mangga = mangga.drop(['G1', 'G2', 'G3'], axis=1) #untuk
  # mengetahui baris G1+G2+G3 ditambahkan , dan hasilnya sama
  # dengan 35 maka axisnya 1.
4 mangga.head() #memanggil variabel mangga dimana ketentuan
  # head ini digunakan untuk mengembalikan baris n atas 5
  # secara default dari frame atau seri data

```

Hasilnya :

```

In [2]: mangga['pass'] = mangga.apply(lambda row: 1 if (row['G1']+row['G2']+row['G3']) >= 35 else 0, axis=1)
Out[2]: mangga = mangga.drop(['G1', 'G2', 'G3'], axis=1) #mendeklarasikan pass/fail nya data berdasarkan G1+G2+G3.
Out[3]: mangga.head() #memanggil variabel mangga dimana ketentuan head ini digunakan untuk mengembalikan baris
Out[4]: mangga
Out[4]:
   ... 1st 2nd address family ... daily absences pass
0   0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
1   0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
2   0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
3   0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
4   0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
[5 rows x 31 columns]

```

Gambar 2.206 Hasil Nomor 2

3. Nomor 3

```

1 # use one-hot encoding on categorical columns
2 mangga = pd.get_dummies(mangga, columns=['sex', 'school',
  'address', 'famsize', 'Pstatus', 'Mjob', 'Fjob',
  'reason', 'guardian', 'schoolsup',
  'famsup', 'paid', 'activities',
  'nursery', 'higher', 'internet',
  'romantic']) #variabel makassar dikonversi menjadi
  # bentuk yang lebih baik dalam prediksi dan memanggil
  # seluruh atribut
5 mangga.head() #memanggil variabel makassar dengan ketentuan
  # head ini digunakan untuk mengembalikan baris n atas 5
  # secara default dari frame atau seri data

```

Hasilnya :

```

In [5]: mangga = pd.get_dummies(mangga, columns=['sex', 'school', 'address', 'family', 'famsup', 'higher', 'internet', 'romantic'])
Out[5]: mangga
Out[5]:
   ... 1st 2nd address family ... Internet_no romanic_no
0   0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
1   0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
2   0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
3   0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
4   0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
[5 rows x 37 columns]

```

Gambar 2.207 Hasil Nomor 3

4. Nomor 4

```

1 # shuffle rows
2 mangga = mangga.sample(frac=1) #mengembalikan variabel mangga
  # menjadi sampel acak dengan frac=1

```

```

3 # split training and testing data
4 mangga_train = mangga[:500] #membuat variabel baru
    mangga_train
5 mangga_test = mangga[500:] #membuat variabel baru
    makassar_test yang sisa dari train
6
7 mangga_train_att = mangga_train.drop(['pass'], axis=1) #
    membuat variabel baru dengan ketentuan dari mangga_train
8 mangga_train_pass = mangga_train['pass'] #membuat variabel
    baru dengan ketentuan dari mangga_train
9
10 mangga_test_att = mangga_test.drop(['pass'], axis=1) #membuat
    variabel baru dengan ketentuan dari mangga_test
11 mangga_test_pass = mangga_test['pass'] #membuat variabel baru
    dengan ketentuan dari mangga_test
12
13 mangga_att = mangga.drop(['pass'], axis=1) #membuat variabel
    mangga_att sebagai salinan dari mangga
14 mangga_pass = mangga['pass'] #membuat variabel mangga_pass
    sebagai salinan dari mangga
15
16 # number of passing students in whole dataset:
17 import numpy as np #mengimport module numpy sebagai np y
18 print("Passing: %d out of %d (%.2f%%)" % (np.sum(mangga_pass)
    , len(mangga_pass), 100*float(np.sum(mangga_pass)) / len(
        mangga_pass))) #untuk mengembalikan nilai passing dari
    pelajar dari keseluruhan dataset dengan cara print.

```

Hasilnya :

```

In [5]: mangga = mangga.sample(frac=1) #mengacak data variabel mangga secara acak, kompat with pandas.DataFrame
... mangga_train = mangga[:500] #membuat variabel baru mangga_train
... mangga_test = mangga[500:] #membuat variabel baru mangga_test
... mangga_train_att = mangga_train.drop(['pass'], axis=1) #membuat variabel baru dengan ketentuan dari mangga_train
... mangga_train_pass = mangga_train['pass'] #membuat variabel baru dengan ketentuan dari mangga_train
... mangga_test_att = mangga_test.drop(['pass'], axis=1) #membuat variabel baru dengan ketentuan dari mangga_test
... mangga_test_pass = mangga_test['pass'] #membuat variabel baru dengan ketentuan dari mangga_test
... mangga_att = mangga.drop(['pass'], axis=1) #membuat variabel mangga_att sebagai salinan dari mangga
... mangga_pass = mangga['pass'] #membuat variabel mangga_pass sebagai salinan dari mangga
...
... print("Passing: %d out of %d (%.2f%%)" % (np.sum(mangga_pass), len(mangga_pass), 100*float(np.sum(mangga_pass)) / len(mangga_pass)))
...
... print("Passing: 200 out of 200 (100.00%)")

```

Gambar 2.208 Hasil Nomor 4

5. Nomor 5

```

1 # fit a decision tree
2 from sklearn import tree #import tree dari library sklearn
3 jeruk = tree.DecisionTreeClassifier(criterion="entropy",
    max_depth=5) #membuat variabel jeruk sebagai decisiontree ,
    dengan criterion fungsi mengukur kualitas split
4 jeruk = jeruk.fit(mangga_train_att, mangga_train_pass) #
    training varibael jeruk dengan data dari variabel makassar
.

```

Hasilnya :

```

In [5]: tree.DecisionTreeClassifier(criterion='entropy', max_depth=5)
... jeruk = tree.DecisionTreeClassifier(criterion='entropy', max_depth=5) #membuat variabel jeruk sebagai decisiontree,
... jeruk = jeruk.fit(mangga_train_att, mangga_train_pass) #training variabel jeruk dengan data dari variabel makassar
...

```

Gambar 2.209 Hasil Nomor 5

6. Nomor 6

```

1 # visualize tree
2 import os
3 os.environ["PATH"] += os.pathsep + 'C:/ Program Files (x86)/
   Graphviz2.38/bin/'
4 import graphviz #import library graphviz sebagai perangkat
   lunak visualisasi grafik open source
5 dot_data = tree.export_graphviz(jeruk, out_file=None, label="all",
   impurity=False, proportion=True,
   feature_names=list(
      mangga_train_att), class_names=["fail", "pass"],
   filled=True, rounded=True) #
   mengambil data untuk diterjemahkan ke grafik
8 graph = graphviz.Source(dot_data) #membuat variabel graph
   sebagai grafik yang di ambil dari dot_data
9 graph #memanggil graph

```

Hasilnya :



Gambar 2.210 Hasil Nomor 6

7. Nomor 7

```

1 # save tree
2 tree.export_graphviz(jeruk, out_file="student-performance.dot",
   label="all", impurity=False, proportion=True,
   feature_names=list(mangga_train_att),
   class_names=["fail", "pass"],
   filled=True, rounded=True) #save tree
   sebagai export graphviz ke file student-performance.dot

```

Hasilnya :

```
tree.export_graphviz(jeruk, out_file="student-performance.dot", label="all", impurity=False, proportion=True,
feature_names=list(mangga_train_att), class_names=["fail", "pass"], filled=True, rounded=True) #save tree
sebagai export graphviz ke file student-performance.dot
```

Gambar 2.211 Hasil Nomor 7

8. Nomor 8

```

1
2 jeruk.score(mangga_att, mangga_pass) #score juga disebut
   prediksi dengan diberi beberapa data input baru

```

Hasilnya :

```
[In [34]:] In[34]: from sklearn.metrics import accuracy_score, cross_val_score
          scores = cross_val_score(jeruk, mangga_att, mangga_pass, cv=5)
          print("Accuracy: %0.2f (+/- %0.2f)" % (scores.mean(), scores.std() * 2))
```

Gambar 2.212 Hasil Nomor 8

9. Nomor 9

```
[1]: from sklearn.model_selection import cross_val_score #import
       class cross_val_score dari sklearn
[2]: scores = cross_val_score(jeruk, mangga_att, mangga_pass, cv=5) #mengevaluasi score dengan validasi silang
[3]: # show average score and +/- two standard deviations away (covering 95% of scores)
[4]: print("Accuracy: %0.2f (+/- %0.2f)" % (scores.mean(), scores.std() * 2)) #print akurasi
```

Hasilnya :

```
In [34]: from sklearn.model_selection import cross_val_score
         scores = cross_val_score(jeruk, mangga_att, mangga_pass, cv=5)
         print("Accuracy: %0.2f (+/- %0.2f)" % (scores.mean(), scores.std() * 2))
Accuracy: 8.80 (+/- 4.80)
```

Gambar 2.213 Hasil Nomor 9

10. Nomor 10

```
[1]: for max_depth in range(1, 20):
[2]:     jeruk = tree.DecisionTreeClassifier(criterion="entropy",
[3]:                                         max_depth=max_depth)
[4]:     scores = cross_val_score(jeruk, mangga_att, mangga_pass,
[5]:                               cv=5)
[6]:     print("Max depth: %d, Accuracy: %0.2f (+/- %0.2f)" % (max_depth, scores.mean(), scores.std() * 2))
[7]: 
```

#Disini ini menunjukkan seberapa dalam di tree itu. Semakin dalam tree, semakin banyak perpecahan yang dimilikinya dan menangkap lebih banyak informasi tentang data.

Hasilnya :

```
In [35]: for max_depth in range(1, 20):
[1]:     jeruk = tree.DecisionTreeClassifier(criterion="entropy",
[2]:                                         max_depth=max_depth)
[3]:     scores = cross_val_score(jeruk, mangga_att, mangga_pass,
[4]:                               cv=5)
[5]:     print("Max depth: %d, Accuracy: %0.2f (+/- %0.2f)" % (max_depth, scores.mean(), scores.std() * 2))
[6]: 
```

Gambar 2.214 Hasil Nomor 10

11. Nomor 11

```

1 depth_acc = np.empty((19,3), float) #Dengan 19 sebagai bentuk
2         array kosong, 3 sebagai output data-type
3 Anggur = 0 #variabel Anggur sebagai array 0
4 for max_depth in range(1, 20): #perulangan dengan max_depth
5     bone = tree.DecisionTreeClassifier(criterion="entropy",
6             max_depth=max_depth) #variabel ular untuk decision tree
7             dengan ketentuan entropy
8             scores = cross_val_score(jeruk, mangga_att, mangga_pass,
9                 cv=5) #scores diambil dari data cross_val_score
10            depth_acc[Anggur,0] = max_depth #mengembalikan array
11            dengan ketentuan 0 dan max_depth
12            depth_acc[Anggur,1] = scores.mean() #mengembalikan array
13            dengan ketentuan 1 dan scores.mean
14            depth_acc[Anggur,2] = scores.std() * 2 #mengembalikan
15            array dengan ketentuan 2 dan scores.std, std berarti
16            menghitung standar deviasi
17            Anggur += 1
18
19 depth_acc #Depth acc akan membuat array kosong dengan
20         mengembalikan array baru dengan bentuk dan tipe yang
21         diberikan

```

Hasilnya :

```

[[{'id': 1, 'value': 0.5847517, 'error': 0.0224964},  
 {'id': 2, 'value': 0.5997507, 'error': 0.0307486},  
 {'id': 3, 'value': 0.5749185, 'error': 0.0709969},  
 {'id': 4, 'value': 0.5899754, 'error': 0.0773737},  
 {'id': 5, 'value': 0.5847517, 'error': 0.0715933},  
 {'id': 6, 'value': 0.5976318, 'error': 0.0519981},  
 {'id': 7, 'value': 0.5976318, 'error': 0.0547447},  
 {'id': 8, 'value': 0.58598974, 'error': 0.08627217},  
 {'id': 9, 'value': 0.58598974, 'error': 0.08627217},  
 {'id': 10, 'value': 0.55224733, 'error': 0.0527373},  
 {'id': 11, 'value': 0.68247485, 'error': 0.0483412},  
 {'id': 12, 'value': 0.57723253, 'error': 0.0421968},  
 {'id': 13, 'value': 0.59580395, 'error': 0.0741988},  
 {'id': 14, 'value': 0.5897795, 'error': 0.0645683},  
 {'id': 15, 'value': 0.58731581, 'error': 0.05961648},  
 {'id': 16, 'value': 0.60000093, 'error': 0.0511427},  
 {'id': 17, 'value': 0.5887795, 'error': 0.0645683},  
 {'id': 18, 'value': 0.57225414, 'error': 0.06204872}]]
```

Gambar 2.215 Hasil Nomor 11

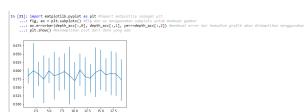
12. Nomor 12

```

1 import matplotlib.pyplot as plt #import matplotlib sebagai
2         plt
3 fig, ax = plt.subplots() #fig dan ax menggunakan subplots
4         untuk membuat gambar
5 ax.errorbar(depth_acc[:,0], depth_acc[:,1], yerr=depth_acc
6             [:,2]) #membuat error bar kemudian grafik akan ditampilkan
7             menggunakan show
8 plt.show() #menampilkan plot dari data yang ada

```

Hasilnya :



Gambar 2.216 Hasil Nomor 12

2.9.3 Penanganan Error

1. ScreenShoot Error

```
File "cipython-input-7-f751199220fa", line 1, in <module>
    import graphviz #import library graphviz sebagai perangkat
lunak visualisasi grafik open source
ModuleNotFoundError: No module named 'graphviz'
```

Gambar 2.217 Module Not Found Error

```
FileNotFoundException: File b'dataset/student-mat.csv' does not exist
```

Gambar 2.218 File Not Found Error

```
File "C:\ProgramData\Anaconda3\lib\site-packages\graphviz
\backend.py", line 162, in run
    raise ExecutableNotFoundError(cmd)
ExecutableNotFoundError: failed to execute ['dot', '-Tsvg'], make sure
the Graphviz executables are on your systems' PATH
Out[8]: <graphviz.files.Source at 0x2595e5d0780>
```

Gambar 2.219 Executable Not Found

2. Tuliskan Kode Error dan Jenis Error

- Module Not Found Error
- File Not Found Error
- Executable Not Found

3. Cara Penanganan Error

- Module Not Found Error

Dengan memperbaiki penulisan atau kesalahan dalam penulisan kode atau melakukan install package atau modul yang belum terinstal

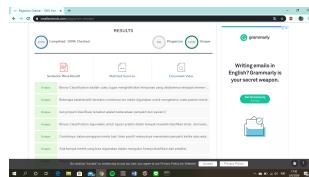
- File Not Found Error

Dengan memperbaiki directory file. Sesuaikan sama tempat penyimpanan di laptop

- Executable Not Found

Dengan menginstal aplikasi graphviz di windows dan menambahkan directory diatas kode program

2.9.4 Bukti Tidak Plagiat



Gambar 2.220 Bukti Tidak Plagiat

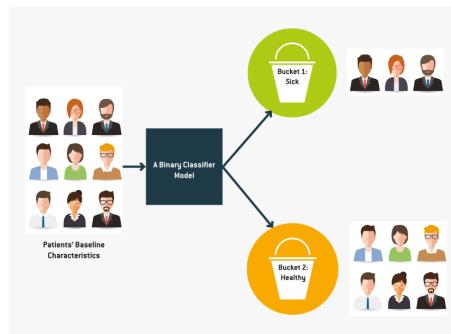
2.9.5 Link Youtube

2.10 1174086 - Tia Nur Candida

2.10.1 Teori

1. Binary Classification

Binary Classification, yang berarti mengklasifikasikan objek dari suatu himpunan menjadi dua kelompok, tetapi teknik ada untuk klasifikasi multikelas.



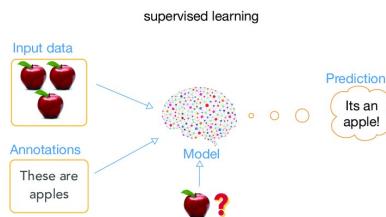
Gambar 2.221 Binary Classification

2. Supervised Learning, Unsupervised Learning dan Clustering

- Supervised Learning

supervised learning mempunyai input dan output yang dapat dibuat menjadi suatu model hubungan matematis sehingga mampu melakukan prediksi dan klasifikasi berdasarkan data yang telah ada sebelumnya. supervised learning membutuhkan data training agar mampu

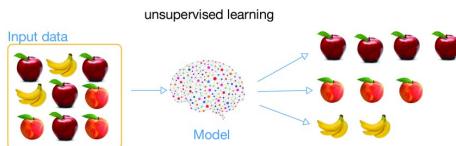
melakukan prediksi maupun klasifikasi. supervised learning, algoritma tersebut seolah-olah dilatih terlebih dahulu agar dapat melakukan prediksi maupun klasifikasi.



Gambar 2.222 Supervised Learning

- **Unsupervised Learning**

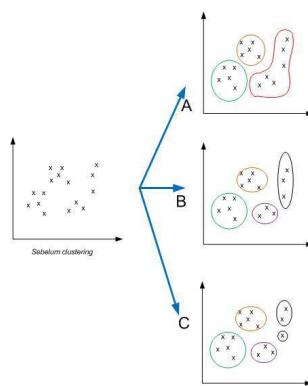
Unsupervised learning tidak menggunakan data latih atau data training untuk melakukan prediksi maupun klasifikasi. Berdasarkan model matematisnya, algoritma ini tidak memiliki target variabel. Salah satu tujuan dari algoritma ini adalah mengelompokkan objek yang hampir sama dalam suatu area tertentu.



Gambar 2.223 Unsupervised Learning

- **Clustering**

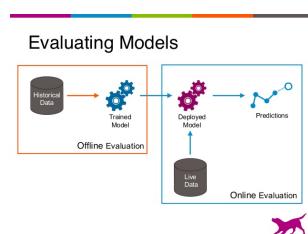
Teknik clustering ini masuk ke dalam kelompok unsupervised learning, yang artinya ini merupakan teknik di mana mesin akan bekerja (belajar) sendiri tanpa kita ajari bagaimana cara memecahkan permasalahannya. Sebagai contoh, anggap saja kita memiliki sebuah data, misal data pelanggan yang berisi tentang jenis kelamin, besarnya penghasilan dan besarnya pembelian produk-produk kita. Maka dengan algoritma clustering kita dapat mengetahui pelanggan kita akan dikelompokkan ke dalam beberapa kluster dengan sendirinya, misal ada pelanggan yang pelit, pelanggan yang royal dan lain sebagainya.

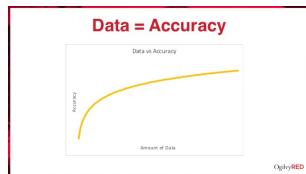
**Gambar 2.224** Clustering

Ada 3 hasil klustering, yaitu A (menjadi 3 kluster), B (menjadi 4 kluster) dan C (menjadi 5 kluster). Manakah yang paling baik pembagiannya? Apakah A, B atau C? Itu semua tergantung kita sebagai pembuat algoritmanya. Jika kita menginginkan 3 kluster saja, maka mungkin A yang terbaik. Jika kita ingin 5 dan sangat detail, maka C yang terbaik. Itupun juga tergantung bagaimana data yang kita miliki.

3. Evaluasi dan Akurasi

Evaluasi adalah kegiatan yang dilakukan berkenaan dengan proses untuk menentukan nilai dari suatu hal. Kita dapat mengevaluasi seberapa baik model bekerja dengan mengukur akurasinya. Akurasi akan didefinisikan sebagai persentase kasus yang diklasifikasikan dengan benar.

**Gambar 2.225** Evaluation



Gambar 2.226 Accuracy

4. Confusion Matrix

Confusion matrix juga sering disebut error matrix. Pada dasarnya confusion matrix memberikan informasi perbandingan hasil klasifikasi yang dilakukan oleh sistem (model) dengan hasil klasifikasi sebenarnya. Confusion matrix berbentuk tabel matriks yang menggambarkan kinerja model klasifikasi pada serangkaian data uji yang nilai sebenarnya diketahui. Terdapat 4 istilah sebagai representasi hasil proses klasifikasi pada confusion matrix

- True Positive (TP) Merupakan data positif yang diprediksi benar. Contohnya, pasien menderita kanker (class 1) dan dari model yang dibuat memprediksi pasien tersebut menderita kanker (class 1).
- True Negative (TN) Merupakan data negatif yang diprediksi benar. Contohnya, pasien tidak menderita kanker (class 2) dan dari model yang dibuat memprediksi pasien tersebut tidak menderita kanker (class 2).
- False Postive (FP) — Type I Error Merupakan data negatif namun diprediksi sebagai data positif. Contohnya, pasien tidak menderita kanker (class 2) tetapi dari model yang telah memprediksi pasien tersebut menderita kanker (class 1).
- False Negative (FN) — Type II Error Merupakan data positif namun diprediksi sebagai data negatif. Contohnya, pasien menderita kanker (class 1) tetapi dari model yang dibuat memprediksi pasien tersebut tidak menderita kanker (class 2).

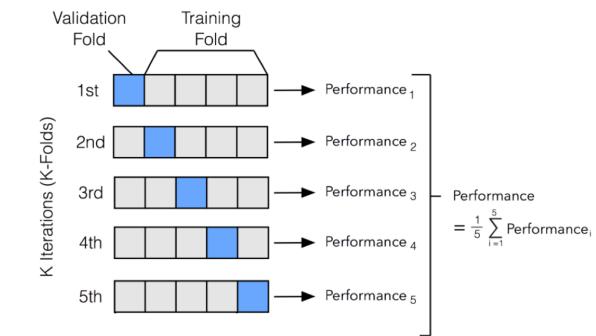
		Predicted Class		
		Positive	Negative	
Actual Class	Positive	True Positive (TP)	False Negative (FN) <i>Type II Error</i>	Sensitivity $\frac{TP}{(TP + FN)}$
	Negative	False Positive (FP) <i>Type I Error</i>	True Negative (TN)	Specificity $\frac{TN}{(TN + FP)}$
		Precision $\frac{TP}{(TP + FP)}$	Negative Predictive Value $\frac{TN}{(TN + FN)}$	Accuracy $\frac{TP + TN}{(TP + TN + FP + FN)}$

Gambar 2.227 Confusion Matrix

5. K-Fold

Cara kerja k-fold validation:

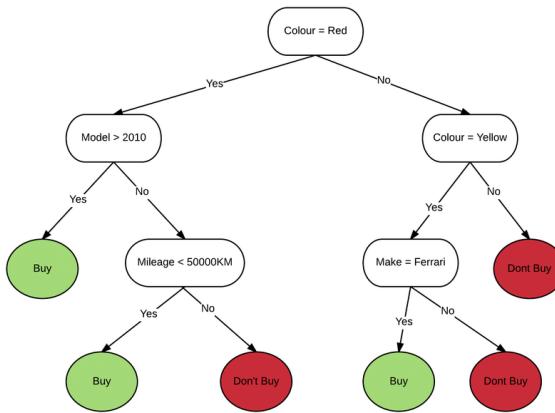
- Total instance dibagi menjadi N bagian.
- Fold yang pertama adalah bagian pertama menjadi data uji (testing data) dan sisanya menjadi training data.
- Lalu hitung akurasi berdasarkan porsi data tersebut dengan menggunakan persamaan.
- Fold yang kedua adalah bagian kedua, yang menjadi data uji(testing data)dan sisanya training data.
- Kemudian hitung akurasi berdasarkan porsi data tersebut.
- Dan seterusnya hingga habis mencapai fold ke-K.
- Terakhir hitung rata-rata akurasi K buah.



Gambar 2.228 contoh K-Fold Validation

6. Decision Tree

Decision tree adalah salah satu metode klasifikasi yang paling populer, karena mudah untuk diinterpretasi oleh manusia. Decision tree adalah model prediksi menggunakan struktur pohon atau struktur berhirarki. Konsep dari pohon keputusan adalah mengubah data menjadi decision tree dan aturan-aturan keputusan.



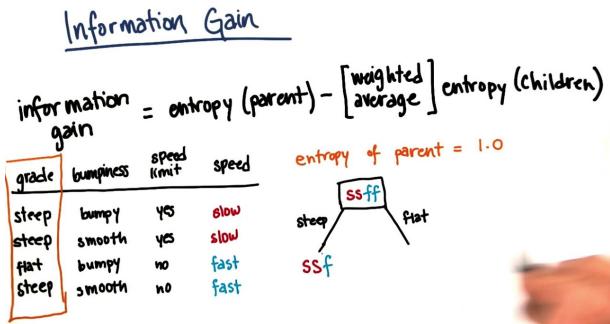
Gambar 2.229 Decision Tree

7. Entropy dan Information Gain

Algoritma pada metode ini menggunakan konsep dari entropi. Konsep Entropi yang digunakan untuk mengukur “seberapa informatifnya” sebuah node (yang biasanya disebut seberapa baiknya).

Entropi adalah nilai informasi yang menyatakan ukuran ketidakpastian (impurity) dari attribut dari suatu kumpulan obyek data dalam satuan bit.

Information Gain adalah ukuran efektifitas suatu atribut dalam mengklasifikasikan data. Digunakan untuk menentukan urutan atribut dimana atribut yang memiliki nilai Information Gain terbesar yang dipilih



Gambar 2.230 Information Gain

2.10.2 Praktek

```

1 # In [0]:
2
3 1174086 % 3
4 #Hasilnya = 0 maka mengambil variable dengan nama makanan
5
6 # In [1]:
7 # load dataset (menggunakan student-mat)
8 import pandas as pd
9 tahu = pd.read_csv('D://TI/SMT 6/AI/Chapter 2/KB3C-master - Copy/
10     src/1174086/2/dataset/student-mat.csv', sep=';')
11 len(tahu)
12
13 # In [2]:
14 # generate binary label (pass/fail) based on G1+G2+G3 (test
15 #   grades, each 0–20 pts); threshold for passing is sum>=30
16 tahu['pass'] = tahu.apply(lambda row: 1 if (row['G1']+row['G2']+
17     row['G3']) >= 35 else 0, axis=1)
18 tahu = tahu.drop(['G1', 'G2', 'G3'], axis=1)
19 tahu.head()
20
21 # In [3]:
22 # use one-hot encoding on categorical columns
23 tahu = pd.get_dummies(tahu, columns=['sex', 'school', 'address',
24     'famsize', 'Pstatus', 'Mjob', 'Fjob',
25     'reason', 'guardian', 'schoolsup',
26     'famsup', 'paid', 'activities',
27     'nursery', 'higher', 'internet',
28     'romantic'])
29 tahu.head()
30
31 # In [4]:
32 # shuffle rows
33 tahu = tahu.sample(frac=1)
34 # split training and testing data
  
```

```
29 tahu_train = tahu[:500]
30 tahu_test = tahu[500:]
31
32 tahu_train_att = tahu_train.drop(['pass'], axis=1)
33 tahu_train_pass = tahu_train['pass']
34
35 tahu_test_att = tahu_test.drop(['pass'], axis=1)
36 tahu_test_pass = tahu_test['pass']
37
38 tahu_att = tahu.drop(['pass'], axis=1)
39 tahu_pass = tahu['pass']
40
41 # number of passing students in whole dataset:
42 import numpy as np
43 print("Passing: %d out of %d (%.2f%%)" % (np.sum(tahu_pass), len(
        tahu_pass), 100*float(np.sum(tahu_pass)) / len(tahu_pass)))
44
45
46 # In [5]:
47 # fit a decision tree
48 from sklearn import tree
49 tempe = tree.DecisionTreeClassifier(criterion="entropy",
        max_depth=5)
50 tempe = tempe.fit(tahu_train_att, tahu_train_pass)
51
52 # In [6]:
53 # visualize tree
54 import os
55 os.environ["PATH"] += os.pathsep + 'C:/Users/Tia/Anaconda3/
    Library/bin/graphviz/'
56 import graphviz
57 oncom = tree.export_graphviz(tempe, out_file=None, label="all",
        impurity=False, proportion=True,
        feature_names=list(tahu_train_att),
        class_names=["fail", "pass"],
        filled=True, rounded=True)
58 makanan = graphviz.Source(oncom)
59 makanan
60
61
62
63 # In [7]:
64 # save tree
65 tree.export_graphviz(tempe, out_file="student-performance.dot",
        label="all", impurity=False, proportion=True,
        feature_names=list(tahu_train_att),
        class_names=["fail", "pass"],
        filled=True, rounded=True)
66
67
68
69 # In [8]:
70
71 tempe.score(tahu_test_att, tahu_test_pass)
72
73 # In [9]:
74 from sklearn.model_selection import cross_val_score
75 perkedel = cross_val_score(tempe, tahu_att, tahu_pass, cv=5)
76 # show average score and +/- two standard deviations away (
        covering 95% of scores)
```

```

77 print("Accuracy: %0.2f (+/- %0.2f)" % (perkedel.mean(), perkedel.
    std() * 2))
78
79
80 # In [10]:
81 for max_depth in range(1, 20):
82     tempe = tree.DecisionTreeClassifier(criterion="entropy",
83                                         max_depth=max_depth)
84     perkedel = cross_val_score(tempe, tahu_att, tahu_pass, cv=5)
85     print("Max depth: %d, Accuracy: %0.2f (+/- %0.2f)" % (
86         max_depth, perkedel.mean(), perkedel.std() * 2))
87
88 # In [11]:
89 depth_acc = np.empty((19,3), float)
90 i = 0
91 for max_depth in range(1, 20):
92     tempe = tree.DecisionTreeClassifier(criterion="entropy",
93                                         max_depth=max_depth)
94     perkedel = cross_val_score(tempe, tahu_att, tahu_pass, cv=5)
95     depth_acc[i,0] = max_depth
96     depth_acc[i,1] = perkedel.mean()
97     depth_acc[i,2] = perkedel.std() * 2
98     i += 1
99 depth_acc
100
101
102 # In [12]:
103 import matplotlib.pyplot as plt
104 fig, burger = plt.subplots()
105 burger.errorbar(depth_acc[:,0], depth_acc[:,1], yerr=depth_acc
106                  [:,2])
107 plt.show()

```

2.10.3 Penanganan Error

1. Screenshoot Error

```
#FileNotFoundError: [Errno 2] File b'student-mat.csv' does not
exist: b'student-mat.csv'
```

Gambar 2.231 File Not Found

```
File "<ipython-input-13-d089b825cled>", line 1, in <module>
    import graphviz
ModuleNotFoundError: No module named 'graphviz'
```

Gambar 2.232 Module Not Found

```
file "C:\Users\ENNO0\Anaconda3\lib\site-packages\graphviz\backend.py", line 162,
    import graphviz
    raise ExecutableNotFoundError()
ExecutableNotFoundError: failed to execute ['dot', '-Tsvg'], make sure the Graphviz
executables are on your systems' PATH
Out[16]: <graphviz.files.Source at 0x2ffadec000>
```

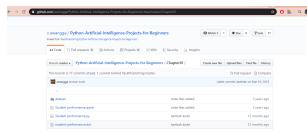
Gambar 2.233 Executable Not Found

2. Jenis Error

- File Not Found
- Module Not Found
- Executable Not Found

3. Solusi Error

- File Not Found
Mendownload filenya di github



Gambar 2.234 File Dataset

- Module Not Found

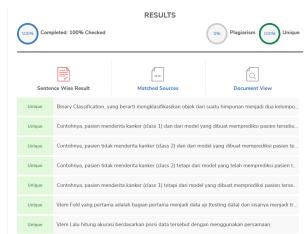
Mendownload library graphviz menggunakan pip install graphviz di Anaconda Prompt



Gambar 2.235 Install graphviz

- Executable Not Found

2.10.4 Bukti Tidak Plagiat



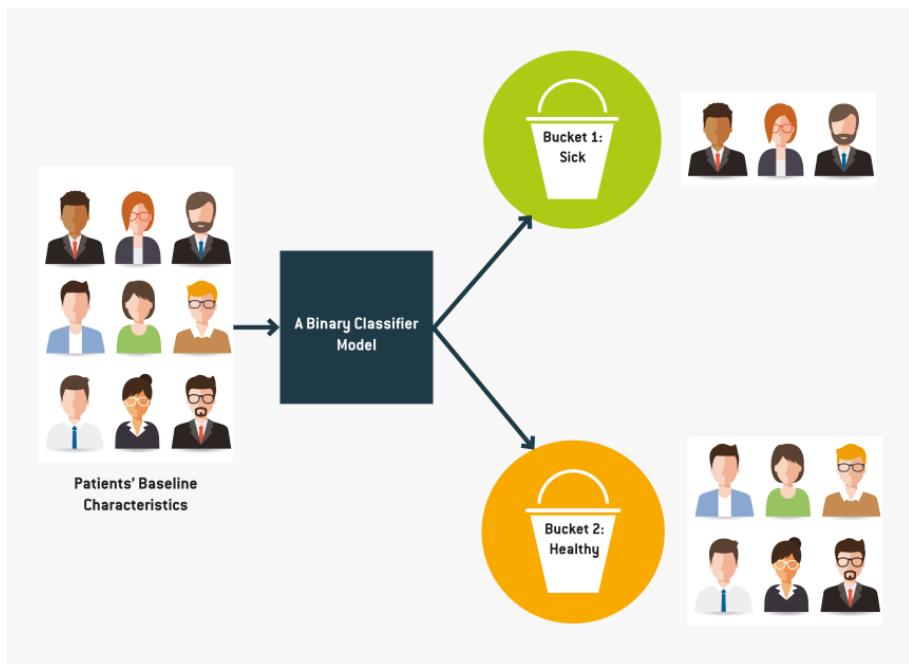
Gambar 2.236 Bukti Tidak Plagiat

2.11 1174079 - Chandra Kirana Poetra

Chapter 2 - Membangun model prediksi

2.11.1 Teori

2.11.1.1 Jelaskan Apa Itu Binary Classification dilengkapi ilustrasi gambar sendiri.



Gambar 2.237 Ilustrasi Binary Classification

Klasifikasi biner merupakan suatu cara kerja atau metode dalam menentukan atau mengelompokkan beberapa elemen atau data menjadi dua kelompok (Menentukan klasifikasi tiap data yang ada) berdasarkan aturan klasifikasi. Konteks yang membutuhkan suatu keputusan mengenai apakah suatu data atau item memiliki sifat kualitatif atau tidak, beberapa karakteristik tertentu yang unik, atau beberapa klasifikasi biner yang tipikal meliputi:

- Tes medis : digunakan untuk menentukan apakah seorang pasien memiliki penyakit tertentu atau tidak, bentuk klasifikasinya disini yaitu keberadaan penyakitnya itu sendiri
- Suatu metode "pass or fail" yang menentukan suatu spesifikasi apakah sudah memenuhi ketentuan atau belum.
- Pengambilan informasi : Menentukan apakah suatu halaman atau artikel berhak untuk tampil dalam suatu result set dari hasil pencarian atau

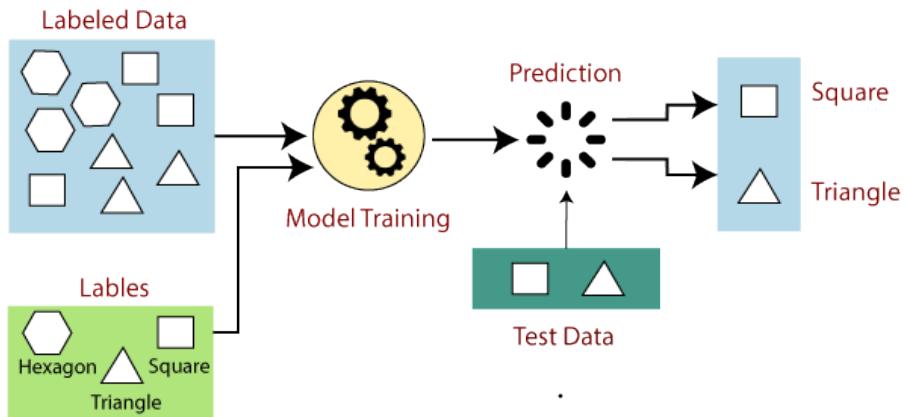
tidak, properti klasifikasi disini adalah relevansi artikelnya itu sendiri atau kegunaannya kepada para pembaca..

2.11.1.2 Jelaskan Apa itu supervised learning , unsupervised learning dan clustering dengan ilustrasi gambar sendiri.

1. supervised learning

Supervised learning merupakan suatu metode dalam bidang machine learning yang digunakan untuk melatih mesin menggunakan data yang sudah dilabeli. itu berarti beberapa data sudah dilabeli sebagai jawaban yang benar atau true. ini semua sama halnya seperti bagaimana seorang guru mengajarkan ilmu kepada para siswanya.

Algorithma Supervised learning belajar dari data training yang sudah dilabeli, yang kemudian akan membantu anda dalam memprediksi hasil yang akan terjadi dari data yang ada.

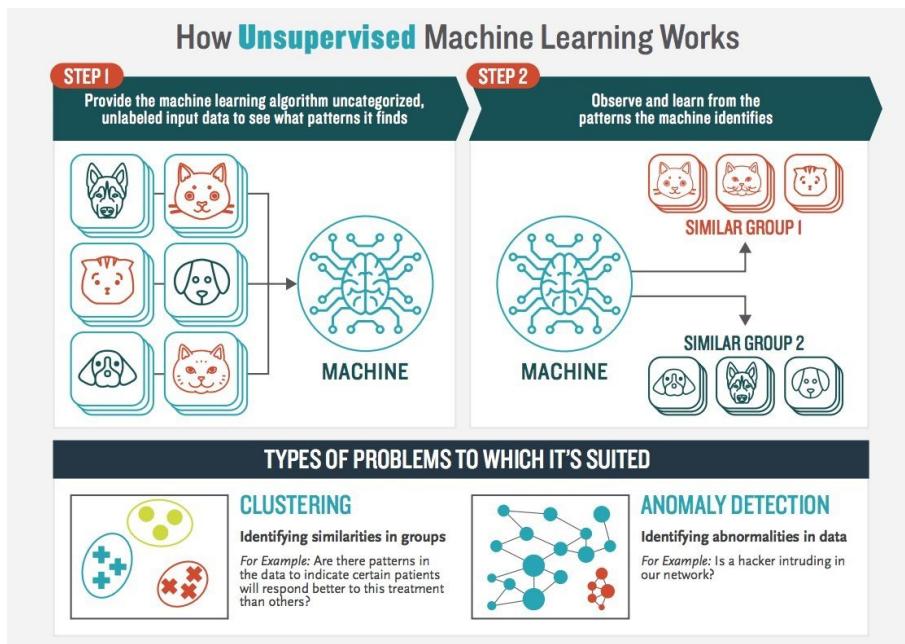


Gambar 2.238 Illustrasi cara kerja Supervised Learning

2. unsupervised learning

Unsupervised learning merupakan suatu algoritma di machine learning yang digunakan untuk menarik kesimpulan dari suatu dataset yang berisi data yang akan dinput tetapi tidak memiliki label.

Metode yang sering dipakai dalam Unsupervised learning adalah analisis cluster, yang menggunakan data untuk menganalisis beberapa pola yang terlihat untuk mengelompokan suatu data. Data akan dikelompokan berdasarkan kemiripan dengan menggunakan basis seperti misalkan Euclidean atau probabilistic distance



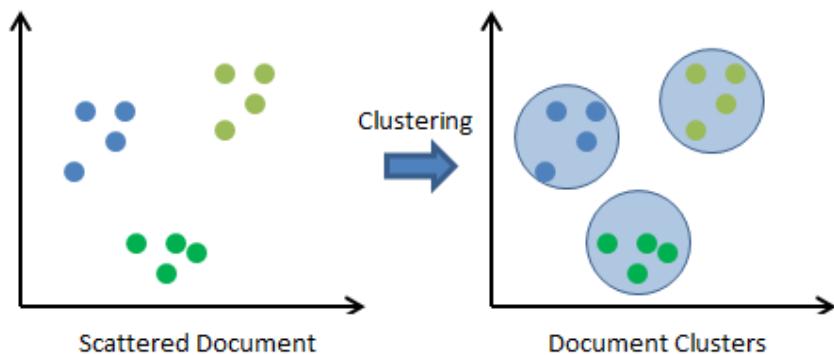
Gambar 2.239 Illustrasi cara kerja unsupervised learning

3. Clustering

2.11.1.3 *Jelaskan apa itu evaluasi dan akurasi dan disertai ilustrasi contoh dengan gambar sendiri*

Evaluasi merupakan tahap untuk mengidentifikasi seberapa baik hasil dari suatu proses yang bekerja untuk kemudian ditingkatkan atau diperbaiki kembali agar mendapatkan hasil yang meningkat atau hasil yang ingin kita dapatkan. Sedangkan akurasi sendiri yaitu mengukur bagaimana suatu klasifikasi membuat keputusan yang benar. Di machine learning sendiri, kita bisa mengevaluasi kinerja dari machine learning itu dengan menggunakan hal yang disebut sebagai confusion matrix

Clustering adalah suatu proses untuk mengelompokkan entity yang mirip secara bersamaan, tujuan dari metode Clustering yang merupakan bagian dari Unsupervised machine learning sendiri adalah untuk mencair kemiripan dari data data yang ada.



Gambar 2.240 gambaran clustering

		Actual Cancer = Yes	Actual Cancer = No
Predicted Cancer = Yes	True Positive (TP)	False Positive (FP)	
Predicted Cancer = No	False Negative (FN)	True Negative (TN)	

Gambar 2.241 contoh confusion matrix

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

- Pertama, kita harus punya dataset sebagai contoh, satu kolom untuk hasil yang diinginkan, dan satu kolumn untuk contoh hasil prediksi machine learning

1	Expected,	Predicted
2	man,	woman
3	man,	man
4	woman,	woman
5	man,	man
6	woman,	man
7	woman,	woman
8	woman,	woman
9	man,	man
10	man,	woman
11	woman,	woman

Gambar 2.242 Tahap 1 Confusion matrix

- Kemudian kita translate coding tersebut menjadi python seperti berikut

```
Created on Sun Mar  8 17:30:26 2020

@author: Chandra
"""

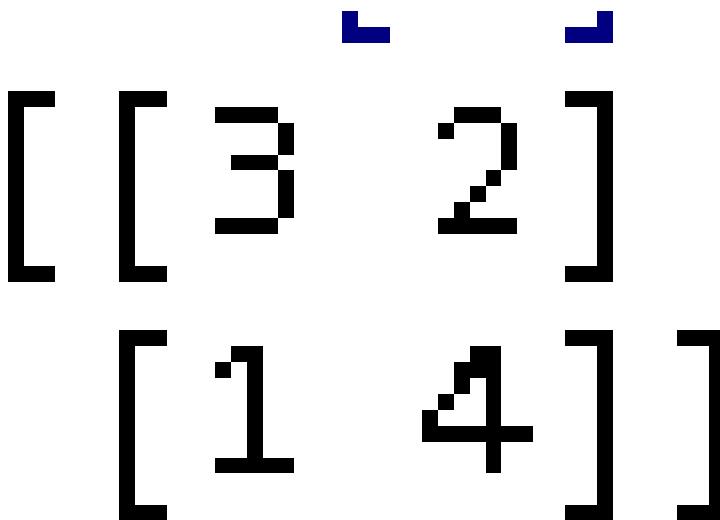
from sklearn.metrics import confusion_matrix
from sklearn.metrics import accuracy_score
from sklearn.metrics import classification_report

actual = ['man', 'man', 'woman', 'man', 'woman', 'woman', 'woman', 'man', 'man', 'woman']
predicted = ['woman', 'man', 'woman', 'man', 'man', 'woman', 'woman', 'man', 'woman', 'woman']
results = confusion_matrix(actual, predicted)

print(results)
```

Gambar 2.243 Tahap 2 confusion matrix

- Jalankan, dan lihat hasilnya



Gambar 2.244 Tahap 3 confusion matrix

4. Hasilnya, lelaki yang teridentifikasi sebagai lelaki adalah 3 orang, lelaki yang teridentifikasi sebagai perempuan sebanyak 2 orang, perempuan yang teridentifikasi sebagai lelaki sebanyak 1 orang dan perempuan yang teridentifikasi sebagai perempuan ada sebanyak 4 orang
5. rumus prediksi adalah sebagai berikut

$$\text{accuracy} = \frac{\# \text{ correct predictions}}{\# \text{ total data points}}$$

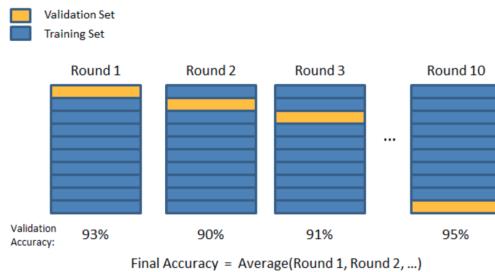
Gambar 2.245 Tahap 4 confusion matrix

6. Menggunakan rumus diatas, berarti jumlah akurasi yang didapat yaitu akurasi = total prediksi / jumlah data. yang berarti akurasi = $7 / 10 = 0,7$ atau 70 %

2.11.1.5 Jelaskan bagaimana K-fold cross validation bekerja dengan gambar ilustrasi contoh buatan sendiri.

Cara kerja k-fold validation adalah dengan cara membagi sample data secara

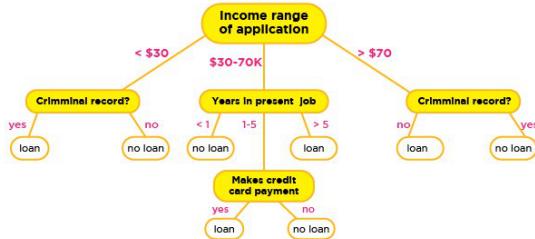
acak sejumlah k yang kita definisikan sebagai jumlah subsample. Misalkan kita punya 12 data, dan kita definisikan k = 3, berarti kita akan membagi jumlah data 12 itu menjadi 3 bagian yang berisi masing masing 4 data, kemudian bagian 1 ini akan kita jadikan sebagai data yang akan digunakan untuk validasi, dan sisanya yaitu bagian 2 dan 3 menjadi training data, setelah itu data pada bagian 1 akan digunakan untuk memvalidasi data pada bagian 2 dan 3 yang kemudian apabila bagian satu sudah selesai dijadikan sebagai validasi data, maka tahap selanjutnya yaitu data pada bagian 2 akan dijadikan data untuk memvalidasi data pada bagian 1 dan 3, dan begitu seterusnya.



Gambar 2.246 contoh K-Fold Validation

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

Decision Tree merupakan suatu konsep untuk yang berisi aturan aturan keputusan. Manfaat dari decision tree sendiri adalah untuk mempermudah pengambilan keputusan yang rumit menjadi lebih sederhana dan tergambarkan sehingga nantinya akan lebih mudah dalam proses pengambilan keputusan.



Gambar 2.247 Decision Tree bermain Credit

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

1. information Gain

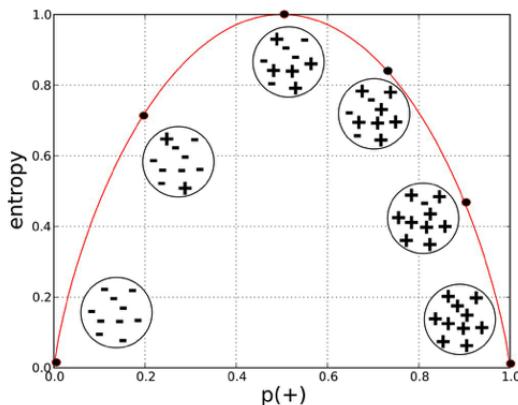
Information gain merupakan cara untuk memberi nilai pada suatu informasi dengan cara rumus dibawah ini

$$Gain(T, X) = Entropy(T) - Entropy(T, X)$$

Gambar 2.248 information gain

2. Entropi

Entropi adalah nilai informasi yang menyatakan ukuran ketidakpastian(impurity) dari attribut dari suatu kumpulan obyek data dalam satuan bit. Lihat gambar sebagai ilustrasi



Gambar 2.249 penggambaran entropi

2.11.2 Praktek

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

¹ 1174079 % 3

² #Hasilnya = 2 maka mengambil variable dengan nama buah

2.11.2.1 Praktek No. 1

```
1 import pandas as pd
```

```

2 apel = pd.read_csv('F:/ Poltekpos/D4 TI 3C/Semester 6/Kecerdasan
    Buatan/Github/Upload 8 Maret 2020/src/1174079/2/dataset/
        student-mat.csv', sep=';')
3 len(apel)

```

load library panda sebagai pd kemudian buat variable bernama apel yang diisi dengan function readcsv dari library pd(panda) dan definisikan file yang dimaksud dengan separator titik koma, kemudian perintah len akan mengukur panjang data dari variable apel

2.11.2.2 Praktek No. 2

```

1 apel[ 'pass' ] = apel . apply (lambda row: 1 if (row[ 'G1' ]+row[ 'G2' ]+
    row[ 'G3' ]) >= 30 else 0, axis=1) #mendeklarasikan pass/fail nya
    data berdasarkan G1+G2+G3.
2 apel = apel . drop ([ 'G1' , 'G2' , 'G3' ] , axis=1)
3 apel . head () #

```

menentukan pass/fail nya data melalui parameter $G1+G2+G3 = 30$. kemudian pada variabel mochi dideklarasikan jika baris dengan ut

2.11.2.3 Praktek No. 3

```

1 apel= pd.get_dummies(apel,columns=[ 'sex' , 'school' , 'address' , ,
    'famsize' , 'Pstatus' , 'Mjob' , 'Fjob' , 'reason' , 'guardian' ,
    'schoolsup' , 'famsup' , 'paid' , 'activities' , 'nursery' , 'higher' ,
    'internet' , 'romantic' ])
2 apel.head()#memanggil variabel dengan ketentuan head ini
    digunakan untuk mengembalikan baris n atas 5 secara default
    dari frame atau seri data

```

One hot encoding merupakan suatu proses yang digunakan untuk mengkonversi data yang bisa dikelompokkan menjadi bentuk yang bisa disediakan untuk algoritma machine learning membacanya secara lebih mudah.

2.11.2.4 Praktek No. 4

```

1 apel= apel.sample(frac=1)#mengembalikan variabel menjadi sampel
    acak dengan frac=1
2 apel_train = apel[:500]
3 apel_test = apel[500:]
4 apel_train_att = apel_train.drop([ 'pass' ] , axis=1)
5 apel_train_pass = apel_train[ 'pass' ]
6 apel_test_att = apel_test.drop([ 'pass' ] , axis=1)
7 apel_test_pass = apel_test[ 'pass' ]
8 apel_att = apel.drop([ 'pass' ] , axis=1)
9 apel_pass = apel[ 'pass' ]
10
11 import numpy as np #mengimport module numpy sebagai np y
12 print("Passing: %d out %d (%.2f%%)" %(np.sum(apel_pass),len(
    apel_pass),100*float(np.sum(apel_pass))/len(apel_pass)))

```

Variable yang ada diisi dengan data sample yang kemudian dihitung keakuratannya pada bagian printf dengan rumus seperti digambar

2.11.2.5 Praktek No. 5

```

1 from sklearn import tree
2 mangga = tree.DecisionTreeClassifier(criterion="entropy",
3                                     max_depth=5) #membuat variabel bandung sebagai decisiontree ,
4                                     dengan criterion fungsi mengukur kualitas split
5 mangga = mangga.fit(apel_train_att, apel_train_pass)

```

Import library class tree dari library sklearn kemudian variable mangga diisi dengan function DecisionTreeClassifier dari class tree dengan parameter criteria sebagai entropy dan maxdepth nya 5 kali, kemudian variable mangga diisi dengan function fit yang akan menghitung data yang ada di parameter dengan metode yang sudah didefinisikan seperti di gambar

2.11.2.6 Praktek No. 6

```

1 import graphviz
2 pir = tree.export_graphviz(mangga, out_file=None, label ="all",
3                            impurity=False , proportion=True , feature_names=list(
4                                apel_train_att), class_names=[" fail", " pass"] , filled=True ,
5                                rounded=True)#mengambil data untuk diterjemahkan ke grafik
6 avokado = graphviz.Source(pir)
7 avokado

```

Graphviz adalah library yang digunakan untuk visualisasasi data, kode-kode pada bagian ini kurang lebih hanya untuk visualiasi data berdasarkan data dan aturan dari apa yang telah didefinisikan di variable pir, kemudian pada variable avokado dia memanggil function source untuk meminta sumber data dan settingnya, dan akhirnya ditampilkan

2.11.2.7 Praktek No. 7

```

1 tree.export_graphviz(mangga, out_file="student-performance.dot" ,
2                      label ="all",impurity=False , proportion=True , feature_names=
3                      list(apel_train_att),class_names=[" fail", " pass"] , filled=True ,
4                      rounded=True)#save tree sebagai export graphviz ke file
5 student-performance.dot

```

tree.export_graphviz digunakan untuk membuat graphic, student-performance dot adalah nama output filenya dan sisanya yaitu parameter yang optional yang digunakan untuk apakah anda ingin menampilkan seluruh data yang ada pada node atau hanya beberapa node saja yang ditampilkan, seperti itu.

2.11.2.8 Praktek No. 8

```

1 mangga.score(apel.att, apel.pass) #score juga disebut prediksi
2 dengan diberi beberapa data input baru

```

Score akan membandingkan parameter pertama sebagai hasil yang diinginkan dan parameter kedua sebagai pembanding yang nantinya akan dikalkulasi tingkat kemiripannya dalam bentuk persen

2.11.2.9 Praktek No. 9

```

1 from sklearn.model_selection import cross_val_score
2 anggur = cross_val_score(mangga, apel_att , apel_pass ,cv=5) #
    mengevaluasi score dengan validasi silang
3 # show average score and +/- two standard deviations away
4 print("Accuracy : %0.2f (+/- %0.2f)" % (anggur.mean() ,anggur .std()
() * 2))

```

Script ini akan mencoba untuk mengevaluasi data dengan metode cross validation . Dimana variabel anggur yang diisi dengan cross valscore yang merupakan fungsi dari class yang kita import. Kemudian akan memperlihatkan score rata rata dan juga nilai deviasinya

2.11.2.10 Praktek No. 10

```

1 for belimbing in range(1,20):
2     mangga = tree.DecisionTreeClassifier(criterion="entropy",
3                                         max_depth=belimbing)
4     anggur = cross_val_score(mangga, apel_att , apel_pass ,cv=5)
5     print("Max depth : %d, Accuracy : %0.2f (+/- %0.2f)" %(belimbing , anggur.mean() ,anggur .std() * 2))

```

script ini berisi tentang perulangan terlebih dahulu yang akan diisi parameternya sebagai 1,20 yang berarti akan diulang perintahnya sebanyak 19 kali, kemudian variable mangga berisi fungsi DecisionTreeClassifier dengan kriteria entropy dengan max-depth yang artinya jumlah fold sebanyak nilai dari variable belimbing, di variable anggur diisi dengan fungsi crossvalscore yang menerima parameter mangga sebagai pengaturan tree, dan apelatt serta apel-pass sebagai data dan cv=5 sebagai jumlah foldnya

2.11.2.11 Praktek No. 11

```

1 stroberi = np.empty((19 ,3) ,float)
2 jeruk = 0
3 for belimbing in range(1,20):
4     mangga = tree.DecisionTreeClassifier(criterion="entropy",
5                                         max_depth=belimbing)#variabel bandung untuk decision tree
6                                         dengan ketentuan entropy
7     anggur = cross_val_score(mangga, apel_att , apel_pass ,cv=5)
8     stroberi[jeruk ,0] = belimbing
9     stroberi[jeruk ,1] = anggur.mean()
10    stroberi[jeruk ,2] = anggur.std() * 2
11    jeruk += 1
12    stroberi

```

Variable stroberi diisi dengan np yang berarti library numpy lalu memanggil function empty dengan nilai 19,3 dengan tipe data float, lalu ada variable jeruk yang diisi 0, setelah itu ada perulangan belimbing yang diulang sebanyak 19 kali, kemudian ada variable mangga yang diisi function DecisionTreeClassifier

dengan settingnya yang sudah dijelaskan sebelumnya, lanjut kebagian variable anggur yang kurang lebih sama seperti penejalan di nomor 10, kemudian variable stroberi akan mengakses arranya pada bagian yang didefinisikan oleh variable jeruk dan juga 0 yang datanya akan diganti oleh data dari variable belimbing, kemudian variable stroberi akan mengakses bagian yang didefinisikan oleh jeruk, dan 1 yang datanya akan diisi dengan nilai rata rata dari variable anggur, lanjut stroberi akan mengakses array pada bagian yang didefinisikan oleh variable jeruk dan 2 yang akan diisi dengan nilai deviasi dari variable anggur dikali dua, kemudian variable jeruk ditambah 1 lalu menampilkan data stroberi

2.11.2.12 Praktek No. 12

```

1 import matplotlib.pyplot as plt
2 manggis, pisang = plt.subplots()
3 pisang.errorbar(stroberi[:,0], stroberi[:,1], yerr=stroberi[:,2]) #
    membuat error bar kemudian grafik akan ditampilkan
    menggunakan show
4 plt.show()

```

Load library pyplot dari matplotlib sebagai plt, dan buat variable manggis serta pisang yang diisi funtion subplot dari pyplot, kemudian variable pisang memanggil function error bar yang diisi data seperti yang didefnisikan di gambar lalu menampilkannya.

2.11.3 Penanganan Error

2.11.3.1 Error No Module Named Graphviz

Solusi: Pastikan library graphviz terinstall, jika belum buka anaconda prompt kemudian ketika conda install python-graphviz untuk menginstall

2.11.4 Bukti Tidak Plagiat



Gambar 2.250 Plagiarism

2.11.5 Link Video Youtube

<https://youtu.be/9fuSYsqxuVg>

2.12 1174053 - Dini Permata Putri

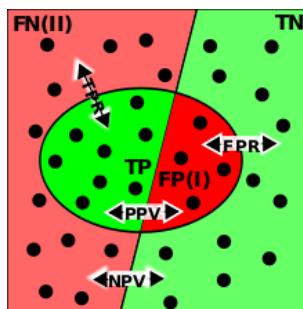
2.12.1 Teori

1. Binary Classification

Binary Classification adalah suatu tugas mengklasifikasikan himpunan yang didalamnya terdapat elemen-elemen yang dimasukkan ke dalam kelompok berdasarkan aturan klasifikasi. Beberapa karakteristik tersebut contohnya tes medis digunakan untuk mengetahui suatu pasien memiliki penyakit tertentu atau tidak. dan properti klasifikasi tersebut adalah keberadaan penyakit dari pasien.

Binary Classification digunakan untuk tujuan praktis dalam banyak masalah klasifikasi biner, dan kedua kelompok tersebut tidak simetris daripada akurasi secara keseluruhan, proporsi relatif dari berbagai macam kesalahan yang menarik. Contohnya, dalam pengujian medis tadi, false positif maksudnya mendetectsi penyakit ketika ada sedangkan untuk false negatif artinya tidak mendetectsi penyakit ketika ada.

Ada banyak metrik yang bisa digunakan dalam mengukur kinerja klasifikasi dan prediksi. misalnya dapat dilihat pada gambar berikut:



Gambar 2.251 Binary Classification

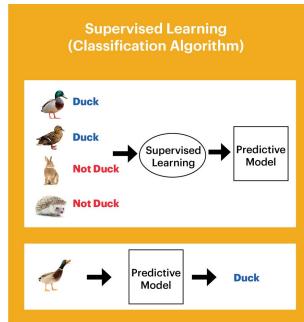
Bagian kiri dan kanan masing-masing memiliki instance yang sebenarnya ada dan tidak memiliki kondisi. Sedangkan bentuk oval tersebut berisi instance yang diklasifikasikan atau diprediksi sebagai positif atau negatif.

2. Supervised Learning, Unsupervised Learning, dan Clustering

- **Supervised Learning**

Dalam Supervised Learning, suatu program komputer diberikan dataset pelatihan yang kemudian diberi label dengan nilai output yang sesuai, dan fungsi tersebut akan ditentukan berdasarkan pada dataset. Fungsi atau algoritma tersebut kemudian akan digunakan untuk mengklasifikasikan data baru untuk memprediksi nilai-nilai output yang sesuai

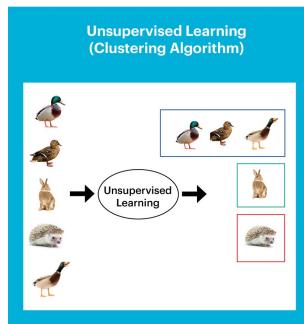
dengan asumsi bahwa data baru sesuai dengan aturan dan fungsi yang digunakan. Berikut adalah contoh supervised learning:



Gambar 2.252 Supervised Learning

- **Unsupervised Learning**

Dalam Unsupervised Learning, dataset pelatihan tidak memiliki label nilai output yang sesuai, karena tidak ada jawaban benar untuk dipelajari, tujuan algoritma ini adalah untuk mengungkap pola-pola menarik yang dapat ditemukan dalam data, dan data baru akan membantu untuk mengonfirmasi atau membatalkan pola-pola yang ditemukannya. Berikut adalah contoh Unsupervised Learning:

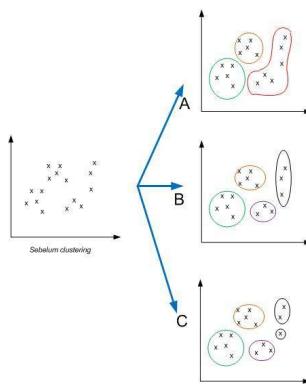


Gambar 2.253 Unsupervised Learning

- **Clustering**

Clustering adalah suatu teknik yang masuk kedalam kelompok Unsupervised Learning yang merupakan teknik dimana mesin akan bekerja atau belajar sendiri tanpa diajari bagaimana cara memecahkan masalahnya. Contohnya, Kita memiliki sebuah data, yaitu data pelanggan yang berisi jenis kelamin, besarnya penghasilan dan besarnya pembelian produk. Maka dengan algoritma Clustering kita dapat

mengetahui pelanggan kita akan dikelompokkan kedalam beberapa kluster dengan sendirinya. Misalnya ada pelanggan yang pelit, pelangan yang royal dan lain sebagainya. Contohnya, Bisa kita lihat bagaimana sebuah teknik clustering bisa mengelompokkan data ke dalam beberapa kluster.



Gambar 2.254 Clustering

3. Evaluasi dan Akurasi

Evaluasi merupakan suatu cara atau teknik dalam mengevaluasi seberapa baik model bekerja dengan mengukur akurasinya. Sedangkan akurasi adalah suatu persentase kasus yang diklasifikasikan dengan benar. Kita bisa menganalisis suatu kesalahan yang dibuat dengan model atau tingkat confusion dengan menggunakan matriks confusion. Berikut adalah contoh klasifikasi biner yang menunjukkan berapa kali model telah membuat prediksi yang benar dari objek.

	Predicted "apple"	Predicted "orange"
True "apple"	20	5
True "orange"	3	22

Gambar 2.255 Evaluasi

Dalam tabel diatas, baris True Apple dan True Orange mengacu pada suatu kasus dimana objek itu sebenarnya sebuah apel atau sebenarnya jeruk. Kolom merujuk pada prediksi yang dibuat oleh model. Kita melihat bahwa ada 20 apel yang diprediksi dengan benar, sementara ada 5 apel yang salah diidentifikasi sebagai jeruk. Sehingga, matriks confusion harus memiliki semua nol, kecuali untuk diagonal sehingga kita dapat menghitung akurasi dengan menambahkan angka secara diagonal seperti pada gambar berikut :

$$\text{Accuracy} = (20 + 22)/20 + 5 + 3 + 22) = 84$$

Gambar 2.256 Akurasi

4. Cara Membuat dan Membaca Confusion Matrix

Confusion Matrix adalah suatu matrix yang memberikan informasi perbandingan hasil klasifikasi yang dilakukan pada sistem atau model dengan hasil klasifikasi sebenarnya. Confusion Matrix berbentuk tabel matriks yang menggambarkan suatu kinerja model klasifikasi dari serangkaian data uji yang nilai sebenarnya diketahui.

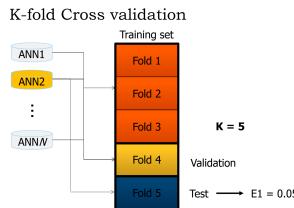
Berikut cara membuat dan membaca confusion matrix : pertama, tentukan terlebih dahulu pokok permasalahan dan atributnya. kedua, buatlah pohon keputusan dan data testingnya. ketiga, carilah nilai a,b,c dan d. Selanjutnya, cari nilai recall, precesion, accuracy serta seeror rate. Berikut contoh Confusion Matrix :

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

Gambar 2.257 Confusion Matrix

5. Cara Kerja K-fold Cross Validation

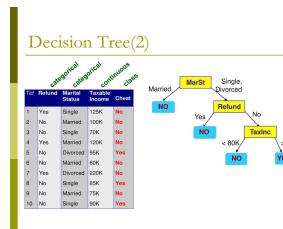
- Pertama, total instance bagi menjadi N bagian
- Fold pertama merupakan bagian peertama yang menjadi data uji atau testing data dan sisanya menjadi training data
- Kemudian, hitung akurasi dari porsi data dengan menggunakan persamaan
- Fol kedua, adalah bagian kedua dengan menjadi data uji atau testing data dan sisanya merupakan training data
- Lalu, hitung akurasi dari porsi data tersebut
- Lakukan hal tersebut, sampai habis mencapai fold ke-K
- Setelah itu, hitung rata-rata akurasi K



Gambar 2.258 K-fold Cross Validation

6. Decision Tree

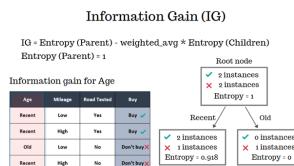
Decision Tree adalah salah satu model prediksi dengan menggunakan struktur pohon atau struktur yang berhirarki. Konsepnya adalah mengubah data menjadi decision tree dan beberapa aturan keputusan. Decision Tree memiliki manfaat yaitu membrekdown proses pengambilan keputusan yang kompleks menjadi lebih simpel sehingga pengambil keputusan akan lebih mudah dipahami.



Gambar 2.259 Decision Tree

7. Information Gain and Entropy

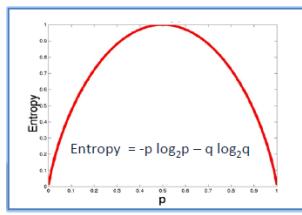
Information Gain adalah suatu teknik yang didasarkan pada penurunan entropi setelah dataset dibagi pada atribut. Membangun keputusan adalah menemukan atribut yang mengembalikan perolehan informasi tertinggi.



Gambar 2.260 Information Gain

Entropi adalah suatu ukuran acak dalam informasi yang sedang diproses. Semakin tinggi suatu entropi, semakin sulit menarik kesimpulan dari informasi tersebut. Decision tree dibangun dari atas kebawah dan meli-

batkan partisi data kedalam himpunan bagian yang berisi instance dengan nilai yang sama. Algoritma ID3 menggunakan entropi untuk menghitung suatu homogenitas sampel.



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

Gambar 2.261 Entropy

2.12.2 Praktek

1. Nomor 1

```

1 seblak = pd.read_csv('student-mat.csv',sep=';')
2 len(seblak)
3
4 #%% 2. generate binary label (pass/fail) based on G1+G2+G3
5 seblak['pass'] = seblak.apply(lambda row: 1 if (row['G1']+row['G2']+row['G3'])>= 35 else 0, axis=1)

```

Hasilnya :

```

In [45]: makassar['pass'] = makassar.apply(lambda row: 1 if (row['G1']+row['G2']+row['G3']) >= 35 else 0, axis=1)
          #memperbaikkan pass/fail nya data berdasarkan
          #jumlah nilai matematika dan hasilnya sama dengan 1 maka artinya lulus
...:1 makassar = makassar.drop(['G1', 'G2', 'G3'], axis=1) #untuk menghindari
...:1 error: 'makassar' object has no attribute 'drop'. Did you mean 'dropna'?
...:1 makassar.head() #menampilkan 5 baris pertama hasilnya sama dengan 1
...:1 df = pd.get_dummies(makassar, drop_first=True) #drop first untuk menghindari berulang n atau 5 secara default dari frame atau seri
...:1 df
Out[45]:
sex age address famsize ... Dalc Walc health absences pass
0 GP 18 U 3GT ... 1 1 3 6 0
1 GP 19 U 3GT ... 1 1 3 6 0
2 GP 15 U L3T ... 2 3 3 3 10 0
3 GP 15 U 3GT ... 1 1 3 5 1
4 GP 18 U 3GT ... 1 2 5 4 0
[5 rows x 31 columns]

```

Gambar 2.262 Hasil Nomor 1

2. Nomor 2

```

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

```

Hasilnya :

```
In [45]: makassar['pass'] = makassar.apply(lambda row: 1 if (row['G1']>row['G2']) & (row['G3'] >= 35) else 0, axis=1)
makassar = makassar.drop(['G1', 'G2', 'G3'], axis=1)
makassar['head'] = makassar['head'].replace(0, 1)
makassar['head'] = makassar['head'].replace(1, 0)
makassar['head'].head()
Out[45]:
   school sex age address famsize ... Dalc Wabs health absences pass
0   GP    F   17      U   GT3 ...   1     1     3     4     0
1   GP    F   17      U   GT3 ...   1     1     3     4     0
2   GP    F   15      U   LSS ...   2     5     3     10    0
3   GP    F   15      U   LSS ...   2     5     3     10    0
4   GP    F   16      U   GT3 ...   1     2     5     3     0
[5 rows x 31 columns]
```

Gambar 2.263 Hasil Nomor 2

3. Nomor 3

```
1 seblak_train_att = seblak_train.drop(['pass'], axis=1)
2 seblak_train_pass = seblak_train['pass']
3 seblak_test_att = seblak_test.drop(['pass'], axis=1)
4 seblak_test_pass = seblak_test['pass']
5 seblak_att = seblak.drop(['pass'], axis=1)
6 seblak_pass = seblak['pass']
```

Hasilnya :

$$\text{Accuracy} = (20 + 22)/20 + 5 + 3 + 22) = 84$$

Gambar 2.264 Hasil Nomor 3

4. Nomor 4

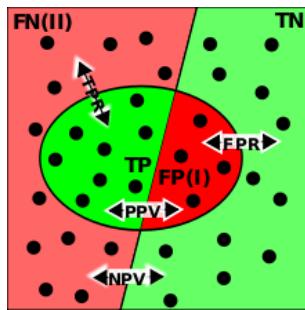
```
1 #%% 5. fit a decision tree
2 from sklearn import tree
3 surabi = tree.DecisionTreeClassifier(criterion="entropy",
4                                     max_depth=5)
4 surabi = surabi.fit(surabi_train_att, subang_train_pass)
5
6 #%% 6. visualize tree
7 import graphviz
8 martabak = tree.export_graphviz(surabi, out_file=None, label = "all",
9                                 impurity=False, proportion=True, feature_names=list(
10                                seblak_train_att), class_names=["fail", "pass"], filled=True,
11                                rounded=True)
9 baso = graphviz.Source(martabak)
10 baso
11
12 #%% 7. save tree
13 tree.export_graphviz(surabi, out_file="student-performance.dot",
14                      label = "all", impurity=False, proportion=True,
15                      feature_names=list(seblak_train_att), class_names=["fail", "pass"],
16                      filled=True, rounded=True)
14
15 #%% 8
16 surabi.score(seblak_test_att, seblak_test_pass)
```

```

17
18 #%% 9
19 from sklearn.model_selection import cross_val_score

```

Hasilnya :



Gambar 2.265 Hasil Nomor 4

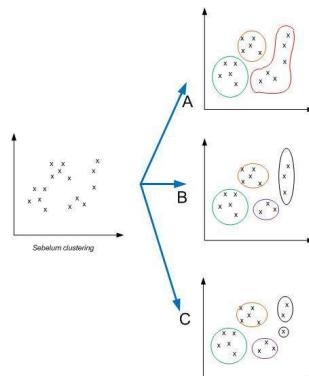
5. Nomor 5

```

1 #%% 10
2 for siomay in range(1,20):
3     surabi = tree.DecisionTreeClassifier(criterion="entropy",
4                                         max_depth=siomay)
5     batagor = cross_val_score(surabi, seblak_att, seblak_pass,
6                               cv=5)
    print("Max depth : %d, Accuracy : %0.2f (+/- %0.2f)" %(siomay, batagor.mean(), batagor.std() * 2))

```

Hasilnya :



Gambar 2.266 Hasil Nomor 5

6. Nomor 6

```

1 tekwan = 0
2 for siomay in range(1,20):
3     surabi = tree.DecisionTreeClassifier(criterion="entropy",
4         max_depth=siomay)
5     batagor = cross_val_score(surabi, seblak_att, seblak_pass,
6         cv=5)
7     pempek[tekwan,0] = siomay
8     pempek[tekwan,1] = batagor.mean()
9     pempek[tekwan,2] = batagor.std() * 2
10    tekwan += 1
11    pempek

```

Hasilnya :

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

Gambar 2.267 Hasil Nomor 6

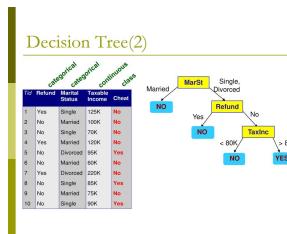
7. Nomor 7

```

1 basreng.errorbar(pempek[:,0], pempek[:,1], yerr=pempek[:,2])
2 plt.show()

```

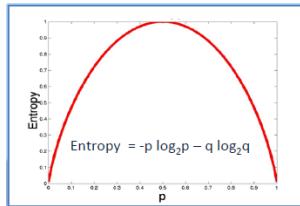
Hasilnya :



Gambar 2.268 Hasil Nomor 7

8. Nomor 8

Hasilnya :



$$Entropy = -0.5 \log_2 0.5 - 0.5 \log_2 0.5 = 1$$

Gambar 2.269 Hasil Nomor 8

9. Nomor 9

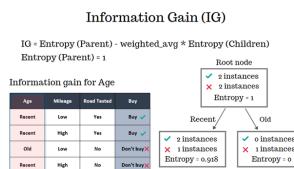
Hasilnya :

	Predicted "apple"	Predicted "orange"
True "apple"	20	5
True "orange"	3	22

Gambar 2.270 Hasil Nomor 9

10. Nomor 10

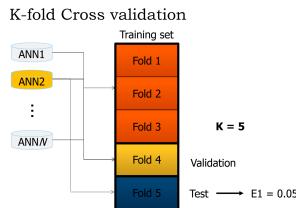
Hasilnya :



Gambar 2.271 Hasil Nomor 10

11. Nomor 11

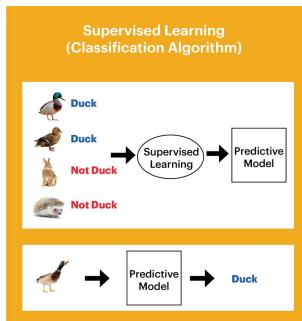
Hasilnya :



Gambar 2.272 Hasil Nomor 11

12. Nomor 12

Hasilnya :



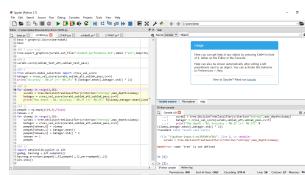
Gambar 2.273 Hasil Nomor 12

2.12.3 Penanganan Error

1. ScreenShoot Error

```
file "D:\Kuliah\Semester 6\Kecerdasan Buatan\Dataset\sklearn.py", line 8, in <module>
    from sklearn import datasets
ImportError: cannot import name 'datasets' from 'sklearn' (D:\Kuliah\Semester 6\Kecerdasan Buatan\sklearn.py)
```

Gambar 2.274 Module Not Found Error



Gambar 2.275 File Not Found Error

2. Tuliskan Kode Error dan Jenis Error

- Module Not Found Error
- File Not Found Error
- Executable Not Found

3. Cara Penanganan Error

- Module Not Found Error

Dengan memperbaiki penulisan atau kesalahan dalam penulisan kode atau melakukan install package atau modul yang belum terinstal

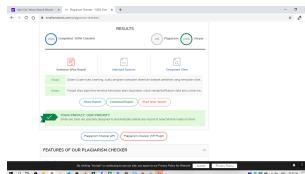
- File Not Found Error

Dengan memperbaiki directory file. Sesuaikan sama tempat penyimpanan di laptop

- Executable Not Found

Dengan menginstal aplikasi graphviz di windows dan menambahkan directory diatas kode program

2.12.4 Bukti Tidak Plagiat



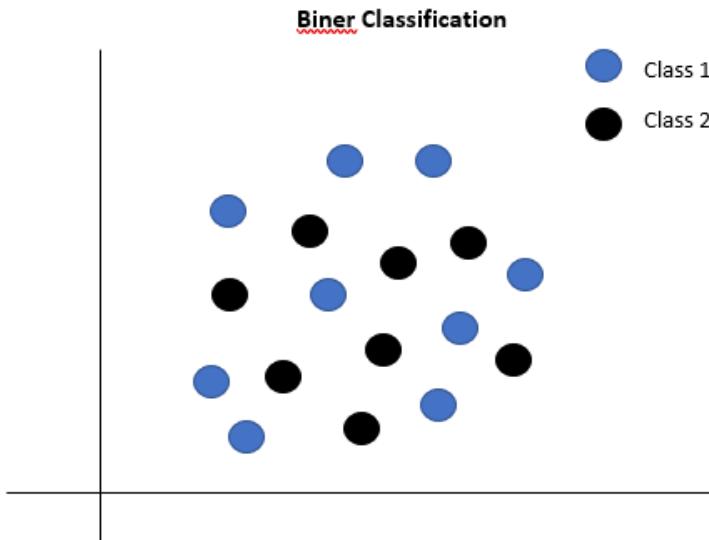
Gambar 2.276 Bukti Tidak Plagiat

2.12.5 Link Youtube

2.13 1174077 - Alvan Alvanzah

2.13.1 Teori

2.13.1.1 *Jelaskan Apa Itu Binary Classification dilengkapi ilustrasi gambar sendiri.*
Klasifikasi biner bertujuan untuk mengklasifikasikan elemen-elemen dari him-punan yang diberikan ke dalam dua kelompok (memprediksi kelompok mana yang masing-masing dimiliki) berdasarkan aturan klasifikasi.

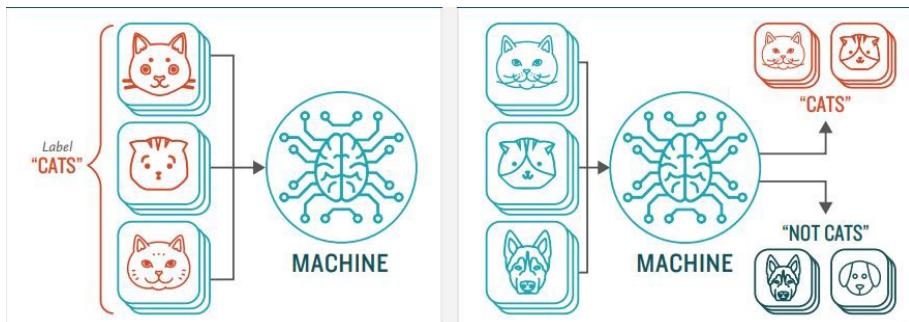


Gambar 2.277 Binary Classification

2.13.1.2 Jelaskan Apa itu supervised learning , unsupervised learning dan clustering dengan ilustrasi gambar sendiri.

1. supervised learning

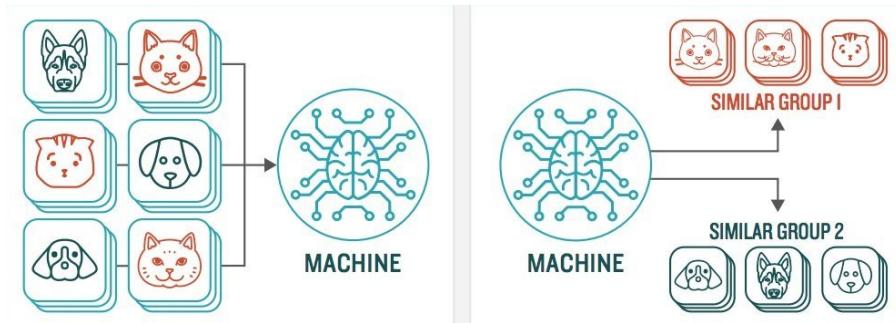
Supervised learning adalah suatu pembelajaran yang terawasi dimana output yang diharapkan telah diketahui sebelumnya. Biasanya pembelajaran ini dilakukan dengan menggunakan data yang telah ada. Dan Supervised Learning dalam bahasa indonesia adalah pembelajaran yang ada supervisornya. Maksudnya ada supervisornya adalah label di tiap data nya.



Gambar 2.278 Supervised Learning

2. unsupervised learning

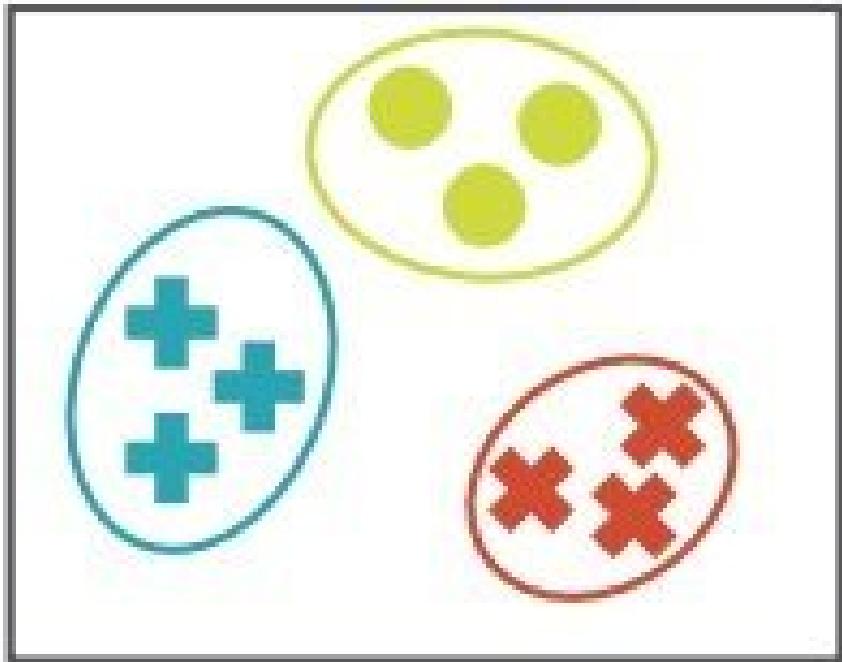
Unsupervised learning berbeda dengan supervised learning. Unsupervised learning memiliki keunggulan dari supervised learning. Supervised learning memiliki label sebagai dasar prediksi untuk membuat clasification dan regression algorithm yang memungkinkan. Tetapi dalam realitanya, data real banyak yang tidak memiliki label. Jadi unsupervised learning menggunakan ke samaan dari attribut attribut yang dimiliki untuk mencari kemiripan,dan kemudian dikelompok kelompokan (clustering). Sehingga hal ini akan menimbulkan kelompok kelompok (cluster).



Gambar 2.279 Unsupervised Learning

3. Clustering

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



Gambar 2.280 gambaran clustering

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

Evaluasi adalah tentang bagaimana kita dapat mengevaluasi seberapa baik model bekerja dengan mengukur akurasinya. Dan akurasi akan didefinisikan sebagai persentase kasus yang diklasifikasikan dengan benar. Kita dapat menganalisis kesalahan yang dibuat oleh model, atau tingkat kebingungannya, menggunakan matriks kebingungan.

	Hasil Prediksi "Mangga"	Hasil Prediksi "Jambu"
True "Mangga"	24	1
True "Jambu"	3	22

Gambar 2.281 Evaluasi dan Akurasi

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

Confusion Matrix suatu metode yang biasanya digunakan untuk melakukan perhitungan akurasi pada konsep data mining atau Sistem Pendukung Keputusan. Pada pengukuran kinerja menggunakan confusion matrix, terdapat 4

(empat) istilah sebagai representasi hasil proses klasifikasi. Keempat istilah tersebut adalah True Positive (TP), True Negative (TN), False Positive (FP) dan False Negative (FN). Nilai True Negative (TN) merupakan jumlah data negatif yang terdeteksi dengan benar, sedangkan False Positive (FP) merupakan data negatif namun terdeteksi sebagai data positif. Sementara itu, True Positive (TP) merupakan data positif yang terdeteksi benar. False Negative (FN) merupakan kebalikan dari True Positive, sehingga data positif, namun terdeteksi sebagai data negatif.

		True Values	
		True	False
Prediction	True	TP Correct result	FP Unexpected result
	False	FN Missing result	TN Correct absence of result

Gambar 2.282 Confusion Matrix

2.13.1.5 Jelaskan bagaimana K-fold cross validation bekerja dengan gambar ilustrasi contoh buatan sendiri.

Cara Kerja k-fold cross validation:

- Total instance dibagi menjadi N bagian.
- Fold yang pertama adalah bagian pertama menjadi data uji (testing data) dan sisanya menjadi training data.
- Lalu hitung akurasi berdasarkan porsi data tersebut dengan menggunakan persamaan.
- Fold yang kedua adalah bagian kedua, yang menjadi data uji(testing data)dan sisanya training data.
- Kemudian hitung akurasi berdasarkan porsi data tersebut.
- Dan seterusnya hingga habis mencapai fold ke-K.
- Terakhir hitung rata-rata akurasi K buah.

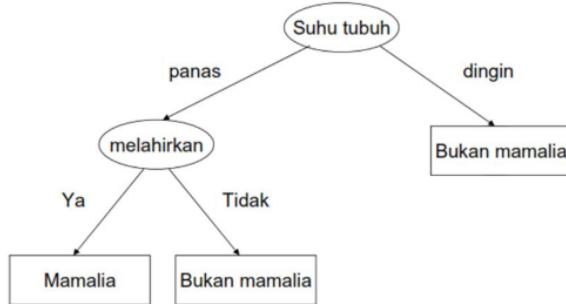


Gambar 2.283 K-Fold Validation

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

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

Contoh Pohon Keputusan : Klasifikasi Vertebrata



Gambar 2.284 Decision Tree Hewan Vertebrata

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

1. information Gain

Information Gain merupakan total data yang didapat dari data - data acak yang data tersebut akan digunakan untuk analisis data lainnya.

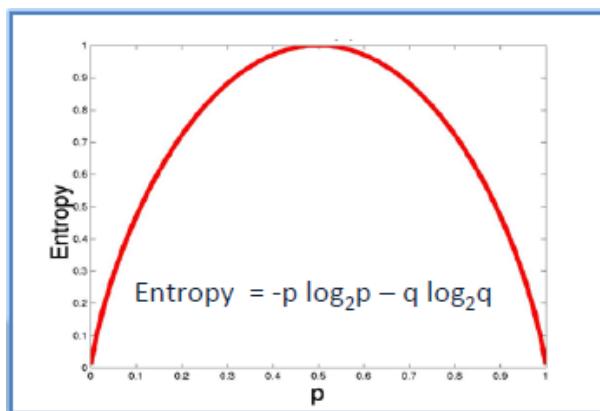
Information Gain ini digunakan pada decision tree sebagai label setiap aksi - aksi yang perlu dinilai validasinya.



Gambar 2.285 information gain

2. Entropi

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



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

Gambar 2.286 Entropi

2.13.2 Praktek

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

```
1
2 # In [0]
```

2.13.2.1 Nomor 1

```
1 # In [1]
2 import pandas as pd #import library pandas dan sebagai pd
3 papeda = pd.read_csv('C:/Users/ASUS/Downloads/KB3C-master/KB3C-
    master/src/1174077/2/dataset/student-mat.csv', sep=';') #
    Membuat variable papeda yang isinya memanggil fungsi membaca
    file csv
```

```
In [1]: import pandas as pd #import library pandas dan sebagai pd
...: papeda = pd.read_csv('C:/Users/ASUS/Downloads/KB3C-master/KB3C-master/src/
1174077/2/dataset/student-mat.csv', sep=';') #Membuat variable papeda yang isinya
memanggil fungsi membaca file csv
...: len(papeda) #Menghitung jumlah data yang ada pada csv yang sudah dibaca
Out[1]: 395
```

Gambar 2.287 Nomor 1

2.13.2.2 Nomor 2

```
1
2 # In [2]
3 # generate binary label (pass/fail) based on G1+G2+G3 (test
    grades, each 0–20 pts); threshold for passing is sum>=30
4 papeda['pass'] = papeda.apply(lambda row: 1 if (row['G1']+row['G2']
    )+row['G3']) >= 35 else 0, axis=1) #Membuat label binary (
    pass/fail) berdasarkan G1+G2+G3 (testgrade, semuanya 0–20
    point); Batas untuk pass adalah sum>=30
5 papeda = papeda.drop(['G1', 'G2', 'G3'], axis=1) #Meghilangkan
    data G1 G2 dan G3
```

	school	sex	age	address	famsize	...	Dalc	Walc	health	absences	pass
0	GP	F	18	U	GT3	...	1	1	3	6	0
1	GP	F	17	U	GT3	...	1	1	3	4	0
2	GP	F	15	U	LE3	...	2	3	3	10	0
3	GP	F	15	U	GT3	...	1	1	5	2	1
4	GP	F	16	U	GT3	...	1	2	5	4	0

[5 rows x 31 columns]

Gambar 2.288 Nomor 2

2.13.2.3 Nomor 3

```

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

```

	age	Medu	Fedu	...	internet_yes	romantic_no	romantic_yes
0	18	4	4	...	0	1	0
1	17	1	1	...	1	1	0
2	15	1	1	...	1	1	0
3	15	4	2	...	1	0	1
4	16	3	3	...	0	1	0

[5 rows x 57 columns]

Gambar 2.289 Nomor 3

2.13.2.4 Nomor 4

```

1
2 # In [4]:
3 # shuffle rows
4 papeda = papeda.sample(frac=1) #Mengambil data sample dari papeda
5 # split training and testing data
6 papeda_train = papeda[:500] #Membagi data untuk training
7 papeda_test = papeda[500:] #Membagi data untuk test
8
9 papeda_train_att = papeda_train.drop(['pass'], axis=1) #Meghapus
    data yang telah pass dan memasukkannya
10 papeda_train_pass = papeda_train['pass'] #Mengambil data yang
    pass saja
11
12 papeda_test_att = papeda_test.drop(['pass'], axis=1) #Meghapus
    data yang telah pass dan memasukkannya
13 papeda_test_pass = papeda_test['pass'] #Mengambil data yang pass
    saja
14
15 papeda_att = papeda.drop(['pass'], axis=1) #Meghapus data yang
    telah pass dan memasukkannya
16 papeda_pass = papeda['pass'] #Mengambil data yang pass saja
17
18 # number of passing students in whole dataset:
19 import numpy as np #Mengimport library numpy sebagai np

```

```

...: import numpy as np #Mengimport Library numpy sebagai np
...: print("Passing: %d out of %d (%.2f%%)" % (np.sum(papeda_pass), len(papeda_pass),
100*float(np.sum(papeda_pass)) / len(papeda_pass))) #Menampilkan data
Passing: 166 out of 395 (42.03%)

```

Gambar 2.290 Nomor 4

2.13.2.5 Nomor 5

```

1
2 # In [5]:
3 # fit a decision tree
4 from sklearn import tree #import Decision tree dari library
    sklearn
5 lontar = tree.DecisionTreeClassifier(criterion="entropy",
    max_depth=5) #Membuat decision tree dengan maximal depthnya 5

```

```

In [10]: from sklearn import tree #import Decision tree dari library sklearn
...: kue_lontar = tree.DecisionTreeClassifier(criterion="entropy", max_depth=5)
#Membuat decision tree dengan maximal depthnya 5
...: kue_lontar = kue_lontar.fit(papeda_train_att, papeda_train_pass) #Memasukkan
data yang akan dijadikan decision treenya

```

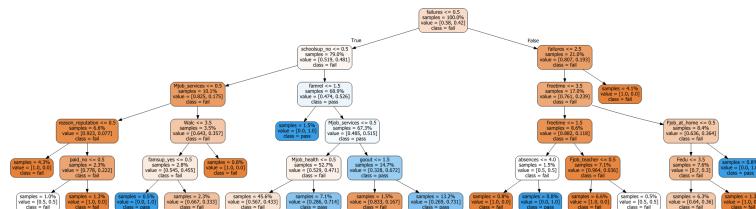
Gambar 2.291 Nomor 5

2.13.2.6 Nomor 6

```

1 # In [6]:
2 # visualize tree
3 import graphviz #Mengimport Library Graphviz untuk
    memvisualisasikan decision tree
4 dot_data = tree.export_graphviz(lontar, out_file=None, label="all"
    ", impurity=False, proportion=True,
5                                     feature_names=list(
    papeda_train_att), class_names=["fail", "pass"],
6                                     filled=True, rounded=True) #
    Mendefinisikan dot_data yang isikan akan berisikan data yang
    akan dijadikan gambar
7 graph = graphviz.Source(dot_data) #Memasukkan data tadi menjadi
    sebuah graph
8 graph #Menampilkan graph menggunakan graphviz

```



Gambar 2.292 Nomor 6

2.13.2.7 Nomor 7

```

1 # save tree
2 tree.export_graphviz(lontar, out_file="student-performance.dot",
3     label="all", impurity=False, proportion=True,
4         feature_names=list(papeda_train_att),
5         class_names=["fail", "pass"],
6             filled=True, rounded=True) #Digunakan untuk
7     mengexport graph tree tadi yang telah kita buat

```

```

In [12]: tree.export_graphviz(kue_lontar, out_file="student-performance.dot",
label="all", impurity=False, proportion=True,
...:     feature_names=list(papeda_train_att), class_names=["fail",
"pass"],
...:             filled=True, rounded=True) #Digunakan untuk mengexport
graph tree tadi yang telah kita buat

```

Gambar 2.293 Nomor 7

2.13.2.8 Nomor 8

```

1 lontar.score(papeda_test_att, papeda_test_pass) #Menghitung
prediksi nilai yang akan datang dimasa depan

```

```

ValueError: Found array with 0 sample(s) (shape=(0, 56)) while a minimum of 1 is
required.

```

Gambar 2.294 Nomor 8

Error dikarenakan tidak ada sample data ditemukan

2.13.2.9 Nomor 9

```

1 from sklearn.model_selection import cross_val_score #Mengimport
    fungsi cross_val_score dari library sklearn
2 abon = cross_val_score(lontar, papeda_att, papeda_pass, cv=5) #
    Mendefinisikan abon yang isinya pembagian data menjadi 5
3 # show average score and +/- two standard deviations away (
    covering 95% of scores)
4 print("Accuracy: %0.2f (+/- %0.2f)" % (abon.mean(), abon.std() *
2)) #Menampilkan data nilai dan +/- dari dua standar deviasi

```

```

In [14]: from sklearn.model_selection import cross_val_score #Mengimport fungsi
cross_val_score dari library sklearn
...: abon_gulung = cross_val_score(kue_lontar, papeda_att, papeda_pass, cv=5)
#Mendefinisikan abon_gulung yang isinya pembagian data menjadi 5
...: # show average score and +/- two standard deviations away (covering 95% of
scores)
...: print("Accuracy: %0.2f (+/- %0.2f)" % (abon_gulung.mean(), abon_gulung.std() *
2)) #Menampilkan data nilai dan +/- dari dua standar deviasi
Accuracy: 0.54 (+/- 0.14)

```

Gambar 2.295 Nomor 9

2.13.2.10 Nomor 10

```

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

```

```

Max depth: 8, Accuracy: 0.57 (+/- 0.15)
Max depth: 9, Accuracy: 0.57 (+/- 0.07)
Max depth: 10, Accuracy: 0.58 (+/- 0.08)
Max depth: 11, Accuracy: 0.57 (+/- 0.16)
Max depth: 12, Accuracy: 0.56 (+/- 0.14)
Max depth: 13, Accuracy: 0.58 (+/- 0.13)
Max depth: 14, Accuracy: 0.57 (+/- 0.13)
Max depth: 15, Accuracy: 0.58 (+/- 0.11)
Max depth: 16, Accuracy: 0.55 (+/- 0.12)
Max depth: 17, Accuracy: 0.56 (+/- 0.09)
Max depth: 18, Accuracy: 0.57 (+/- 0.09)
Max depth: 19, Accuracy: 0.56 (+/- 0.14)

```

Gambar 2.296 Nomor 10

2.13.2.11 Nomor 11

```

1 depth_acc = np.empty((19,3), float) #Membuat array baru
2 i = 0 #Membuat variable berisikan 0
3 for max_depth in range(1, 20): #Perulangan untuk memasukkan data
4     lontar = tree.DecisionTreeClassifier(criterion="entropy",
5     max_depth=max_depth) #Membuat decision Tree
6     abon = cross_val_score(lontar, papeda_att, papeda_pass, cv=5)
7     #Mendefinisikan abon yang isinya pembagian data menjadi 5
8     depth_acc[i,0] = max_depth #Memasukkan data max_depth ke
9     array depth_acc
10    depth_acc[i,1] = abon.mean() #Memasukkan data rata-rata dari
11    abon ke array depth_acc
12    depth_acc[i,2] = abon.std() * 2 #Memasukkan data akar 2 dari
13    abon ke array depth_acc
14    i += 1
15
16 depth_acc

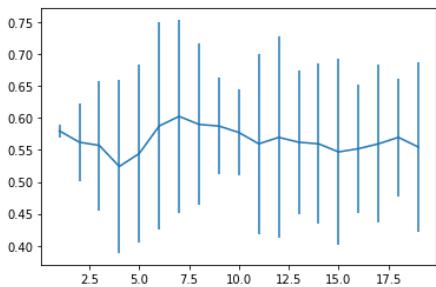
```

```
[8.00000000e+00, 5.89873418e-01, 1.26683504e-01],
[9.00000000e+00, 5.87341772e-01, 7.61179563e-02],
[1.00000000e+01, 5.77215190e-01, 6.71721477e-02],
[1.10000000e+01, 5.59493671e-01, 1.40865091e-01],
[1.20000000e+01, 5.69620253e-01, 1.57695306e-01],
[1.30000000e+01, 5.62025316e-01, 1.12764848e-01],
[1.40000000e+01, 5.59493671e-01, 1.25463410e-01],
[1.50000000e+01, 5.46835443e-01, 1.46223079e-01],
[1.60000000e+01, 5.51898734e-01, 1.00758221e-01],
[1.70000000e+01, 5.59493671e-01, 1.23403115e-01],
[1.80000000e+01, 5.69620253e-01, 9.19792513e-02],
[1.90000000e+01, 5.54430380e-01, 1.33386733e-01]])
```

Gambar 2.297 Nomor 11**2.13.2.12 Nomor 12**

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

```
In [17]: import matplotlib.pyplot as plt #Menimport fungsi pyplot dari library matplotlib
sebagai plt
...: fig, keripik_keladi = plt.subplots() #Membuat plot baru
...: keripik_keladi.errorbar(depth_acc[:,0], depth_acc[:,1], yerr=depth_acc[:,2])
#Mengisikan data plot
...: plt.show() #Menampilkan plot
```

**Gambar 2.298** Nomor 12**2.13.3 Penanganan Error****2.13.3.1 Error**

1. ModuleNotFoundError

```
File "<ipython-input-17-e9072bd6b1ea>", line 1, in <module>
    import graphviz #Mengimport Library Graphviz untuk memvisualisasikan decision tree
ModuleNotFoundError: No module named 'graphviz'
```

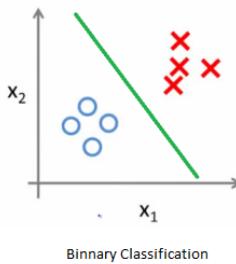
Gambar 2.299 ModuleNotFoundError**2.13.3.2 Solusi**

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

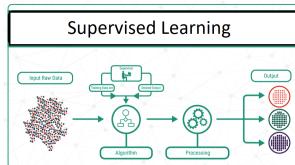
```
1 conda install graphviz
```

2.13.4 Bukti Tidak Plagiat**Gambar 2.300** Bukti Tidak Plagiat**2.14 Muhammad Abdul Gani Wijaya (1174071)****2.14.1 Teori**

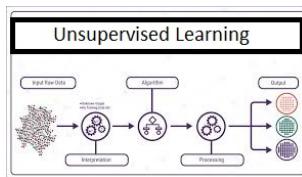
1. Jelaskan apa itu binary classification dilengkapi ilustrasi gambar sendiri
Klasifikasi biner berfungsi untuk mengklasifikasikan elemen-elemen yang diberikan ke dalam dua kelompok berdasarkan aturan klasifikasi.
2. Jelaskan apa itu supervised learning dan unsupervised learning dan clustering dengan ilustrasi gambar sendiri



Gambar 2.301 Binary Classification



Gambar 2.302 Supervised Learning



Gambar 2.303 Unsupervised learning

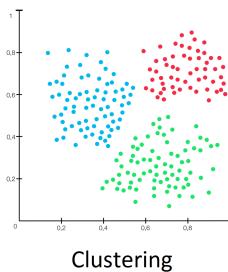
Supervised Learning adalah jenis learning yang memiliki data dari variable input dan variable output, dan menggunakan satu algoritma atau lebih untuk mempelajari fungsi pemetaan dari input ke output.

Unsupervised learning merupakan tipe learning di mana kita hanya mempunyai data masukan (input data) tetapi tidak ada output variable yang berhubungan.

Analisis atau pengelompokan cluster adalah pengelompokan kumpulan objek sedemikian rupa sehingga objek yang mirip dikelompokkan satu sama lain dan begitu juga pada kelompok lain.

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

Evaluasi adalah serapan dari bahasa Inggris yaitu "evaluation" yang diartikan sebagai penaksiran atau penilaian. Nurkancana (1983) menyatakan



Gambar 2.304 Clustering

		Tabel Confusion Matrix	
		True Values	
Prediction	True	True	False
		TP Correct result	FP Unexpected result
Prediction	False	FN Missing result	TN Correct absence of result

Gambar 2.305 Clustering

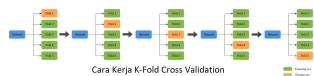
bawa evaluasi adalah kegiatan yang dilakukan berkenaan dengan proses untuk menentukan nilai dari suatu hal.

Akurasi adalah derajat dari kesesuaian, yaitu tingkat yang mana pengukuran adalah tepat ketika dibandingkan dengan nilai absolut.

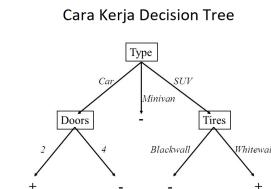
- Jelaskan bagaimana cara membuat dan membaca confusion matrix, buat confusion matrix buatan sendiri.

Confusion matrix merupakan suatu metode yang biasanya digunakan untuk melakukan perhitungan dan akurasi pada konsep data mining atau Sistem Pendukung Keputusan. Pada pengukuran kinerja menggunakan confusion matrix, terdapat 4 (empat) istilah sebagai representasi hasil proses klasifikasi. Keempat istilah tersebut adalah True Positive (TP), True Negative (TN), False Positive (FP) dan False Negative (FN). Nilai True Negative (TN) merupakan jumlah data negatif yang terdeteksi dengan benar, sedangkan False Positive (FP) merupakan data negatif namun terdeteksi sebagai data positif. Sementara itu, True Positive (TP) merupakan data positif yang terdeteksi benar. False Negative (FN) merupakan kebalikan dari True Positive, sehingga data positif, namun terdeteksi sebagai data negatif.

- Jelaskan bagaimana K-fold cross validation bekerja dengan gambar ilustrasi contoh buatan sendiri.



Gambar 2.306 K-fold cross validation



Gambar 2.307 Decision Tree

Cross-validasi, atau juga disebut estimasi rotasi, adalah teknik validasi model untuk menilai bagaimana hasil statistik analisis akan menggeneralisasi kumpulan data independen. Teknik ini utamanya digunakan untuk melakukan prediksi model dan memperkirakan seberapa akurat sebuah model prediktif ketika dijalankan dalam praktiknya. Dalam sebuah masalah prediksi, sebuah model biasanya diberikan kumpulan data (dataset) yang diketahui untuk digunakan dalam menjalankan pelatihan (dataset pelatihan), serta kumpulan data yang tidak diketahui (atau data yang pertama kali dilihat) terhadap model yang diuji (pengujian dataset). Tujuan dari validasi silang adalah untuk mendefinisikan dataset untuk "menguji" model dalam tahap pelatihan (yaitu, validasi data), dalam rangka untuk membatasi masalah seperti terjadinya overfitting, memberikan wawasan tentang bagaimana model akan menggeneralisasi independen dataset (yaitu, dataset tidak diketahui, misalnya dari masalah nyata), dll.

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

Decision tree adalah alat pendukung keputusan yang menggunakan model keputusan seperti pohon dan konsekuensinya yang mungkin, termasuk hasil acara kebetulan, biaya sumber daya, dan utilitas. Ini adalah salah satu cara untuk menampilkan algoritma yang hanya berisi pernyataan kontrol bersyarat.

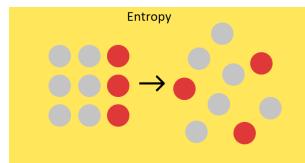
7. Jelaskan apa itu information gain dan entropi dengan gambar ilustrasi buatan sendiri.

$$Gain(T, X) = Entropy(T) - Entropy(T, X)$$

$$\begin{aligned} G(\text{PlayGolf}, \text{Outlook}) &= E(\text{PlayGolf}) - E(\text{PlayGolf}, \text{Outlook}) \\ &= 0.940 - 0.693 = 0.247 \end{aligned}$$

Information Gain

Gambar 2.308 Information Gain



Gambar 2.309 Entropy

Metode Information Gain merupakan metode yang menggunakan teknik scoring untuk pembobotan sebuah fitur yang menggunakan maksimal entropy.

Entropy merupakan salah satu besaran termodinamika yang mengukur energi dalam sistem per satuan temperatur yang tak dapat digunakan untuk melakukan usaha.

2.14.2 scikit-learn

```

1 print(1174071%3)
2 #%% 1.Load Dataset Student
3 import pandas as pd # load dataset (menggunakan student-mat.csv)
4 isekai = pd.read_csv('C:/Users/muham/Downloads/Compressed/KB3C-
    master_3/KB3C-master/src/1174071/2/dataset/student-mat.csv',
    sep=';')
5 #variabel isekai memanggil fungsi untuk read file student-mat.csv
6 len(isekai)
7 #mengetahui jumlah data baris pada data yang dipanggil
8
9 #%% 2.Men erenate binary label (pass/fail) based on G1+G2+G3
10 isekai['pass'] = isekai.apply(lambda row: 1 if (row['G1']+row['G2']+
    ]+row['G3'])>= 35 else 0, axis=1)
11 #mendeklarasikan data pass/fail nya data berdasarkan G1+G2+G3.
12 isekai = isekai.drop(['G1', 'G2', 'G3'], axis=1)
13 #Mengetahui baris G1+G2+G3 ditambahkan, dan hasilnya sama dengan
    35 maka axisnya 1.
14 isekai.head()
15 #Memanggil variabel isekai untuk mengembalikan baris n atas 5
    secara default dari frame atau seri data
16
17 #%% 3.Menggunakan one-hot encoding pada kolom kategori
18 isekai = pd.get_dummies(isekai, columns=['sex', 'school', 'address',
    'famsize', 'Pstatus', 'Mjob', 'Fjob', 'reason', 'guardian', ''

```

```

    schoolsup ','famsup ','paid ','activities ','nursery ','higher ',''
    internet ','romantic'])
19 isekai.head()
20 #memanggil variabel isekai untuk mengembalikan baris n atas 5
    secara default dari frame atau seri data
21
22 #%% 4.shuffle baris
23 isekai = isekai.sample(frac=1)
24 #Memamnggul fungsi sample acak dengan frac=1 pada variabel isekai
25 isekai_train = isekai[:500]
26 isekai_test = isekai[500:]
27 isekai_train_att = isekai_train.drop(['pass'],axis=1)
28 isekai_train_pass = isekai_train['pass']
29 isekai_test_att = isekai_test.drop(['pass'],axis=1)
30 isekai_test_pass = isekai_test['pass']
31 isekai_att = isekai.drop(['pass'],axis=1)
32 isekai_pass = isekai['pass']
33
34 import numpy as np #Import modul numpy sebagai dengan alias np
35 print("Passing: %d out %d (%.2f%%)" %(np.sum(isekai_pass),len(
    isekai_pass),100*float(np.sum(isekai_pass))/len(isekai_pass)))
36 #%% 5.fit a decision tree
37 from sklearn import tree
38 raftel = tree.DecisionTreeClassifier(criterion="entropy",
    max_depth=5)
39 #Membuat variabel raftel sebagai decisiontree , dengan criterion
    fungsi mengukur kualitas split
40 raftel = raftel.fit(isekai_train_att,isekai_train_pass)
41
42 #%% 6.visualize tree
43 import os
44 os.environ["PATH"] += os.pathsep + 'D:/graphviz-2.38/release/bin'
45
46 import graphviz
47 enies = tree.export_graphviz(raftel,out_file=None,label ="all",
    impurity=False, proportion=True, feature_names=list(
        isekai_train_att), class_names=["fail","pass"], filled=True,
    rounded=True)
48 #Mengubah data menjadi grafik
49 lobby = graphviz.Source(emies)
50 lobby
51
52 #%% 7.save tree
53 tree.export_graphviz(raftel,out_file="student-performance.dot",
    label ="all",impurity=False, proportion=True, feature_names=
    list(isekai_train_att), class_names=["fail","pass"], filled=
    True, rounded=True)
54 #Meng export data dari graphviz ke file student-performance.dot
55
56 #%% 8
57 raftel.score(isekai_test_att,isekai_test_pass)
58 #Memprediksi dengan memberikan beberapa data baru
59
60 #%% 9
61 from sklearn.model_selection import cross_val_score

```

```
| File "cipython-input-6-9636e2e0f000", line 4, in <module>
|     import graphviz
| 
| ModuleNotFoundError: No module named 'graphviz'
```

Gambar 2.310 Module Not Found

```
62 skypiea = cross_val_score(raftel ,isekai_att ,isekai_pass ,cv=5)
63 #Mengevaluasi score menggunakan validasi silang
64 print("Accuracy : %0.2f (+/- %0.2f)" % (skypiea.mean() ,skypiea.
   std() * 2))
65
66 #%% 10
67 for water in range(1,20):
68     raftel = tree.DecisionTreeClassifier(criterion="entropy",
69     max_depth=water)
70     skypiea = cross_val_score(raftek ,isekai_att ,isekai_pass ,cv=5)
71     print("Max depth : %d, Accuracy : %0.2f (+/- %0.2f)" %(arabasta ,skypiea.mean() ,skypiea.std() * 2))
72 #Menunjukkan data tree. Semakin dalam tree , semakin banyak
#perpecahan yang dimilikinya dan menangkap lebih banyak
#informasi tentang data .
73 #%% 11
74 seven = np.empty((19,3),float)
75 wano = 0
76 for water in range(1,20):
77     raftel = tree.DecisionTreeClassifier(criterion="entropy",
78     max_depth=water)
79     #Membuat rabel raftel untuk decision tree dengan
#ketentuan entropy
80     skypiea = cross_val_score(raftel ,isekai_att ,isekai_pass ,cv=5)
81     seven[wano,0] = water
82     seven[wano,1] = skypiea.mean()
83     seven[wano,2] = skypiea.std() * 2
84     wano += 1
85     seven
86 #%% 12
87 import matplotlib.pyplot as plt
88 blitar , kediri = plt.subplots()
89 kediri.errorbar(seven[:,0],seven[:,1],yerr=seven[:,2])
90 #Membuat error pada bar kemudian grafik akan ditampilkan
#menggunakan show
91 plt.show()
```

2.14.3 Pengangan Error

1. Screenshot Error

2. Jenis Error

- Module Not Found

3. Solusi Error



Gambar 2.311 Install graphviz



Gambar 2.312 Scan plagiarisme

▪ Module Not Found

Mendownload library graphviz menggunakan pip install graphviz di Anaconda Prompt

2.14.4 Scan Plagiarisme

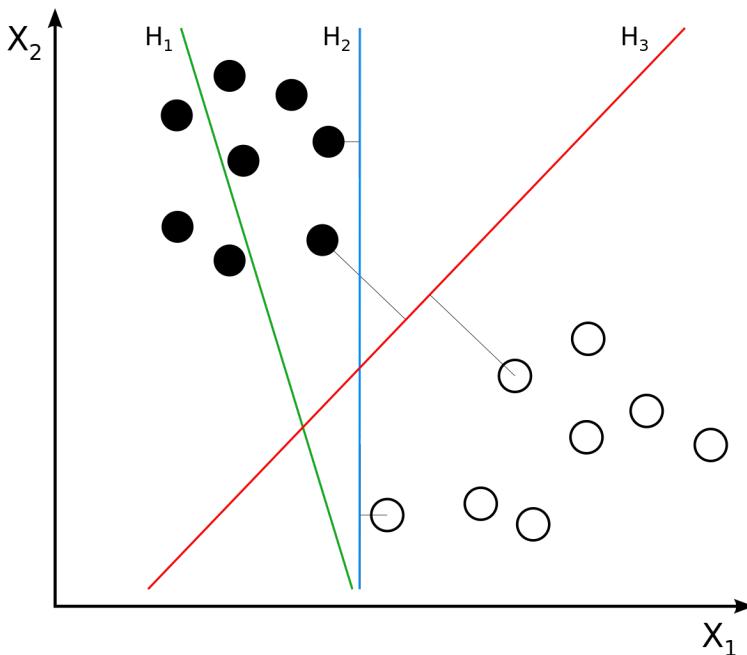
2.14.5 Link Youtube

<https://youtu.be/1RX-a3SOLt4>

2.15 Handi Hermawan (1174080)

2.15.1 Teori (Membangun Model Prediksi)

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



Gambar 2.313 Binary Classification

1. Tes Medis

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

2. Metode Uji "lulus atau gagal" atau "kontrol kualitas di pabrik"

yaitu memutuskan apakah suatu spesifikasi telah atau belum terpenuhi - klasifikasi Go/no go.

3. Pengambilan informasi

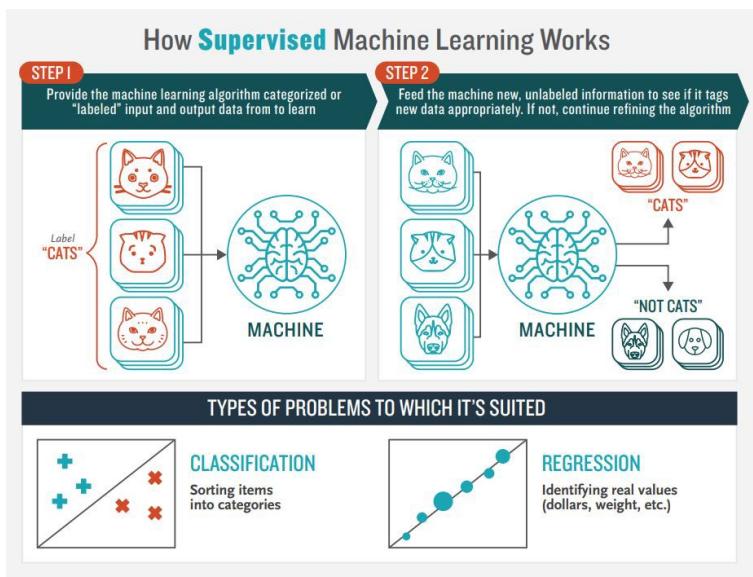
yaitu memutuskan apakah suatu halaman atau artikel harus dalam hasil pencarian atau tidak - properti klasifikasi adalah relevansi artikel, atau kegunaannya bagi pengguna.

2.15.1.2 Supervised Learning , Unsupervised Learning dan Clustering dengan ilustrasi gambar

1. Supervised Learning

Supervised Learning dalam bahasa indonesia adalah pembelajaran yang ada supervisornya. Maksud disini ada supervisornya adalah label di tiap data nya. Label maksudnya adalah tag dari data yang ditambahkan

dalam machine learning model. Contohnya gambar kucing di tag “kucing” di tiap masing masing image kucing dan gambar anjing di tag “anjing” di tiap masing gambar anjing. Machine learning kategori dapat berupa clasification (“anjing”, “kucing”, “beruang”, dsb) dan regression (berat badan, tinggi badan dsb). Supervised learning banyak digunakan dalam memprediksi pola dimana pola tersebut sudah ada contoh data yang lengkap, jadi pola yang terbentuk adalah hasil pembelajaran data lengkap tersebut. Tentunya jika kita memasukan data baru, setelah kita melakukan ETL (Extract Transform Load) maka kita mendapat info feature feature dari sample baru tersebut. Kemudian dari feature feature tersebut di compare dengan pattern clasification dari model yang didapat dari labeled data. Setiap label akan dicompare sampai selesai, dan yang memiliki percentage lebih banyak akan diambil sebagai prediksi akhir.

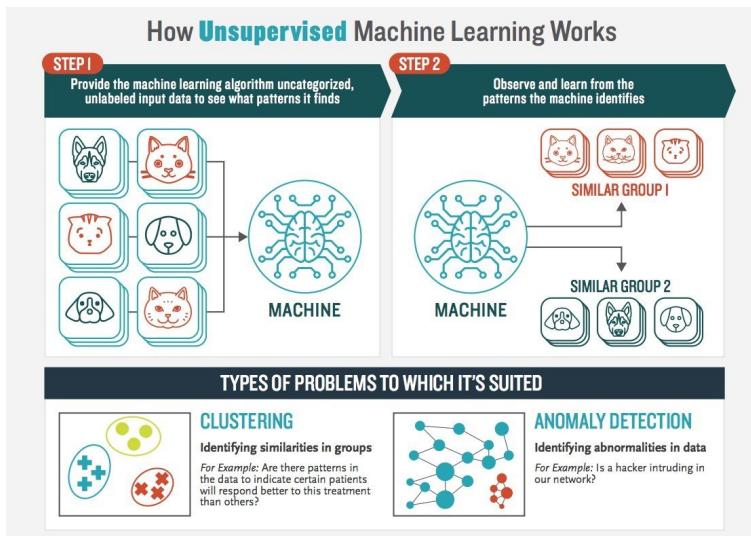


Gambar 2.314 Supervised Learning

2. Unsupervised Learning

Unsupervised learning memiliki keunggulan dari unsupervised learning. Jika unsupervised learning memiliki label sebagai dasar prediksi baik serta membuat clasification dan regression algorithm memungkinkan. Tetapi dalam realitanya, data real itu banyak yang tidak memiliki label. Label kebanyakan jika data sudah masuk ke ERP apapun bentuk ERPnya dan bagaimana kalo datanya berupa natural input seperti suara, gambar, dan video. Unsupervised learning tidak menggunakan label dalam memprediksi target feautures / variable. Melainkan menggunakan ke samaan

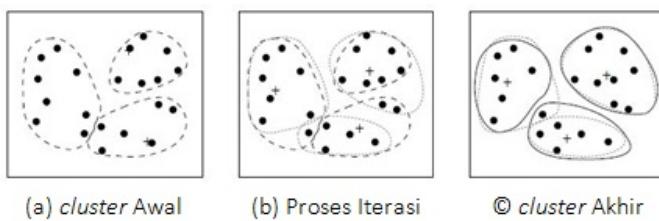
dari attribut attribut yang dimiliki. Jika attribut dan sifat sifat dari data data feature yang diekstrak memiliki kemiripan miripan, maka akan dikelompok kelompokan (clustering). Sehingga hal ini akan menimbulkan kelompok kelompok (cluster). Jumlah cluster bisa unlimited. Dari kelompok kelompok itu model melabelkan, dan jika data baru mau di prediksi, maka akan dicocok kan dengan kelompok yang mirip mirip featurenya.



Gambar 2.315 Unsupervised Learning

3. Clustering

Clustering atau klasterisasi adalah metode pengelompokan data. Menurut Tan, 2006 clustering adalah sebuah proses untuk mengelompokan data ke dalam beberapa cluster atau kelompok sehingga data dalam satu cluster memiliki tingkat kemiripan yang maksimum dan data antar cluster memiliki kemiripan yang minimum.



Gambar 2.316 Clustering

2.15.1.3 Evaluasi dan Akurasi Evaluasi adalah tentang bagaimana kita dapat mengevaluasi seberapa baik model bekerja dengan mengukur tingkat akurasinya. Dan akurasi akan didefinisikan sebagai persentase kasus yang diklasifikasikan dengan benar. Kita dapat menganalisis kesalahan yang dibuat oleh model, atau tingkat kebingungannya, menggunakan matriks kebingungan (confusion matrix). Matriks kebingungan mengacu pada kebingungan dalam model, tetapi matriks kebingungan ini bisa menjadi sedikit sulit untuk dipahami ketika mereka menjadi sangat besar.

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

Gambar 2.317 Evaluasi

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

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

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

Gambar 2.318 Contoh Confusion Matrix

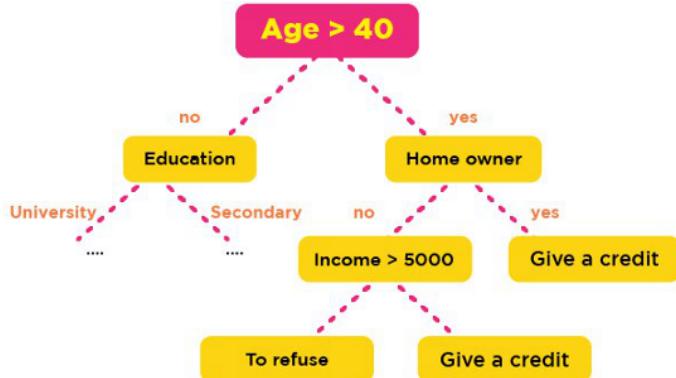
2.15.1.5 Bagaimana Fold-cross validation Cross-validation (CV) adalah metode statistik yang dapat digunakan untuk mengevaluasi kinerja model atau algoritma dimana data dipisahkan menjadi dua subset yaitu data proses pembelajaran dan data validasi / evaluasi. Model atau algoritma dilatih oleh subset pembelajaran dan divalidasi oleh subset validasi. Selanjutnya pemilihan jenis CV dapat didasarkan pada ukuran dataset. Biasanya CV K-fold digunakan karena dapat mengurangi waktu komputasi dengan tetap menjaga keakuratan estimasi.

Gambar 1 – Skema 10 fold CV

Gambar 2.319 Contoh Fold-Cross Validation

1. 10 fold CV adalah salah satu K fold CV yang direkomendasikan untuk pemilihan model terbaik karena cenderung memberikan estimasi akurasi yang kurang bias dibandingkan dengan CV biasa, leave-one-out CV dan bootstrap. Dalam 10 fold CV, data dibagi menjadi 10 fold berukuran kira-kira sama, sehingga kita memiliki 10 subset data untuk mengevaluasi kinerja model atau algoritma. Untuk masing-masing dari 10 subset data tersebut, CV akan menggunakan 9 fold untuk pelatihan dan 1 fold untuk pengujian seperti diilustrasikan pada Gambar diatas.

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



Gambar 2.320 Contoh Decision Tree

2.15.1.7 Information Gain dan Entropi

- Information Gain

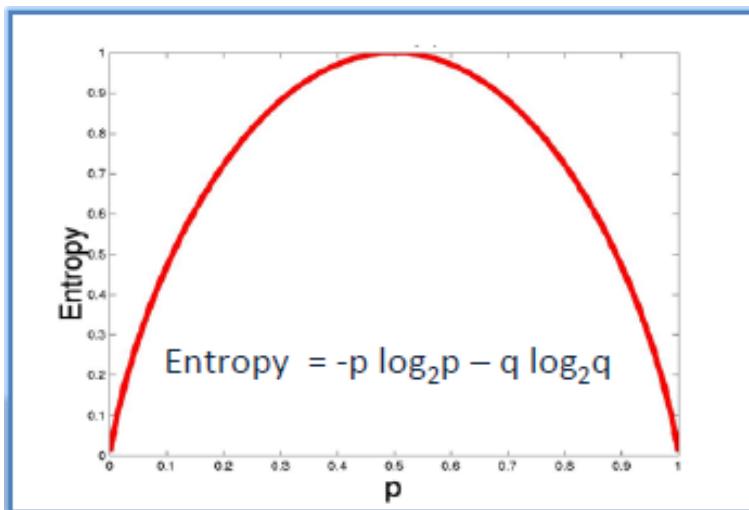
Information Gain merupakan total data yang didapat dari data - data acak yang data tersebut akan digunakan untuk analisis data lainnya. Information Gain ini digunakan pada decision tree sebagai label setiap aksi - aksi yang perlu dinilai validasinya.



Gambar 2.321 Information Gain

- Entropi

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



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

Gambar 2.322 Entropi

2.15.2 Praktek

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

```

1 # In [0]
2 1174066 % 3 #Hasilnya 1 maka akan menggunakan nama Kota

```

2.15.2.1 Nomor 1

```

1 import pandas as pd #Import library pandas menggantinya nama yang
    akan dipanggil jadi pd
2 tokyo = pd.read_csv('dataset/student-mat.csv', sep=';') #Membuat
    variabel tokyo yang isinya memanggil fungsi membaca file csv
3 len(tokyo) #Menghitung jumlah data yang ada pada csv yang tadi
    sudah dibaca

```

The screenshot shows a Jupyter Notebook interface. On the left, a code cell contains Python code for importing a CSV file and creating a binary 'pass' column based on the sum of test scores. On the right, a 'Help' pop-up window provides information on using the help function.

```

1 # In [1]:
2 # Create a new file
3 # Created in Sun Mar  4 12:50:42 2018
4 # Author: Ats
5 #
6 # In [2]:
7 1174866 8 $ download 3 make about meepgamer home.html
8
9 # In [3]:
10 import pandas as pd
11 dataset = pd.read_csv('dataset/student-mat.csv', sep=',')
12 dataset.head()

```

Gambar 2.323 Nomor 1

2.15.2.2 Nomor 2

```

1 # In [2]
2 # generate binary label (pass/fail) based on G1+G2+G3 (test
3 #   grades, each 0–20 pts); threshold for passing is sum>=30
4 tokyo['pass'] = tokyo.apply(lambda row: 1 if (row['G1']+row['G2']
5     )+row['G3']) >= 35 else 0, axis=1) #Membuat label binary (
6 #   pass/fail) berdasarkan G1+G2+G3 (testgrade, semuanya 0–20
7 #   point); Batas untuk pass adalah sum>=30
8 tokyo = tokyo.drop(['G1', 'G2', 'G3'], axis=1) #Meghilangkan data
9 G1 G2 dan G3
10 tokyo.head() #Menampilkan data

```

The screenshot shows a Jupyter Notebook interface. On the left, a code cell contains Python code for generating a 'pass' column and dropping 'G1', 'G2', and 'G3' columns. On the right, the resulting DataFrame is displayed.

```

# In [2]:
tokyo['pass'] = tokyo.apply(lambda row: 1 if (row['G1']+row['G2']+row['G3']) >= 35 else 0, axis=1)
tokyo.head()

# In [3]:
In [3]: import pandas as pd
...: dataset = pd.read_csv('dataset/student-mat.csv', sep=',')
Out[3]: None

# In [4]:
def get_pass(row):
    if (row['G1']+row['G2']+row['G3']) >= 35 else 0, axis=1)
        tokyo['pass'] = tokyo.apply(get_pass, axis=1)
        tokyo.head()

File "c:\python\python37-32\lib\site-packages\ipykernel\ipkernel.py", line 1, in module
    import ipykernel.ipkernel
ModuleNotFoundError: No module named 'ipykernel'

# In [5]:
In [5]: tokyo['pass'] = tokyo.apply(lambda row: 1 if (row['G1']+row['G2']+row['G3']) >= 35 else 0, axis=1)
...: tokyo = tokyo.drop(['G1', 'G2', 'G3'], axis=1)

Out[5]:
sex age address family ... Dalc heatt absences pass
1 G 17 U R3 ... 1 1 2 4 0
1 G 17 U R3 ... 1 1 2 4 0
1 G 17 U R3 ... 1 1 2 4 0
1 G 17 U R3 ... 1 1 2 4 0
1 G 17 U R3 ... 1 1 2 4 0
[4 rows x 11 columns]

```

Gambar 2.324 Nomor 2

2.15.2.3 Nomor 3

```

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

```

```
# In [1]:
# Import library
import pandas as pd
# Read data
tokyo = pd.read_csv('tokyo.csv')
# Select columns
columns=['sex', 'school', 'address', 'family', 'status', 'age', 'religion',
          'nursery', 'higher', 'internet', 'romantic']
# Head
tokyo.head()

# Out[1]:
In [1]: tokyo = pd.read_csv('tokyo.csv', columns=['sex', 'school', 'address', 'family', 'status', 'age', 'religion', 'nursery', 'higher', 'internet', 'romantic'])
...: tokyo.head()
...: 
...: Out[1]:
   sex school address family status age religion nursery higher internet romantic
0   M   Pedu. Feda ... Internet_no romantic_no
1   M    L  1  1  ...  1  1  0
2   M    L  1  1  ...  1  1  0
3   M    L  1  1  ...  1  1  0
4   M    L  1  1  ...  1  1  0
... ...
[5 rows x 17 columns]
In [1]:
```

Gambar 2.325 Nomor 3

2.15.2.4 Nomor 4

```
# In [4]:
# shuffle rows
tokyo = tokyo.sample(frac=1) #Mengambil data sample dari tokyo
# split training and testing data
tokyo_train = tokyo[:500] #Membagi data untuk training
tokyo_test = tokyo[500:] #Membagi data untuk test
# drop pass column
tokyo_train_att = tokyo_train.drop(['pass'], axis=1) #Meghapus data yang telah pass dan memasukkannya
tokyo_train_pass = tokyo_train['pass'] #Mengambil data yang pass saja
# drop pass column
tokyo_test_att = tokyo_test.drop(['pass'], axis=1) #Meghapus data yang telah pass dan memasukkannya
tokyo_test_pass = tokyo_test['pass'] #Mengambil data yang pass saja
# drop pass column
tokyo_att = tokyo.drop(['pass'], axis=1) #Meghapus data yang telah pass dan memasukkannya
tokyo_pass = tokyo['pass'] #Mengambil data yang pass saja
# number of passing students in whole dataset:
import numpy as np #Mengimport library numpy sebagai np
print("Passing: %d out of %d (%.2f%%)" % (np.sum(tokyo_pass), len(tokyo_pass), 100*float(np.sum(tokyo_pass)) / len(tokyo_pass)))
#Menampilkan data
```

```
# In [2]:
# Import library
import pandas as pd
# Read data
tokyo = pd.read_csv('tokyo.csv')
# Select columns
columns=['sex', 'school', 'address', 'family', 'status', 'age', 'religion',
          'nursery', 'higher', 'internet', 'romantic']
# Head
tokyo.head()

# Out[2]:
In [2]: tokyo = pd.read_csv('tokyo.csv')
...: tokyo = tokyo.sample(frac=1)
...: tokyo_train = tokyo[:500]
...: tokyo_test = tokyo[500:]
...: tokyo_train_att = tokyo_train.drop(['pass'], axis=1)
...: tokyo_train_pass = tokyo_train['pass']
...: tokyo_test_att = tokyo_test.drop(['pass'], axis=1)
...: tokyo_test_pass = tokyo_test['pass']
...: tokyo_att = tokyo.drop(['pass'], axis=1)
...: tokyo_pass = tokyo['pass']
...
...: # drop pass column
...: tokyo_train_att = tokyo_train.drop(['pass'], axis=1)
...: tokyo_train_pass = tokyo_train['pass']
...: tokyo_test_att = tokyo_test.drop(['pass'], axis=1)
...: tokyo_test_pass = tokyo_test['pass']
...: tokyo_att = tokyo.drop(['pass'], axis=1)
...: tokyo_pass = tokyo['pass']
...
...: # number of passing students in whole dataset:
...: import numpy as np
...: print("Passing: %d out of %d (%.2f%%)" % (np.sum(tokyo_pass), len(tokyo_pass), 100*float(np.sum(tokyo_pass)) / len(tokyo_pass)))
...: 
...: Out[2]:
ModuleNotFoundError: No module named 'graphviz'
In [3]:
# Import library
import pandas as pd
# Read data
tokyo = pd.read_csv('tokyo.csv')
# Select columns
columns=['sex', 'school', 'address', 'family', 'status', 'age', 'religion',
          'nursery', 'higher', 'internet', 'romantic']
# Head
tokyo.head()

# Out[3]:
In [3]: tokyo = pd.read_csv('tokyo.csv')
...: tokyo = tokyo.sample(frac=1)
...: tokyo_train = tokyo[:500]
...: tokyo_test = tokyo[500:]
...: tokyo_train_att = tokyo_train.drop(['pass'], axis=1)
...: tokyo_train_pass = tokyo_train['pass']
...: tokyo_test_att = tokyo_test.drop(['pass'], axis=1)
...: tokyo_test_pass = tokyo_test['pass']
...: tokyo_att = tokyo.drop(['pass'], axis=1)
...: tokyo_pass = tokyo['pass']
...
...: # drop pass column
...: tokyo_train_att = tokyo_train.drop(['pass'], axis=1)
...: tokyo_train_pass = tokyo_train['pass']
...: tokyo_test_att = tokyo_test.drop(['pass'], axis=1)
...: tokyo_test_pass = tokyo_test['pass']
...: tokyo_att = tokyo.drop(['pass'], axis=1)
...: tokyo_pass = tokyo['pass']
...
...: # number of passing students in whole dataset:
...: import numpy as np
...: print("Passing: %d out of %d (%.2f%%)" % (np.sum(tokyo_pass), len(tokyo_pass), 100*float(np.sum(tokyo_pass)) / len(tokyo_pass)))
...: 
...: Out[3]:
Passing: 146 out of 399 (%.4f%%)
```

Gambar 2.326 Nomor 4

2.15.2.5 Nomor 5

```
# In [5]:
# fit a decision tree
from sklearn import tree #import Decision tree dari library
sklearn
```

```

4 kyoto = tree.DecisionTreeClassifier(criterion="entropy",
5 max_depth=5) #Membuat decision tree dengan maximal depthnya 5
6 kyoto = kyoto.fit(tokyo_train_att, tokyo_train_pass) #Memasukkan
    data yang akan dijadikan decision treenya

```

```

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

```

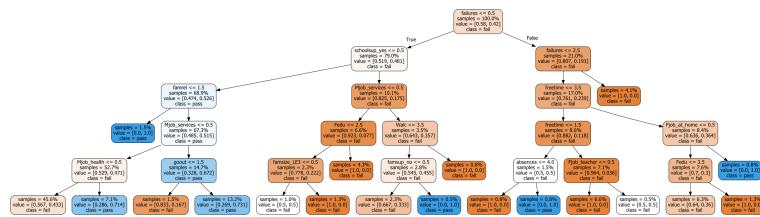
Gambar 2.327 Nomor 5

2.15.2.6 Nomor 6

```

1 # In [6]:
2 # visualize tree
3 import graphviz #Mengimport Library Graphviz untuk
    memvisualisasikan decision tree
4 dot_data = tree.export_graphviz(kyoto, out_file=None, label="all",
    , impurity=False, proportion=True,
5                                         feature_names=list(
        tokyo_train_att), class_names=["fail", "pass"],
6                                         filled=True, rounded=True) #
    Mendefinisikan dot_data yang isikan akan berisikan data yang
    akan dijadikan gambar
7 graph = graphviz.Source(dot_data) #Memasukkan data tadi menjadi
    sebuah graph
8 graph #Menampilkan graph menggunakan graphviz

```



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

Gambar 2.329 Nomor 7

2.15.2.8 Nomor 8

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

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

Gambar 2.330 Nomor 8

2.15.2.9 Nomor 9

```
1 # In [9]:
2 from sklearn.model_selection import cross_val_score #Mengimport
  fungsi cross_val_score dari library sklearn
3 nagoya = cross_val_score(kyoto, tokyo_att, tokyo_pass, cv=5) #
  Mendefinisikan nagoya yang isinya pembagian data menjadi 5
4 # show average score and +/- two standard deviations away (
  covering 95% of scores)
5 print("Accuracy: %.2f (+/- %.2f)" % (nagoya.mean(), nagoya.std()
  () * 2)) #Menampilkan data nilai dan +/- dari dua standar
  deviasi
```

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

Gambar 2.331 Nomor 9

2.15.2.10 Nomor 10

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

```
In [70]: for max_depth in range(1, 20):
...:     kyoto = tree.DecisionTreeClassifier(criterion="entropy", max_depth=max_depth)
...:     nagoya = cross_val_score(kyoto, tokyo_att, tokyo_pass, cv=5)
...:     print("Max depth: %d, Accuracy: %0.2f (+/- %0.2f)" % (max_depth, nagoya.mean(), nagoya.std() * 2))
Max depth: 1, Accuracy: 0.56 (+/- 0.01)
Max depth: 2, Accuracy: 0.56 (+/- 0.08)
Max depth: 3, Accuracy: 0.56 (+/- 0.12)
Max depth: 4, Accuracy: 0.57 (+/- 0.08)
Max depth: 5, Accuracy: 0.55 (+/- 0.09)
Max depth: 6, Accuracy: 0.57 (+/- 0.18)
Max depth: 7, Accuracy: 0.57 (+/- 0.15)
Max depth: 8, Accuracy: 0.54 (+/- 0.18)
Max depth: 9, Accuracy: 0.54 (+/- 0.11)
Max depth: 10, Accuracy: 0.58 (+/- 0.11)
Max depth: 11, Accuracy: 0.56 (+/- 0.10)
Max depth: 12, Accuracy: 0.60 (+/- 0.08)
Max depth: 13, Accuracy: 0.59 (+/- 0.05)
Max depth: 14, Accuracy: 0.58 (+/- 0.05)
Max depth: 15, Accuracy: 0.57 (+/- 0.07)
Max depth: 16, Accuracy: 0.58 (+/- 0.06)
Max depth: 17, Accuracy: 0.60 (+/- 0.11)
Max depth: 18, Accuracy: 0.58 (+/- 0.08)
Max depth: 19, Accuracy: 0.58 (+/- 0.06)
```

Gambar 2.332 Nomor 10

2.15.2.11 Nomor 11

```
1 # In[11]:
2 depth_acc = np.empty((19,3), float) #Membuat array baru
3 i = 0 #Membuat variable berisikan 0
4 for max_depth in range(1, 20): #Perulangan untuk memasukkan data
5     kyoto = tree.DecisionTreeClassifier(criterion="entropy",
6     max_depth=max_depth)#Membuat decision Tree
7     nagoya = cross_val_score(kyoto, tokyo_att, tokyo_pass, cv=5)
8     #Mendefinisikan nagoya yang isinya pembagian data menjadi 5
9     depth_acc[i,0] = max_depth #Memasukkan data max_depth ke
10    array depth_acc
11    depth_acc[i,1] = nagoya.mean() #Memasukkan data rata-rata
12    dari nagoya ke array depth_acc
13    depth_acc[i,2] = nagoya.std() * 2 #Memasukkan data akar 2
14    dari nagoya ke array depth_acc
15    i += 1
16
17 depth_acc
```

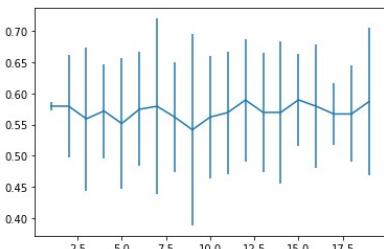
```
In [71]: depth_acc = np.empty((19,3), float)
...: i = 0
...: for max_depth in range(1, 20):
...:     kyoto = tree.DecisionTreeClassifier(criterion="entropy", max_depth=max_depth)
...:     nagoya = cross_val_score(kyoto, tokyo_att, tokyo_pass, cv=5)
...:     depth_acc[i,0] = max_depth
...:     depth_acc[i,1] = nagoya.mean()
...:     depth_acc[i,2] = nagoya.std() * 2
...:     i += 1
...:
...:
...: depth_acc
Out[71]:
array([[1.00000000e+00, 5.79751704e-01, 6.30768599e-03],
       [2.00000000e+00, 5.79654333e-01, 8.25005697e-02],
       [3.00000000e+00, 5.54241318e-01, 1.21808510e-01],
       [4.00000000e+00, 5.71964460e-01, 8.72318860e-02],
       [5.00000000e+00, 5.46678027e-01, 9.66616528e-02],
       [6.00000000e+00, 5.69561819e-01, 1.08113451e-01],
       [7.00000000e+00, 5.67030996e-01, 1.40406275e-01],
       [8.00000000e+00, 5.47030996e-01, 1.22447311e-01],
       [9.00000000e+00, 5.51966082e-01, 6.68884125e-02],
       [1.00000000e+01, 5.87314184e-01, 7.12478298e-02],
       [1.10000000e+01, 5.77124310e-01, 7.61119153e-02],
       [1.20000000e+01, 5.89719247e-01, 4.34452239e-02],
       [1.30000000e+01, 5.97569783e-01, 3.87484760e-02],
       [1.40000000e+01, 5.77026939e-01, 7.68881184e-02],
       [1.50000000e+01, 5.97347452e-01, 5.69404888e-02],
       [1.60000000e+01, 5.74720058e-01, 1.290008478e-01],
       [1.70000000e+01, 5.82282538e-01, 5.71762018e-02],
       [1.80000000e+01, 5.74720870e-01, 3.56163418e-02],
       [1.90000000e+01, 5.77250893e-01, 8.75345835e-02]])
```

Gambar 2.333 Nomor 11

2.15.2.12 Nomor 12

```
1 # In [12]:
2 import matplotlib.pyplot as plt #Menimport fungsi pyplot dari
   library matplotlib sebagai plt
3 fig, ax = plt.subplots() #Membuat plot baru
4 ax.errorbar(depth_acc[:,0], depth_acc[:,1], yerr=depth_acc[:,2])
   #Mengisikan data plot
5 plt.show() #Menampilkan plot
```

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



Gambar 2.334 Nomor 12

2.15.3 Penanganan Error

2.15.3.1 Error

1. ModuleNotFoundError

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

Gambar 2.335 ModuleNotFoundError

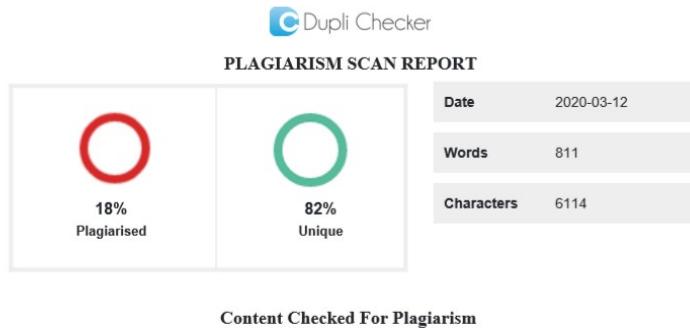
2.15.3.2 Solusi

1. Intall library Graphviz dengan cara download graphviz di google

setelah instalasi buka anaconda prompt sebagai admin lalu mengetikkan

```
1 conda install graphviz
```

2.15.4 Bukti Tidak Plagiat



Handi Hermawan 1174080 Klasifikasi biner (Binary Classification) atau binomial adalah tugas untuk mengklasifikasikan elemen-elemen dari himpunan tertentu ke dalam dua kelompok (memprediksi kelompok mana yang masing-masing dimiliki) berdasarkan aturan klasifikasi. Konteks yang membutuhkan keputusan apakah suatu item memiliki sifat kualitatif atau tidak, beberapa karakteristik tertentu, atau beberapa klasifikasi biner khas. Tes Medis untuk menentukan apakah pasien memiliki penyakit tertentu atau tidak - properti klasifikasi adalah keberadaan penyakit. Metode Uji "lulus atau gagal" atau "kontrol kualitas di pabrik" yaitu memutuskan apakah suatu spesifikasi telah atau belum terpenuhi - klasifikasi Go/no go. Pengambilan informasi yaitu memutuskan apakah suatu halaman atau artikel harus dalam basis pencarian atau tidak - properti klasifikasi adalah relevansi artikel, atau kegunaannya bagi pengguna. Supervised Learning dalam bahasa

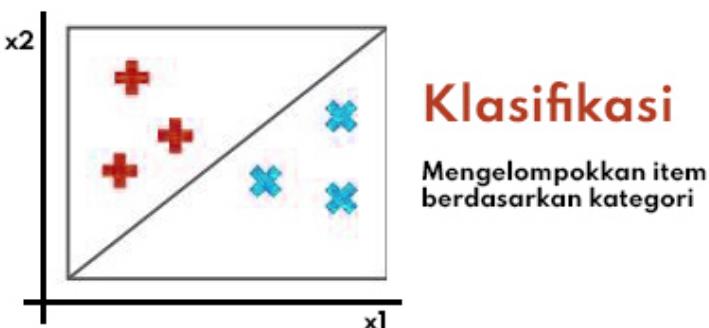
Gambar 2.336 Bukti Tidak Plagiat

2.15.5 Link Youtube:

<https://youtu.be/QL05oLFErFk>

2.16 Advent Nopele Olansi Damiahan Sihite(1174089)

2.16.1 Teori



Gambar 2.337 Binary Classification

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

1. Tes medis

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

2. Metode uji "lulus atau gagal"

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

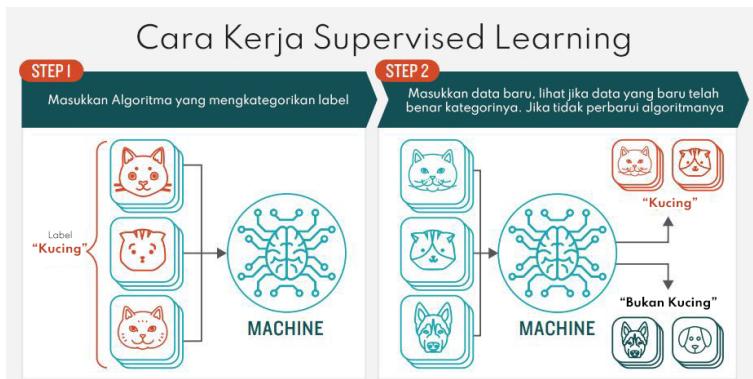
3. Pengambilan informasi

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

2.16.1.2 Supervised Learning , Unsupervised Learning dan Clustering

1. Supervised Learning

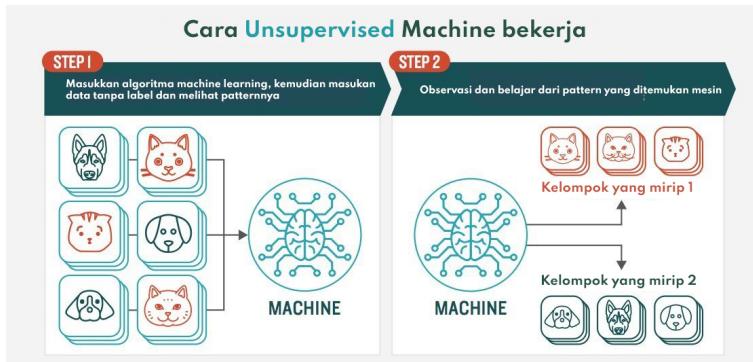
Supervised learning adalah suatu pembelajaran yang terawasi dimana jika output yang diharapkan telah diketahui sebelumnya. Biasanya pembelajaran ini dilakukan dengan menggunakan data yang telah ada. Dan Supervised Learning dalam bahasa indonesia adalah pembelajaran yang ada supervisornya. Maksud disini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



Gambar 2.338 Supervised Learning

2. Unsupervised Learning

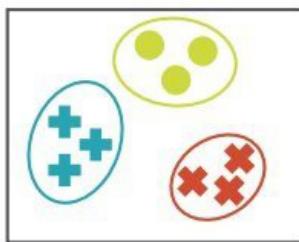
Unsupervised learning menggunakan kemiripan dari attribut yang dimiliki suatu item. Jika attribut dan sifat dari data yang diekstrak memiliki kemiripan, maka akan dikelompokkan (clustering). Sehingga hal ini akan menimbulkan kelompok (cluster). Jumlah cluster bisa unlimited. Dari kelompok-kelompok itu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



Gambar 2.339 Unsupervised Learning

3. Clustering

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



CLUSTERING

Melihat kemiripan dari setiap item dan mengelompokkannya ke sebuah group

Gambar 2.340 Clustering

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

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

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

Gambar 2.341 Evaluasi

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

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

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

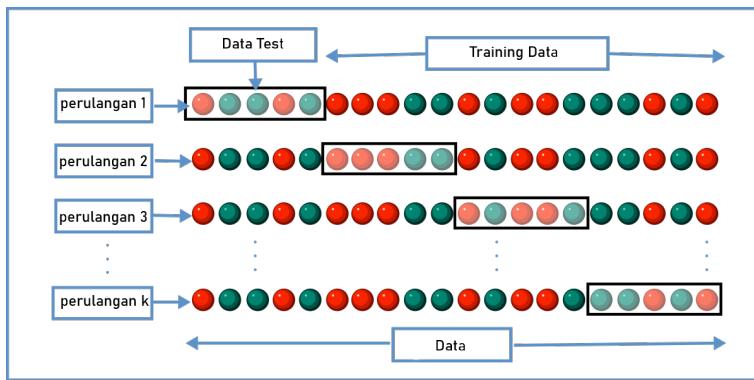
Gambar 2.342 Contoh Confusion Matrix

2.16.1.5 Bagaimana K-fold cross validation bekerja

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

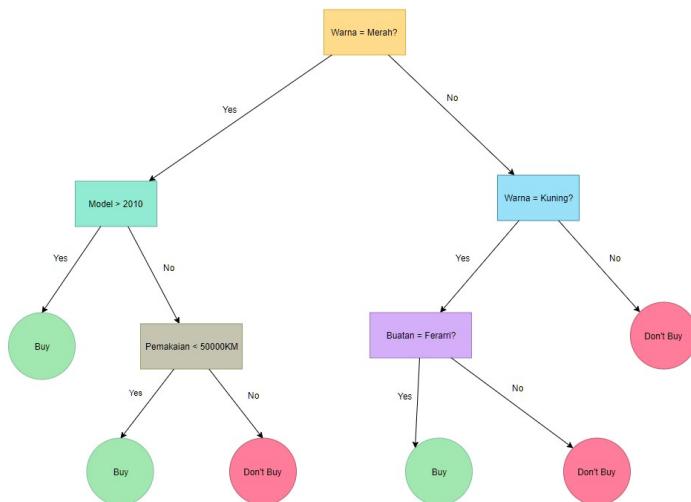
6. Dan seterusnya hingga habis mencapai fold ke-K.

7. Terakhir hitung rata-rata akurasi K buah.



Gambar 2.343 Contoh K-Fold Cross Validation

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



Gambar 2.344 Contoh Decision Tree membeli mobil

2.16.1.7 Information Gain dan Entropi

- Information Gain

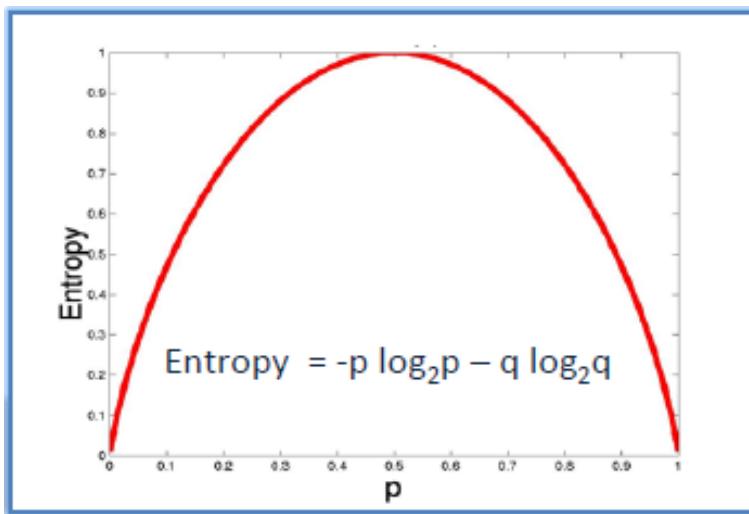
Information Gain merupakan total data yang didapat dari data - data acak yang data tersebut akan digunakan untuk analisis data lainnya. Information Gain ini digunakan pada decision tree sebagai label setiap aksi - aksi yang perlu dinilai validasinya.



Gambar 2.345 Information Gain

- Entropi

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



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

Gambar 2.346 Entropi

2.16.2 Praktek

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

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

2.16.2.1 Nomor 1

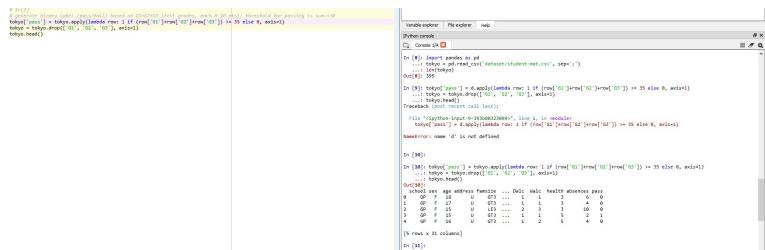
```
1 tokyo = tokyo.drop(['G1', 'G2', 'G3'], axis=1) #Menghilangkan data
      G1 G2 dan G3
2 tokyo.head() #Menampilkan data
```

The screenshot shows a Jupyter Notebook interface. On the left, there are two code cells. The first cell contains the code to drop columns 'G1', 'G2', and 'G3' from the 'tokyo' DataFrame. The second cell contains the command to display the first few rows of the DataFrame. On the right, the output pane shows a message indicating that no output was generated for the second cell.

Gambar 2.347 Nomor 1

2.16.2.2 Nomor 2

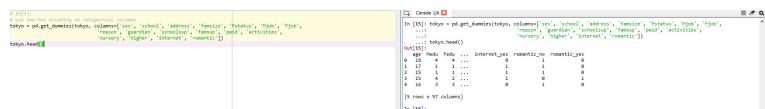
```
1 # use one-hot encoding on categorical columns
2 tokyo = pd.get_dummies(tokyo, columns=['sex', 'school', 'address',
3     'famsize', 'Pstatus', 'Mjob', 'Fjob',
4     'reason', 'guardian', 'schoolsupsup',
5     'famsup', 'paid', 'activities',
6     'nursery', 'higher', 'internet', 'romantic'])
7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



Gambar 2.348 Nomor 2

2.16.2.3 Nomor 3

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



Gambar 2.349 Nomor 3

2.16.2.4 Nomor 4

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

```

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

```

```

# In [20]:
tokyo = tokyo.sample(frac=0.5)
tokyo_train = tokyo[0:100]
tokyo_test = tokyo[100:150]
tokyo_train_att = tokyo_train.drop(['pass'], axis=1)
tokyo_train_pass = tokyo_train['pass']
tokyo_test_att = tokyo_test.drop(['pass'], axis=1)
tokyo_test_pass = tokyo_test['pass']

kyoto = tree.DecisionTreeClassifier(criterion='entropy',
max_depth=5)

kyoto = kyoto.fit(tokyo_train_att, tokyo_train_pass)

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

```

```

ModuleNotFoundError: No module named 'graphviz'

In [20]:
In [21]: tokyo = tokyo.sample(frac=0.5)
...: tokyo_train = tokyo[0:100]
...: tokyo_test = tokyo[100:150]
...: tokyo_train_att = tokyo_train.drop(['pass'], axis=1)
...: tokyo_train_pass = tokyo_train['pass']
...: tokyo_test_att = tokyo_test.drop(['pass'], axis=1)
...: tokyo_test_pass = tokyo_test['pass']
...:
...: kyoto = tree.DecisionTreeClassifier(criterion='entropy',
...: max_depth=5)
...:
...: kyoto = kyoto.fit(tokyo_train_att, tokyo_train_pass)
...:
...: print("Passing: %d out of %d (%.2f%%)" % (np.sum(tokyo_pass), len(tokyo_pass), 100*float(np.sum(tokyo_pass)) / len(tokyo_pass)))
...: Passing: 145 out of 150 (96.67%)

```

Gambar 2.350 Nomor 4

2.16.2.5 Nomor 5

```

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

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

```

Gambar 2.351 Nomor 5

2.16.2.6 Nomor 6

```

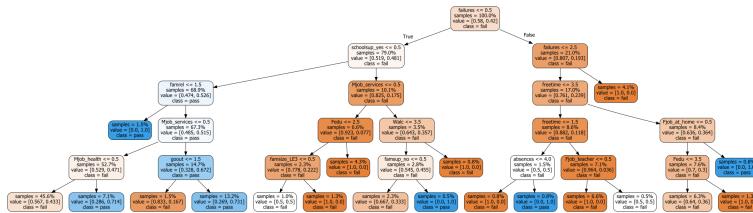
1 graph #Menampilkan graph menggunakan graphviz
2
3

```

```

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

```



Gambar 2.352 Nomor 6

2.16.2.7 Nomor 7

```

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

```

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

Gambar 2.353 Nomor 7

2.16.2.8 Nomor 8

```

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

```

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

Gambar 2.354 Nomor 8

2.16.2.9 Nomor 9

```

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

```

```

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

```

Gambar 2.355 Nomor 9

2.16.2.10 Nomor 10

```

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

```

```

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

```

Gambar 2.356 Nomor 10

2.16.2.11 Nomor 11

```

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

```

```

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

```

```

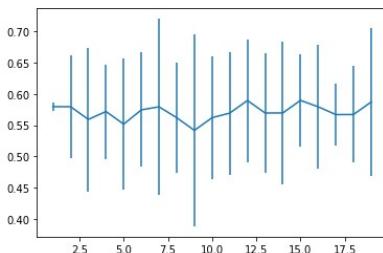
In [71]: depth_acc = np.empty((19,3), float)
...: i = 0
...: for max_depth in range(1, 20):
...:     kyoto = tree.DecisionTreeClassifier(criterion="entropy", max_depth=max_depth)
...:     nagoya = cross_val_score(kyoto, tokyo_att, tokyo_pass, cv=5)
...:     depth_acc[i,0] = max_depth
...:     depth_acc[i,1]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

```

Gambar 2.357 Nomor 11

2.16.2.12 Nomor 12

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



Gambar 2.358 Nomor 12

2.16.3 Penanganan Error

2.16.3.1 Error

1. ModuleNotFoundError

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

Gambar 2.359 ModuleNotFoundError

2.16.3.2 Solusi

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

BAB 3

CHAPTER 3

3.1 1174006 - Kadek Diva Krishna Murti

Lorem ipsum dolor sit amet, consectetur adipisicing elit.

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



Gambar 3.1 Kecerdasan Buatan.

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

3.1.1 Teori

3.1.2 Praktek

3.1.3 Penanganan Error

3.1.4 Bukti Tidak Plagiat



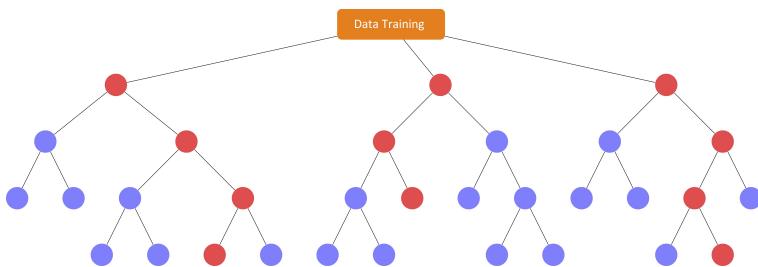
Gambar 3.2 Kecerdasan Buatan.

3.2 D.Irga B. Naufal Fakhri (1174066)

3.2.1 Teori

3.2.1.1 Apa Itu Random Forest

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

**Gambar 3.3** Random Forest

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

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

3.2.1.3 Cross Validation

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

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

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

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

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

Perhitungan Confusion Matriks dapat dilakukan dengan cara dibawah ini.

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

```
1 # -*- coding: utf-8 -*-
2 """
3 Created on Tue Mar 17 21:24:46 2020
4
5 @author: Rin
6 """
7 import numpy as np
8 import matplotlib.pyplot as plt
9 import pandas as pd
10 y_actu = pd.Series([2, 0, 2, 2, 0, 1, 1, 2, 2, 0, 1, 2], name
11                   ='Actual')
12 y_pred = pd.Series([0, 0, 2, 1, 0, 2, 1, 0, 2, 0, 2, 2], name
13                   ='Predicted')
14 df_confusion = pd.crosstab(y_actu, y_pred)
15 df_confusion = pd.crosstab(y_actu, y_pred, rownames=[ 'Actual' ,
16                             ], colnames=[ 'Predicted' ], margins=True)
17 def plot_confusion_matrix(df_confusion , title='Confusion
18                           matrix' , cmap=plt.cm.gray_r):
19     plt.matshow(df_confusion , cmap=cmap) # imshow
20     #plt.title(title)
21     plt.colorbar()
22     tick_marks = np.arange(len(df_confusion.columns))
```

```

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

```



Gambar 3.4 Confusion Matriks

3.2.1.6 Apa itu voting pada random forest

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

Untuk menggunakan Voting pada Random Forest dapat dilihat code ini:

```

1 # -*- coding: utf-8 -*-
2 """
3 Created on Tue Mar 17 21:32:47 2020
4
5 @author: Rin
6 """
7
8 import numpy as np
9 import matplotlib.pyplot as plt
10 from sklearn.linear_model import LogisticRegression
11 from sklearn.naive_bayes import GaussianNB
12 from sklearn.ensemble import RandomForestClassifier
13 from sklearn.ensemble import VotingClassifier
14 clf1 = LogisticRegression(solver='lbfgs', max_iter=1000,
   random_state=123)
15 clf2 = RandomForestClassifier(n_estimators=100, random_state=123)
16 clf3 = GaussianNB()
17 X = np.array([[-1.0, -1.0], [-1.2, -1.4], [-3.4, -2.2], [1.1,
   1.2]])
18 y = np.array([1, 1, 2, 2])
19 eclf = VotingClassifier(estimators=[('lr', clf1), ('rf', clf2),
   ('gnb', clf3)],
   voting='soft',
   weights=[1, 1, 5])
20 # predict class probabilities for all classifiers
21 probas = [c.fit(X, y).predict_proba(X) for c in (clf1, clf2, clf3,
   , eclf)]

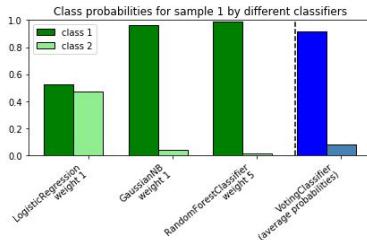
```

```

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

```

In [2]: runfile('N:/Tugas/Kuliah/Semester 6/Kecerdasan Bustan/KB3C_Ngerjain/src/1174066/3/untitled1.py', wdir='N:/Tugas/Kuliah/Semester 6/Kecerdasan Bustan/KB3C_Ngerjain/src/1174066/3')



Gambar 3.5 Voting Random Matriks

3.2.2 Praktek

3.2.2.1 Nomor 1

```

1 # In [0]
2 import pandas as dirga # Melakukan import library pandas menjadi
   nama sendiri yaitu dirga
3
4 aplikasi = {"Nama Aplikasi" : [ 'VSCode' , 'Atom' , 'Sublime' , 'Notepad
   ++' ]} # Membuat variabel yang bernama aplikasi , dan mengisi
   dataframanya dengan nama nama aplikasi koding
5 x = dirga.DataFrame(aplikasi) # Membuat variabel x yang akan
   membuat DataFrame dari library pandas yang akan memanggil
   variabel aplikasi.
6 print (' Dirga pake aplikasi: ' + x) #print hasil dari x

```

In [6]: import pandas as dirga # Melakukan import library pandas menjadi nama sendiri yaitu dirga

```

....:
....: aplikasi = {"Nama Aplikasi" : [
....: mengisi dataframanya dengan nama nama aplikasi
....: ....: x = dirga.DataFrame(aplikasi) # memanggil variabel aplikasi.
....: ....: print (' Dirga pake aplikasi: '
....:         Nama Aplikasi
0      Dirga pake aplikasi: VSCode
1      Dirga pake aplikasi: Atom
2      Dirga pake aplikasi: Sublime
3      Dirga pake aplikasi: Notepad++

```

Gambar 3.6 Membuat Aplikasi pakai pandas

3.2.2.2 Nomor 2

```

1 import numpy as dirga #Melakukan import library numpy menjadi
   nama sendiri yaitu dirga
2
3 matrix_x = dirga.eye(10) #Membuat sebuah matrix pake numpy dengan
   menggunakan fungsi eye
4 matrix_x #Mendeklarasikan matrix_x yang tadi dibuat
5
6 print (matrix_x) #print matrix_x yang tadi dibuat yang berbentuk
   10x10

```

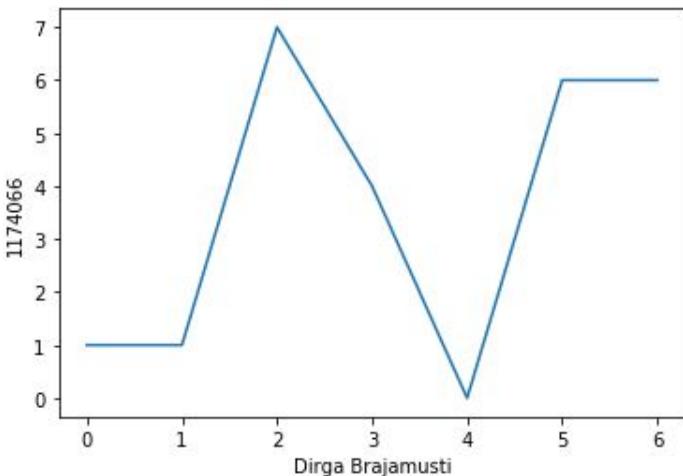
```
In [7]: import numpy as dirga #Melakukan
....:
....: matrix_x = dirga.eye(10) #Membuat
....: matrix_x #Menampilkan matriks
....:
....: print (matrix_x) #print matriks
[[1. 0. 0. 0. 0. 0. 0. 0. 0.]
 [0. 1. 0. 0. 0. 0. 0. 0. 0.]
 [0. 0. 1. 0. 0. 0. 0. 0. 0.]
 [0. 0. 0. 1. 0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 1. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0. 1. 0. 0. 0.]
 [0. 0. 0. 0. 0. 0. 1. 0. 0.]
 [0. 0. 0. 0. 0. 0. 0. 1. 0.]
 [0. 0. 0. 0. 0. 0. 0. 0. 1.]]
```

Gambar 3.7 Membuat Aplikasi pakai numpy

3.2.2.3 Nomor 3

```
1 import matplotlib.pyplot as dirga #Melakukan import library numpy
   menjadi nama sendiri yaitu dirga
2 dirga.plot([1,1,7,4,0,6,6]) #Memasukkan nilai pada plot
3 dirga.xlabel('Dirga Brajamusti') #Menambahkan label pada x
4 dirga.ylabel('1174066') #Menambahkan label pada y
5 dirga.show() #Menampilkan grafik plot
```

```
In [8]: import matplotlib.pyplot as dirga #Melakukan import
...: dirga.plot([1,1,7,4,0,6,6]) #Memasukkan nilai pada list
...: dirga.xlabel('Dirga Brajamusti') #Menambahkan Label pada x
...: dirga.ylabel('1174066') #Menambahkan label pada y
...: dirga.show() #Menampilkan grafik plot
```



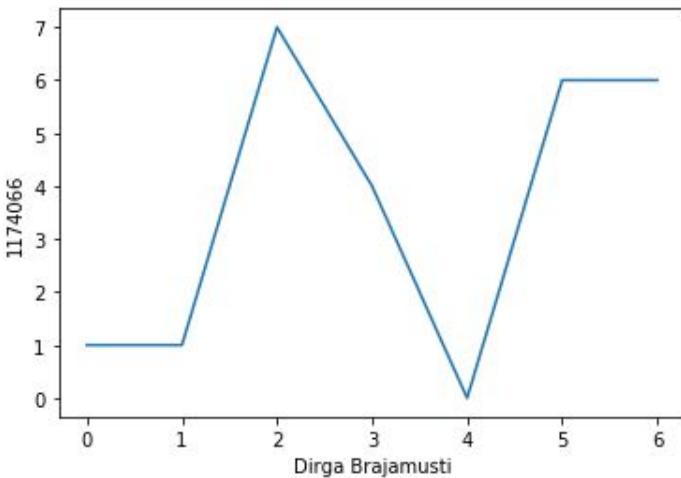
Gambar 3.8 Membuat Aplikasi pakai matplotlib

3.2.2.4 Nomor 4

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

```
1 import pandas as pd #Melakukan import library numpy menjadi
2           pd
3
3 imgatt = pd.read_csv("N:/Tugas/Kuliah/Semester 6/Kecerdasan
4          Buatan/KB3C Ngerjain/src/1174066/3/CUB_200_2011/attributes
5          /image_attribute_labels.txt",
6          sep='\s+', header=None, error_bad_lines=
7          False, warn_bad_lines=False,
8          usecols=[0,1,2], names=['imgid', 'attid',
9          , 'present']) #Membuat variable imgatt untuk membaca file
10         csv dari dataset, dengan ketentuan yang ada.
```

```
In [8]: import matplotlib.pyplot as dirga #Melakukan import
      ...: dirga.plot([1,1,7,4,0,6,6]) #Memasukkan nilai pada list
      ...: dirga.xlabel('Dirga Brajamusti') #Menambahkan Label pada x
      ...: dirga.ylabel('1174066') #Menambahkan label pada y
      ...: dirga.show() #Menampilkan grafik plot
```



Gambar 3.9 Hasil 4 Bagian 1

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

```
1 imgatt.head() #Menampilkan data yang sudah dibaca tadi tapi
   cuman data paling atas
```

```
In [12]: import pandas as pd #Melakukan import library numpy menjadi pd
      ...:
      ...: imgatt = pd.read_csv("CIB_200_2011/attributes/image_attribute_labels.txt",
      ...: sep="\t", header=None, error_bad_lines=False, warn_bad_lines=False,
      ...: usecols=[0,1,2], names=['imgid', 'attid', 'present'],
      file csv dari dataset, dengan ketentuan yang ada.
```

Gambar 3.10 Hasil 4 Bagian 2

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

```
1 imgatt.head() #Menampilkan data yang sudah dibaca tadi tapi
   cuman data paling atas
```

In [13]: `imgatt.head()` #Menampilkan 5 baris pertama

Out[13]:

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

Gambar 3.11 Hasil 4 Bagian 3

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

```
1 imgatt2 = imgatt.pivot(index='imgid', columns='attid', values
   ='present') #Membuat sebuah variabel baru dari fungsi
   imgatt, dengan mengganti index menjadi kolom dan kolom
   menjadi index
```

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

Gambar 3.12 Hasil 4 Bagian 4

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

```
1 imgatt2.head() #Menampilkan data yang sudah dibaca tadi tapi
   cuman data paling atas
```

In [16]: `imgatt2.head()` #Menampilkan data yang sudah dibaca tadi tapi cuman data paling atas

Out[16]:

	1	2	3	4	5	6	7	...	306	307	308	309	310	311	312
imgid	0	0	0	0	1	0	0	...	0	0	1	0	0	0	0
attid	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0
imgid	0	0	0	0	1	0	0	...	0	0	1	0	0	1	0
attid	0	0	0	0	1	0	0	...	1	0	0	1	0	0	0
imgid	0	0	0	0	1	0	0	...	0	0	0	0	0	0	0

[5 rows x 312 columns]

Gambar 3.13 Hasil 4 Bagian 5

- Menghasilkan jumlah kolom dan baris pada DataFrame imgatt2.

```
1 imgatt2.shape #Menampilkan jumlah seluruh data , kolom-nya
```

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

Gambar 3.14 Hasil 4 Bagian 6

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

```
1 imglabels = pd.read_csv("N:/Tugas/Kuliah/Semester 6/  
Kecerdasan Buatan/KB3C Ngerjain/src/1174066/3/CUB_200_2011/  
/image_class_labels.txt",  
2 sep=' ', header=None, names=['imgid',  
'label']) #Membaca data dimasukkan ke variable imglabels  
3  
4 imglabels = imglabels.set_index('imgid') #Variable imglabels  
dan set index (imgid)
```

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

Gambar 3.15 Hasil 4 Bagian 7

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

```
1 imglabels.head() #Menampilkan data yang sudah dibaca tadi  
tapi cuman data paling atas
```

```
In [22]: imglabels.head()
Out[22]:
      label
imgid
1          1
2          1
3          1
4          1
5          1
```

Gambar 3.16 Hasil 4 Bagian 8

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

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

```
In [23]: imglabels.shape
Out[23]: (11788, 1)
```

Gambar 3.17 Hasil 4 Bagian 9

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

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

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

Gambar 3.18 Hasil 4 Bagian 10

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

```
1 df_att = df.iloc[:, :312] #Membuat kolom dengan ketentuan 312
2 df_label = df.iloc[:, 312:] #Membuat kolom dengan ketentuan
3 312
```

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

Gambar 3.19 Hasil 4 Bagian 11

- Mengecek isi 5 data teratas pada df.att.

```
1 df_att.head() #Menampilkan data yang sudah dibaca tadi tapi
2 cuman data paling atas
```

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

[5 rows x 312 columns]

Gambar 3.20 Hasil 4 Bagian 12

- Mengecek isi data teratas dari df.label.

```
1 df_label.head() #Menampilkan data yang sudah dibaca tadi tapi
2 cuman data paling atas
```

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

Gambar 3.21 Hasil 4 Bagian 13

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

```
1 df_train_att = df_att[:8000] #Data akan dibagi dari 8000 row
   pertama menjadi data training dan sisanya adalah data
   testing
2 df_train_label = df_label[:8000] #Data akan dibagi dari 8000
   row pertama menjadi data training dan sisanya adalah data
   testing
3 df_test_att = df_att[8000:] #Berbalik dari sebelumnya data
   akan dibagi mulai dari 8000 row pertama menjadi data
   training dan sisanya adalah data testing
4 df_test_label = df_label[8000:] #Berbalik dari sebelumnya
   data akan dibagi mulai dari 8000 row pertama menjadi data
   training dan sisanya adalah data testing
5
6 df_train_label = df_train_label['label'] #Menambahkan label
7 df_test_label = df_test_label['label'] #Menambahkan label
```

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

Gambar 3.22 Hasil 4 Bagian 14

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

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

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

Gambar 3.23 Hasil 4 Bagian 15

- Menunjukkan hasil prediksi dari Random Forest.

```
1 clf.fit(df_train_att, df_train_label) #Variabnle clf untuk fit yaitu menjadi data training
```

```
In [31]: clf.fit(df_train_att, df_train_label) #Variabnle clf untuk fit yaitu menjadi data training
Out[31]:
RandomForestClassifier(bootstrap=True, class_weight=None, criterion='gini',
   max_depth=None, max_features=50, max_leaf_nodes=None,
   min_impurity_decrease=0.0, min_impurity_split=None,
   min_samples_leaf=1, min_samples_split=2,
   min_weight_fraction_leaf=0.0, n_estimators=100,
   n_jobs=None, oob_score=False, random_state=0, verbose=0,
   warm_start=False)
```

Gambar 3.24 Hasil 4 Bagian 16

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

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

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

Gambar 3.25 Hasil 4 Bagian 17

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

```
1 clf.score(df_test_att, df_test_label) #Memunculkan clf sebagai testing yang sudah di training tadi
```

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

Gambar 3.26 Hasil 4 Bagian 18

3.2.2.5 Nomor 5

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

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

Gambar 3.27 Hasil 5 Bagian 1

```
1 cm #Memunculkan data label berbentuk array
```

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

Gambar 3.28 Hasil 5 Bagian 2

```
1 import matplotlib.pyplot as plt #Mengimport library matplotlib sebagai plt
2 import itertools #Mengimport library itertools
3 def plot_confusion_matrix(cm, classes,
                           normalize=False,
```

```

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

```

```

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

```

Gambar 3.29 Hasil 5 Bagian 3

```

2           sep='\s+', header=None, usecols=[1], names=['
  birdname']) #membaca csv dengan ketentuan nama birdname
3 birds = birds['birdname'] #nama birds dengan ketentuan birdname
4 birds #Menampilkan data birds

```

```

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

```

Gambar 3.30 Hasil 5 Bagian 4

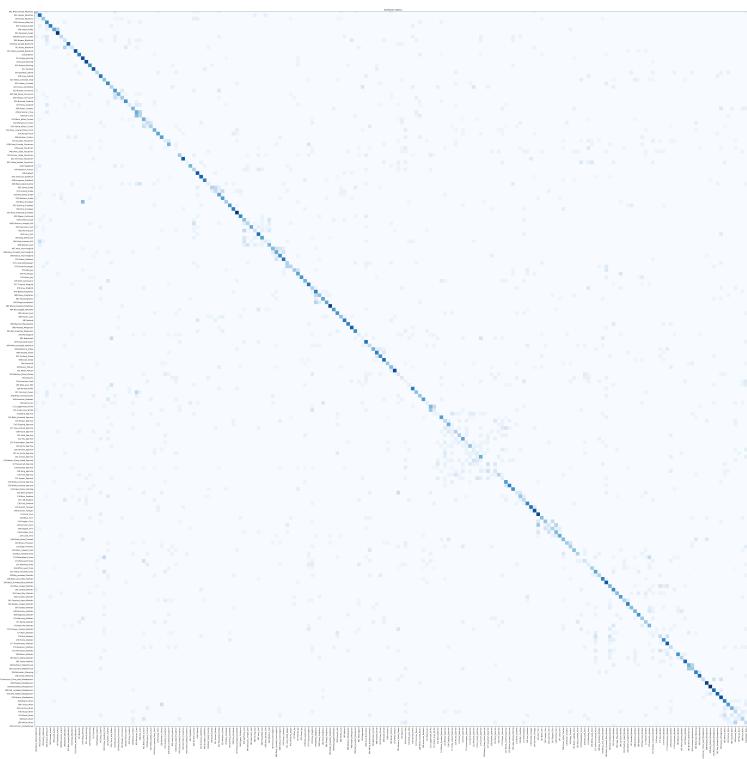
```

1 import numpy as np #Mengimport library numpy sebagai np
2 np.set_printoptions(precision=2) #np sebagai variabel yang
   membuat set precision=2
3 plt.figure(figsize=(60,60), dpi=300) #Plot sebagai figure dengan
   ketentuan sizw 60,60 dan dpi 300
4 plot_confusion_matrix(cm, classes=birds, normalize=True) #Data cm
   dan clas birds dibuat sebagai plot
5 #plt.show
6 #plt.savefig('hasil.png') #

```

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

Gambar 3.31 Hasil 5 Bagian 5



Gambar 3.32 Plot Hasil 5 Bagian 5

3.2.2.6 Nomor 6

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

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

Gambar 3.33 Hasil 6 Bagian 1

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

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

Gambar 3.34 Hasil 6 Bagian 2

3.2.2.7 Nomor 7

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

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

Gambar 3.35 Hasil 7 Bagian 1

```

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

```

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

Gambar 3.36 Hasil 7 Bagian 2

```

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

```

```
In [27]: scoressvm = cross_val_score(clfsvm, df_train_att, df_train_label, cv=5) #membuat variable data training
....: print("Accuracy: %.2f (+/- %.2f)" % (scoressvm.mean(), scoressvm.std() * 2)) #menampilkan data testing dan output
....:
C:\ProgramData\Anaconda3\lib\site-packages\sklearn\svm\base.py:193: FutureWarning: The default value of gamma will change
from 'auto' to 'scale' in version 0.22 to account better for unscaled features. Set gamma explicitly to 'auto' or 'scale' to
avoid this warning.
    "avoid this warning.", FutureWarning)
C:\ProgramData\Anaconda3\lib\site-packages\sklearn\svm\base.py:193: FutureWarning: The default value of gamma will change
from 'auto' to 'scale' in version 0.22 to account better for unscaled features. Set gamma explicitly to 'auto' or 'scale' to
avoid this warning.
    "avoid this warning.", FutureWarning)
C:\ProgramData\Anaconda3\lib\site-packages\sklearn\svm\base.py:193: FutureWarning: The default value of gamma will change
from 'auto' to 'scale' in version 0.22 to account better for unscaled features. Set gamma explicitly to 'auto' or 'scale' to
avoid this warning.
    "avoid this warning.", FutureWarning)
C:\ProgramData\Anaconda3\lib\site-packages\sklearn\svm\base.py:193: FutureWarning: The default value of gamma will change
from 'auto' to 'scale' in version 0.22 to account better for unscaled features. Set gamma explicitly to 'auto' or 'scale' to
avoid this warning.
    "avoid this warning.", FutureWarning)
C:\ProgramData\Anaconda3\lib\site-packages\sklearn\svm\base.py:193: FutureWarning: The default value of gamma will change
from 'auto' to 'scale' in version 0.22 to account better for unscaled features. Set gamma explicitly to 'auto' or 'scale' to
avoid this warning.
    "avoid this warning.", FutureWarning)
Accuracy: 0.27 (+/- 0.03)
```

Gambar 3.37 Hasil 7 Bagian 3

3.2.2.8 Nomor 8

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

```

....      ..._param[4] = scores.mean() + scores.std() * 2
....      rf_params[i,3] = scores.std() * 2 #index 3
....      i += 1 #Dengan ketentuan i += 1
....      print("Max features: %d, num estimators: %d, acc
n_estimators, scores.mean(), scores.std() * 2))
....      #Print hasil pengulangan yang sudah ditentukan
Max features: 5, num estimators: 10, accuracy: 0.26 (+/- 0.03)
Max features: 5, num estimators: 30, accuracy: 0.36 (+/- 0.03)
Max features: 5, num estimators: 50, accuracy: 0.39 (+/- 0.03)
Max features: 5, num estimators: 70, accuracy: 0.41 (+/- 0.02)
Max features: 5, num estimators: 90, accuracy: 0.42 (+/- 0.02)
Max features: 5, num estimators: 110, accuracy: 0.43 (+/- 0.03)
Max features: 5, num estimators: 130, accuracy: 0.43 (+/- 0.02)
Max features: 5, num estimators: 150, accuracy: 0.44 (+/- 0.02)
Max features: 5, num estimators: 170, accuracy: 0.44 (+/- 0.03)
Max features: 5, num estimators: 190, accuracy: 0.45 (+/- 0.01)
Max features: 10, num estimators: 10, accuracy: 0.29 (+/- 0.02)
Max features: 10, num estimators: 30, accuracy: 0.38 (+/- 0.02)
Max features: 10, num estimators: 50, accuracy: 0.41 (+/- 0.03)
Max features: 10, num estimators: 70, accuracy: 0.43 (+/- 0.02)
Max features: 10, num estimators: 90, accuracy: 0.43 (+/- 0.02)
Max features: 10, num estimators: 110, accuracy: 0.44 (+/- 0.02)
Max features: 10, num estimators: 130, accuracy: 0.45 (+/- 0.03)
Max features: 10, num estimators: 150, accuracy: 0.45 (+/- 0.03)
Max features: 10, num estimators: 170, accuracy: 0.45 (+/- 0.02)
Max features: 10, num estimators: 190, accuracy: 0.45 (+/- 0.03)
Max features: 15, num estimators: 10, accuracy: 0.31 (+/- 0.03)
Max features: 15, num estimators: 30, accuracy: 0.40 (+/- 0.03)
Max features: 15, num estimators: 50, accuracy: 0.42 (+/- 0.03)
Max features: 15, num estimators: 70, accuracy: 0.43 (+/- 0.02)
Max features: 15, num estimators: 90, accuracy: 0.43 (+/- 0.03)
Max features: 15, num estimators: 110, accuracy: 0.44 (+/- 0.02)
Max features: 15, num estimators: 130, accuracy: 0.44 (+/- 0.02)
Max features: 15, num estimators: 150, accuracy: 0.45 (+/- 0.02)
Max features: 15, num estimators: 170, accuracy: 0.45 (+/- 0.02)
Max features: 15, num estimators: 190, accuracy: 0.45 (+/- 0.02)

```

Gambar 3.38 Hasil 8 Bagian 1

```

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

```

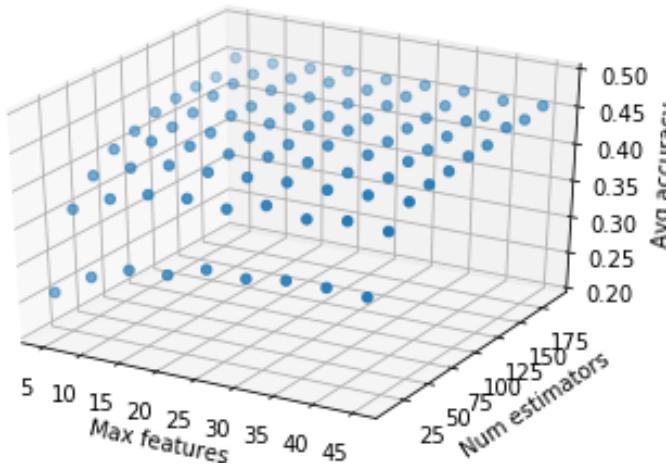
Gambar 3.39 Hasil 8 Bagian 1 Akhir kode

```

1 import matplotlib.pyplot as plt #Mengimport library matplotlib
sebagai plt

```

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



Gambar 3.40 Hasil 8 Bagian 2

3.2.3 Penanganan Error

1. ScreenShoot Error

`FileNotFoundException: [line 2] File 'b\Tugas\Latin\Semester 6\Kependidikan Butiran\KBCI Ngajar\jain\src\11274066\7\CBR_200_2011_image_class_labels.txt' does not exist.' b\Tugas\Latin\Semester 6\Kependidikan Butiran\KBCI Ngajar\src\11274066\7\`

Gambar 3.41 FileNotFoundError

2. Tuliskan Kode Error dan Jenis Error

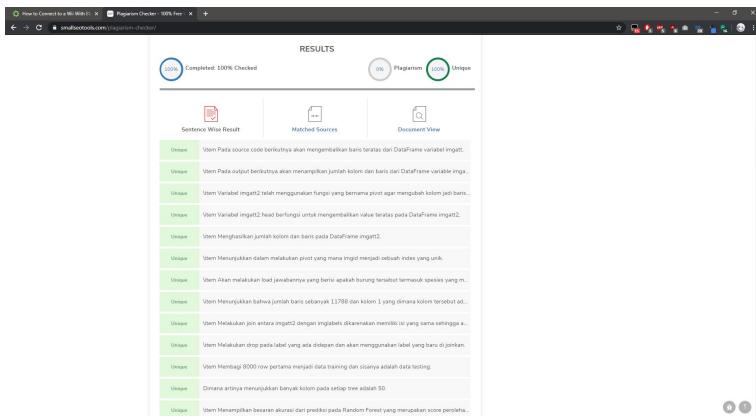
- FileNotFoundException

3. Cara Penangan Error

- **FileNotFoundException**

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

3.2.4 Bukti Tidak Plagiat



Gambar 3.42 Bukti Tidak Plagiat

3.2.5 Link Youtube:

https://youtu.be/69BWGlcf_CQ

3.3 1174083 - Bakti Qilan Mufid

Chapter 3 - Prediksi dengan Random Forest

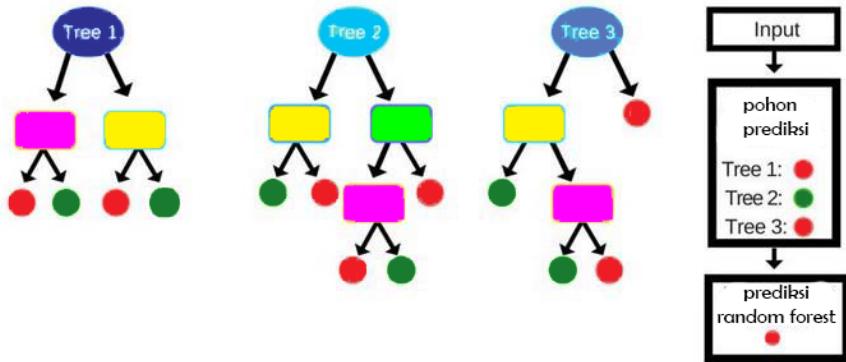
3.3.1 Teori

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

Random Forest adalah konstruksi data yang diterapkan pada machine learning yang mengembangkan sejumlah besar pohon keputusan acak yang menganalisis sekumpulan variabel. Jenis algoritma ini membantu meningkatkan cara teknologi menganalisis data yang kompleks. Juga merupakan algoritma machine learning yang fleksibel, mudah digunakan, bahkan tanpa penyetelan hyperparameter, dengan hasil yang baik. Ini juga merupakan salah satu al-

goritma yang paling banyak digunakan, karena kesederhanaan dan faktanya dapat digunakan untuk tugas klasifikasi dan regresi.

algoritma random forest



Gambar 3.43 contoh random forest

3.3.1.2 Jelaskan cara membaca dataset kasus dan artikan makna setiap file dan isi field masing masing file.

Dataset adalah kumpulan data. Paling umum satu data set sesuai dengan isi tabel database tunggal, atau matriks data statistik tunggal, dimana setiap kolom tabel mewakili variabel tertentu, dan setiap baris sesuai dengan anggota tertentu dari dataset yang dipertanyakan.

1. cara membaca dataset

- Gunakan librari Pandas pada python untuk dapat membaca dataset dengan format text file.
- Setelah itu, buat variabel baru "dataset" yang berisikan perintah untuk membaca file csv. seperti berikut

```

1 import pandas as pd
2
3 dataset = pd.read_csv('E:/backup/sem 6/Kecerdasan Buatan/
   KB3C - Copy/src/1174083/src3/case.csv', sep=',')
4 dataset.head()

```

Pada kodingan diatas dapat dijelaskan bahwa : Memanggil Librari Panda untuk membaca dataset Membuat variabel "Dataset" yang berisikan pd.read_csv untuk membaca dataset.

- setelah di run maka hasilnya seperti berikut

Index	case_id	province	city	group	infection_case	confirmed	latitude	longitude
0	1	Seoul	Guro-gu	True	Guro-gu Call Center	70	37.508163	126.884387
1	2	Seoul	Eunpyeong-gu	True	Eunpyeong St. Mary's Hospital	14	37.63369	126.9165
2	3	Seoul	Seongdong-gu	True	Seongdong-gu APT	13	37.55713	127.0403
3	4	Seoul	Jongno-gu	True	Jongno Community Center	10	37.57681	127.0065
4	5	Seoul	Dongdaemun-gu	True	Donghan Church	9	37.592888	127.056766
5	6	Seoul	Jung-gu	True	Jung-gu Fashion Company	7	37.562405	126.984377
6	7	Seoul	from other city	True	Shincheonji Church	6	-	-
7	8	Seoul	-	False	etc	46	-	-
8	9	Busan	Dongnae-gu	True	Onchun Church	34	35.21628	129.0771
9	10	Busan	from other city	True	Shincheonji Church	18	-	-
10	11	Busan	Suyeong-gu	True	Suyeong-gu Kindergarten	6	35.16708	129.1124
11	12	Busan	Haeundae-gu	True	Haeundae-gu Catholic Church	6	35.20599	129.1256
12	13	Busan	Jin-gu	True	Jin-gu Academy	4	35.17371	129.0633
13	14	Busan	from other city	True	Cheongdo Daemam Hospital	1	-	-
14	15	Busan	-	False	etc	29	-	-
15	16	Daegu	Nam-gu	True	Shincheonji Church	4126	35.84008	128.5667
16	17	Daegu	-	False	contact with patient	688	-	-
17	18	Daegu	from other city	True	Cheongdo Daemam Hospital	2	-	-
18	19	Daegu	-	False	etc	1059	-	-
19	20	Incheon	from other city	True	Guro-gu Call Center	15	-	-
20	21	Incheon	-	False	contact with patient	4	-	-

Gambar 3.44 contoh dataset

Pertama tama gambar diatas merupakan dataset yang digunakan untuk mengetahui penyebaran virus corona di negara korea selatan. Datasetnya dapat didapatkan dari laman <https://www.kaggle.com/kimjihoo/covid19-southkorea>. Penjelasan dari isi field diatas adalah sebagai berikut :

- Atribut Index merupakan atribut otomatis untuk penomoran data yang ada.
- Atribut case_id merupakan nomor id dari case
- Atribut province merupakan nama provinsi
- Atribut city merupakan nama kota
- Atribut group terdiri dari dua variable, yaitu TRUE (untuk terindikasi infeksi dari group) dan FALSE(sebaliknya)
- Atribut infection_case merupakan nama kasus yang menginfeksi
- Atribut confirmed merupakan kasus yang sudah terkonfirmasi
- Atribut latitude merupakan titik koordinat
- Atribut longitude merupakan titik koordinat

3.3.1.3 Jelaskan apa itu cross validation

Cross-validation (CV) adalah metode statistik yang dapat digunakan untuk mengevaluasi kinerja model atau algoritma dimana data dipisahkan menjadi dua subset yaitu data proses pembelajaran dan data validasi / evaluasi. Model

atau algoritma dilatih oleh subset pembelajaran dan divalidasi oleh subset validasi. Selanjutnya pemilihan jenis CV dapat didasarkan pada ukuran dataset. Biasanya CV K-fold digunakan karena dapat mengurangi waktu komputasi dengan tetap menjaga keakuratan estimasi. Teknik ini utamanya digunakan untuk melakukan prediksi model dan memperkirakan seberapa akurat sebuah model prediktif ketika dijalankan dalam praktiknya. Dalam sebuah masalah prediksi, sebuah model biasanya diberikan kumpulan data (dataset) yang diketahui untuk digunakan dalam menjalankan pelatihan (dataset pelatihan), serta kumpulan data yang tidak diketahui (atau data yang pertama kali dilihat) terhadap model yang diuji (pengujian dataset) Metode 3 - fold cross validation membagi sebuah himpunan contoh secara acak menjadi 3 subset yang saling bebas. Dilakukan pengulangan sebanyak 3kali untuk pelatihan dan pengujian. Pada setiap ulangan, disisakan satu subset untuk pengujian dan subset lainnya untuk pelatihan. Tingkat akurasi dihitung dengan membagi jumlah keseluruhan klasifikasi yang benar dengan jumlah semua instance pada data awal.

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

Itu merupakan persentase keakurasi prediksi yang dilakukan pada saat testing menggunakan label pada dataset yang digunakan. Score merupakan mendefinisikan aturan evaluasi model. Maka pada saat dijalankan akan muncul persentase tersebut yang menunjukkan keakurasi atau keberhasilan dari prediksi yang dilakukan. Jika menggunakan Random Forest maka hasilnya 40% , jika menggunakan Decision Tree hasil prediksinya yaitu 27% dan pada SVM 29%

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

Perhitungan Confusion Matriks dapat dilakukan sebagai berikut. Disini saya menggunakan data yang dibuat sendiri untuk menampilkan data aktual dan prediksi.

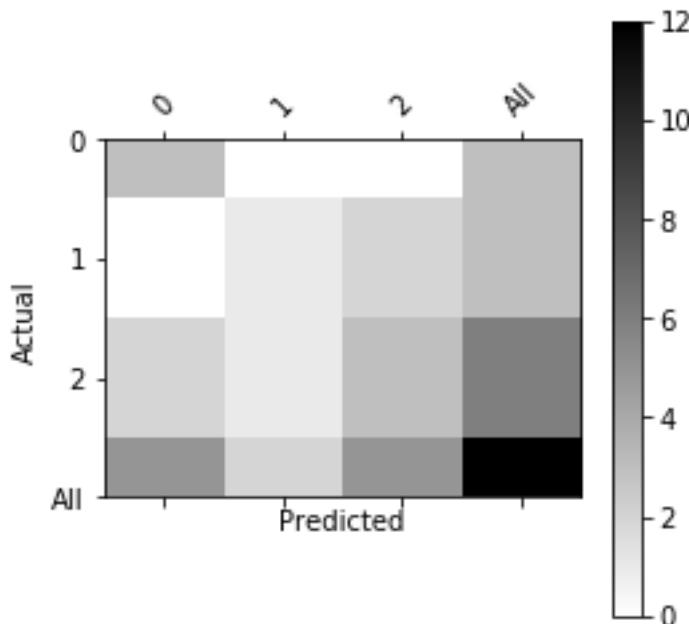
- Import librari Pandas, Matplotlib, dan Numpy.
- Buat variabel y_actu yang berisikan data aktual.
- Buat variabel y_pred berisikan data yang akan dijadikan sebagai prediksi.
- Buat variabel df_confusion yang berisikan crosstab untuk membangun tabel tabulasi silang yang dapat menunjukkan frekuensi kemunculan kelompok data tertentu.
- Pada variabel df_confusion definisikan lagi nama baris yaitu Actual dan kolomnya Predicted
- Kemudian definisikan suatu fungsi yang diberi nama plot_confusion_matrix yang berisikan pendefinisian confusion matrix dan juga akan di plotting. untuk code lengkapnya sebagai berikut:

```

1 import pandas as pd
2 import numpy as np
3 import matplotlib.pyplot as plt
4
5 y_actu = pd.Series([2, 0, 2, 2, 0, 1, 1, 2, 2, 0, 1, 2], name
6                   ='Actual')
7 y_pred = pd.Series([0, 0, 2, 1, 0, 2, 1, 0, 2, 0, 2, 2], name
8                   ='Predicted')
9 df_confusion = pd.crosstab(y_actu, y_pred, rownames=['Actual',
10                           ], colnames=['Predicted'], margins=True)
11
12 def plot_confusion_matrix(df_confusion, title='Confusion
13                           matrix', cmap=plt.cm.gray_r):
14     plt.matshow(df_confusion, cmap=cmap) # imshow
15     #plt.title(title)
16     plt.colorbar()
17     tick_marks = np.arange(len(df_confusion.columns))
18     plt.xticks(tick_marks, df_confusion.columns, rotation=45)
19     plt.yticks(tick_marks, df_confusion.index)
20     plt.ylabel(df_confusion.index.name)
21     plt.xlabel(df_confusion.columns.name)
22
23 plot_confusion_matrix(df_confusion)

```

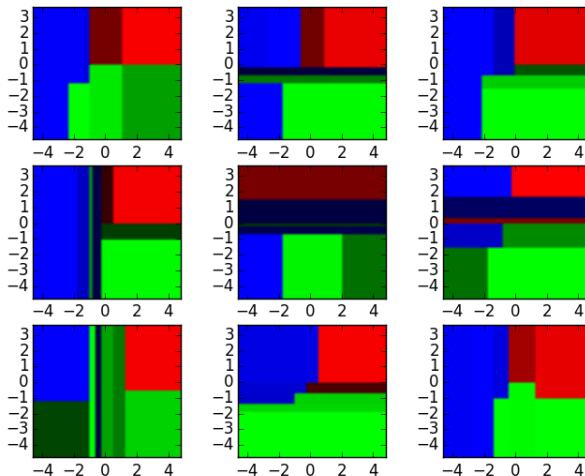
Hasil dari kode diatas akan menghasilkan confusion matrix seperti berikut:



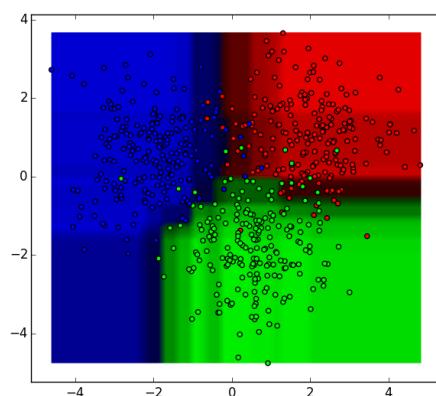
Gambar 3.45 hasil confusion matrix

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

Voting yaitu suara untuk setiap target yang diprediksi pada saat melakukan Random Forest. Pertimbangkan target prediksi dengan voting tertinggi sebagai prediksi akhir dari algoritma random forest. contohnya pada gambar berikut, dimana ada 9 hasil yang berbeda dari decision tree, dan menghasilkan suatu random forest



Gambar 3.46 contoh 9 hasil decision tree yang berbeda



Gambar 3.47 contoh random forest

3.3.2 Praktek

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

Disini saya akan membuat program sederhana menggunakan Pandas yaitu untuk memilih baris dari DataFrame yang diberikan berdasarkan value disalah satu kolom.

```

1 import pandas as pd
2
3 d = { 'col1':[1 ,2 ,3 ,4 ,5] , 'col2' :[3 ,5 ,5 ,4 ,2] , 'col3' :[2 ,3 ,1 ,1 ,4] }
4 df = pd.DataFrame(data=d)
5 print("nomor 1")
6 print("Hasil: ")
7 print(df)

```

Dari code diatas dapat dijelaskan perbarisnya sebagai berikut :

- Baris pertama, yaitu import pandas yang artinya kita akan mengimport librari Pandas dari python dengan inisiasi pd.
- Variabel d didefinisikan sebagai data. data untuk col1, col2, dan col3.
- Variabel df akan mengubah data pada variabel d disejajarkan menjadi baris dan kolom dengan menggunakan fungsi pd.DataFrame(data=d).
- Baris selanjutnya yaitu akan mencetak atau menampilkan tulisan "nomor 1" dan juga "Hasil: " pada jendela console
- print(df) artinya akan mencetak atau menampilkan DataFrame dari data yang telah dibuat tadi.

Hasilnya seperti berikut:

```

nomor 1
Hasil:
    col1  col2  col3
0      1      3      2
1      2      5      3
2      3      5      1
3      4      4      1
4      5      2      4

```

Gambar 3.48 contoh aplikasi sederhana menggunakan pandas

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

Disini saya akan membuat program sederhana menggunakan numpy, dimana saya membuat array dari angka 1 sebanyak 26 data, artinya dari angka 1-25. dan mengubah array tersebut menjadi matrix 5x5.

```

1 import numpy as np
2
3 slur = np.arange(1,26).reshape(5,5)
4 print("nomor 2")
5 print(slur)

```

Dari code diatas dapat dijelaskan perbarisnya sebagai berikut :

- baris pertama kita meng-import numpy dan menamainya dengan np
- lalu kita membuat variabel slur dengan diisi arraya yang dimulai dari angka 1 sebanyak 26 data(sampai angka 25) lalu mengubahnya kedalam bentuk matrix 5x5
- Baris selanjutnya yaitu akan mencetak atau menampilkan tulisan "nomor 2" pada jendela console
- print(slur) artinya akan mencetak atau menampilkan data yang ada pada variable slur

Hasilnya seperti berikut:

```

nomor 2
[[ 1  2  3  4  5]
 [ 6  7  8  9 10]
 [11 12 13 14 15]
 [16 17 18 19 20]
 [21 22 23 24 25]]

```

Gambar 3.49 contoh aplikasi sederhana menggunakan numpy

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

Disini saya akan membuat program sederhana menggunakan matplotlib.

```

1 import matplotlib.pyplot as plt
2
3 t = np.arange(0., 5., 0.2)
4 print("nomor 3")
5 plt.plot(t, t, 'r--', t, t**2, 'bo', t, t**3, 'g^')
6 plt.show()

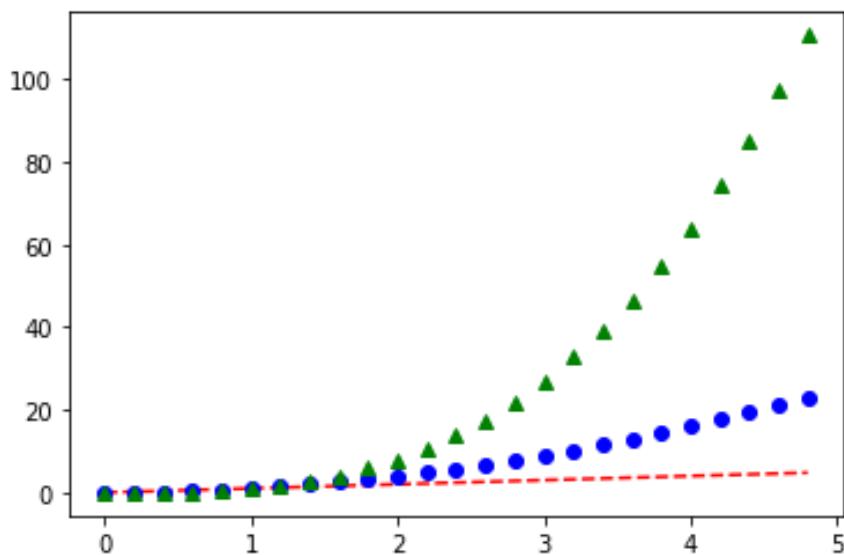
```

Dari code diatas dapat dijelaskan perbarisnya sebagai berikut :

- baris pertama kita meng-import matplotlib.pyplot dan menamainya dengan plt
- membuat variable t dan mengisi datanya dengan array(menggunakan library numpy) yang dimulai dari angka 0 sebanyak 5 array dengan interval 0.2 untuk setiap arraynya.
- Baris selanjutnya yaitu akan mencetak atau menampilkan tulisan "nomor 3" pada jendela console
- plt.plot akan membuat grafik dengan data yang ada pada variable t. 'r-' itu untuk membuat garis putus-putus berwarna merah, t^{**2} itu sama dengan t pangkat 2, 'bo' sama dengan black oval, 'g^' sama dengan segitiga berwarna hijau.
- plt.show() digunakan untuk menampilkan grafik pada saat skrip dijalankan.

Hasilnya seperti berikut:

nomor 3



Gambar 3.50 contoh aplikasi sederhana menggunakan matplotlib

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

yang didapatkan.

Pertama dataset kita baca terlebih dahulu.

```
1 # In [4]: Random Forest
2 import pandas as pd #import library pandas sebagai pd
3
4 imgatt = pd.read_csv("E:/backup/sem 6/Kecerdasan Buatan/
5 CUB_200_2011/attributes/image_attribute_labels.txt",
6 sep='\s+', header=None, error_bad_lines=
7 False, warn_bad_lines=False,
8 usecols=[0,1,2], names=['imgid', 'attid', 'present'])#untuk membaca file txt
```

Listing 3.1 Membaca data file txt

Hasilnya seperti berikut:

```
Name      Type      Size           Value
imgatt  DataFrame  (3677856, 3)  Column names: imgid, attid, present
```

Variable explorer File explorer Help

IPython console

Console 1/A

```
In [1]: import pandas as pd #import library pandas sebagai pd
...
...: imgatt = pd.read_csv("E:/backup/sem 6/Kecerdasan Buatan/CUB_200_2011/attributes/
image_attribute_labels.txt",
...: sep='\s+', header=None, error_bad_lines=False,
warn_bad_lines=False,
...: usecols=[0,1,2], names=['imgid', 'attid', 'present'])#untuk
membaca file txt
```

```
In [2]:
```

Gambar 3.51 Hasil dari listing 3.1

Melihat sebagian data awal, dengan menggunakan listing 3.2

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

Listing 3.2 Melihat sebagian data(bagian awal)

Hasilnya seperti berikut:

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

Gambar 3.52 Hasil dari listing 3.2

Melihat jumlah data menggunakan listing 3.3

```
1 imgatt.shape
```

Listing 3.3 Mengetahui jumlah data

Hasilnya seperti berikut:

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

Gambar 3.53 Hasil dari listing 3.3

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

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

Listing 3.4 Pivot dataset

Hasilnya seperti berikut:

	Column names: 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 1 ...														
In [5]:	imgatt2.head()														
Out[5]:															
attid	1	2	3	4	5	6	7	...	306	307	308	309	310	311	312
imgid								...							
1	0	0	0	0	1	0	0	...	0	0	1	0	0	0	0
2	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0
3	0	0	0	0	1	0	0	...	0	0	1	0	0	1	0
4	0	0	0	0	1	0	0	...	1	0	0	1	0	0	0
5	0	0	0	0	1	0	0	...	0	0	0	0	0	0	0

[5 rows x 312 columns]

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

Gambar 3.54 Hasil dari listing 3.4

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

```
1 imglabels = pd.read_csv("E:/backup/sem 6/Kecerdasan Buatan/
CUB_200_2011/image_class_labels.txt",
2                                     sep=' ', header=None, names=['imgid' , 'label'])
3
4 imglabels = imglabels.set_index('imgid')
5 imglabels .head()
6 imglabels .shape
```

Listing 3.5 Membaca dataset label file txt

Hasilnya seperti berikut:

imglabels	DataFrame	(11788, 1)	Column names: label
In [8]: imglabels.head()			
Out[8]:			
		label	
imgid			
1		1	
2		1	
3		1	
4		1	In [9]: imglabels.shape
5		1	Out[9]: (11788, 1)

Gambar 3.55 Hasil dari listing 3.5

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

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

Listing 3.6 Menggabungkan field dari dua file terpisah

Hasilnya seperti berikut:

Name	Type	Size	Value
df	DataFrame	(11788, 313)	Column names: 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 1 ...

Gambar 3.56 Hasil dari listing 3.6

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

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

Listing 3.7 Memisahkan dan memilih label

Hasilnya seperti berikut:

Name	Type	Size	Value
df_att	DataFrame	(11788, 312)	Column names: 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 1 ...
df_label	DataFrame	(11788, 1)	Column names: label

Gambar 3.57 Hasil dari listing 3.7

Kita bisa mengecek isinya dengan perintah listing 3.8

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

Listing 3.8 Melihat isi masing-masing dataframe

Hasilnya seperti berikut:

```
In [12]: df_att.head()
Out[12]:
   1    2    3    4    5    6    7    ...   306   307   308   309   310   311   312
imgid
3652    0    0    0    0    0    0    0    ...    1    0    0    1    0    0    0
647     0    0    0    0    0    0    1    ...    0    0    0    1    0    0    0
11192   0    0    0    0    0    0    0    ...    0    1    0    0    0    0    0
8033    0    0    0    0    0    0    1    ...    0    0    0    1    0    0    0
7286    0    0    0    0    0    0    0    ...    0    0    0    0    0    1    0

[5 rows x 312 columns]

In [13]: df_label.head()
Out[13]:
      label
imgid
3652      63
647       12
11192     191
8033      137
7286      125
```

Gambar 3.58 Hasil dari listing 3.8

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

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

Listing 3.9 Pembagian data training dan test

Hasilnya seperti berikut:

Name	Type	Size	Value
df_test_att	DataFrame	(3788, 312)	Column names: 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 1 ...
df_test_label	Series	(3788,)	Series object of pandas.core.series module
df_train_att	DataFrame	(8000, 312)	Column names: 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 1 ...
df_train_label	Series	(8000,)	Series object of pandas.core.series module

Gambar 3.59 Hasil dari listing 3.9

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

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

Listing 3.10 Pembagian data training dan testInstansiasi kelas Random Forest

Hasilnya seperti berikut:

```
In [15]: from sklearn.ensemble import RandomForestClassifier
...: clf = RandomForestClassifier(max_features=50, random_state=0, n_estimators=100)
```

Gambar 3.60 Hasil dari listing 3.10

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

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

Listing 3.11 Fitting random forest dengan dataset training

Hasilnya seperti berikut:

```
In [16]: clf.fit(df_train_att, df_train_label)
Out[16]:
RandomForestClassifier(bootstrap=True, class_weight=None, criterion='gini',
   max_depth=None, max_features=50, max_leaf_nodes=None,
   min_impurity_decrease=0.0, min_impurity_split=None,
   min_samples_leaf=1, min_samples_split=2,
   min_weight_fraction_leaf=0.0, n_estimators=100,
   n_jobs=None, oob_score=False, random_state=0, verbose=0,
   warm_start=False)
```

Gambar 3.61 Hasil dari listing 3.11

Hasilnya bisa kita dapatkan dengan perintah predict dengan perintah listing 3.12

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

Listing 3.12 Melihat Hasil prediksi

Hasilnya seperti berikut:

```
In [17]: print(clf.predict(df_train_att.head()))
[ 63  12 191 137 125]
```

Gambar 3.62 Hasil dari listing 3.12

Untuk besaran akurasinya dengan perintah listing 3.13

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

Listing 3.13 Score perolehan dari klasifikasi

Hasilnya seperti berikut:

```
In [18]: clf.score(df_test_att, df_test_label)
Out[18]: 0.4263463569165787
```

Gambar 3.63 Hasil dari listing 3.13

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

Dari Random Forest kita coba petakan ke dalam Confusion Matrix dan lihat hasilnya dengan perintah listing 3.14

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

Listing 3.14 Membuat Confusion Matrix

Hasilnya seperti berikut:

Name	Type	Size	Value
cm	int64	(200, 200)	[[5 0 0 ... 0 0 0] [0 11 0 ... 0 0 0]
pred_labels	int64	(3788,)	[44 19 11 ... 45 47 70]

```
In [20]: cm
Out[20]:
array([[ 5,  0,  0, ...,  0,  0,  0],
       [ 0, 11,  0, ...,  0,  0,  0],
       [ 2,  0,  6, ...,  0,  0,  0],
       ...,
       [ 0,  0,  0, ...,  4,  0,  0],
       [ 0,  0,  0, ...,  0,  8,  0],
       [ 0,  0,  0, ...,  0,  0, 13]], dtype=int64)
```

Gambar 3.64 Hasil dari listing 3.14

Kemudian kita plot dengan perintah

```
1 import matplotlib.pyplot as plt
2 import itertools
3 def plot_confusion_matrix(cm, classes,
4                           normalize=False,
5                           title='Confusion matrix',
6                           cmap=plt.cm.Blues):
7     """
8     This function prints and plots the confusion matrix.
9     Normalization can be applied by setting `normalize=True`.
10    """
11    if normalize:
12        cm = cm.astype('float') / cm.sum(axis=1)[:, np.newaxis]
13        print("Normalized confusion matrix")
14    else:
15        print('Confusion matrix, without normalization')
16
17    print(cm)
18
19    plt.imshow(cm, interpolation='nearest', cmap=cmap)
20    plt.title(title)
21    #plt.colorbar()
22    tick_marks = np.arange(len(classes))
23    plt.xticks(tick_marks, classes, rotation=90)
24    plt.yticks(tick_marks, classes)
25
26    fmt = '.2f' if normalize else 'd'
27    thresh = cm.max() / 2.
28
29    plt.tight_layout()
30    plt.ylabel('True label')
31    plt.xlabel('Predicted label')
```

Listing 3.15 Plotting Confusion Matrix

Hasilnya seperti berikut:

```
In [21]: import matplotlib.pyplot as plt
...: import itertools
...: def plot_confusion_matrix(cm, classes,
...:                         normalize=False,
...:                         title='Confusion matrix',
...:                         cmap=plt.cm.Blues):
...:     """
...:     This function prints and plots the confusion matrix.
...:     Normalization can be applied by setting `normalize=True`.
...:     """
...:     if normalize:
...:         cm = cm.astype('float') / cm.sum(axis=1)[:, np.newaxis]
...:         print("Normalized confusion matrix")
...:     else:
...:         print('Confusion matrix, without normalization')
...:
...:     print(cm)
...:
...:     plt.imshow(cm, interpolation='nearest', cmap=cmap)
...:     plt.title(title)
...:     #plt.colorbar()
...:     tick_marks = np.arange(len(classes))
...:     plt.xticks(tick_marks, classes, rotation=90)
...:     plt.yticks(tick_marks, classes)
...:
...:     fmt = '.2f' if normalize else 'd'
...:     thresh = cm.max() / 2.
...:
...:     plt.tight_layout()
...:     plt.ylabel('True label')
...:     plt.xlabel('Predicted label')
```

Gambar 3.65 Hasil dari listing 3.15

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

```
1 birds = pd.read_csv("E:/backup/sem 6/Kecerdasan Buatan/  
2 CUB_200_2011/classes.txt",  
3 sep='\s+', header=None, usecols=[1], names=[  
4 'birdname'])  
birds = birds['birdname']  
birds
```

Listing 3.16 Membaca file classes.txt

Hasilnya seperti berikut:

```
In [22]: birds = pd.read_csv("E:/backup/sem 6/Kecerdasan Buatan/CUB_200_2011/classes.txt",
...:                         sep='\s+', header=None, usecols=[1], names=['birdname'])
...: birds = birds['birdname']
...: birds
Out[22]:
0      001.Black_footed_Albatross
1      002.Laysan_Albatross
2      003.Sooty_Albatross
3      004.Groove_billed_Ani
4      005.Crested_Auklet
...
195      196.House_Wren
196      197.Marsh_Wren
197      198.Rock_Wren
198      199.Winter_Wren
199      200.Common_Yellowthroat
Name: birdname, Length: 200, dtype: object
```

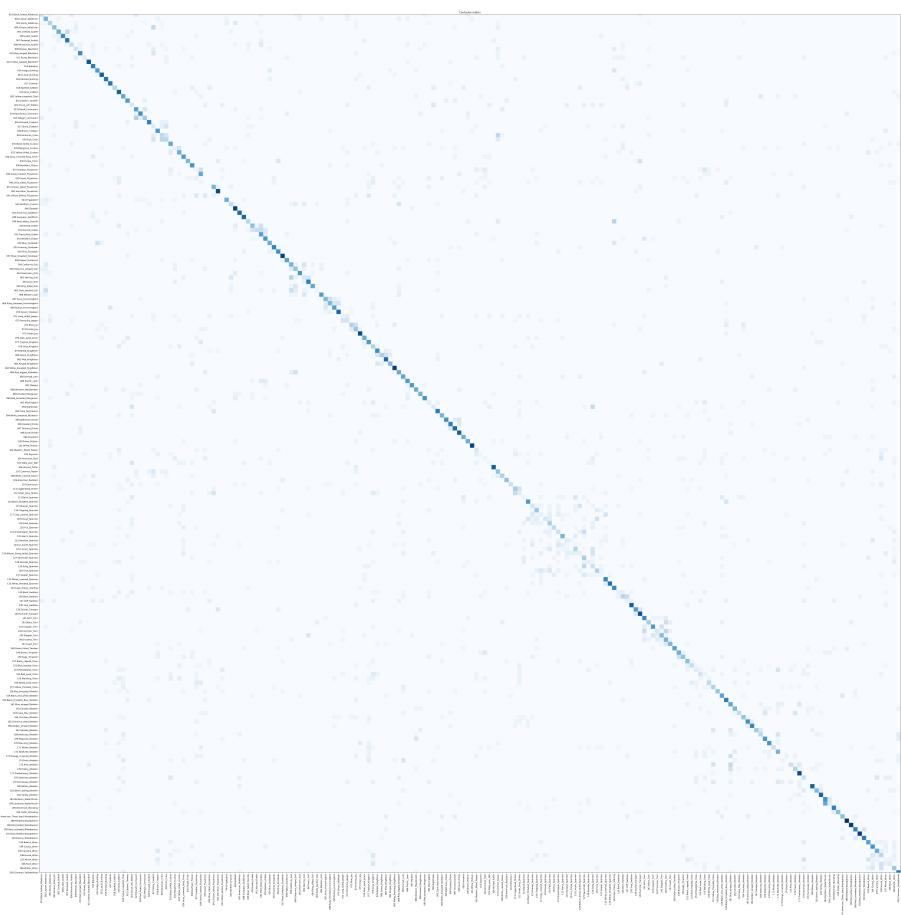
Gambar 3.66 Hasil dari listing 3.16

Lalu kita plot

```
1 import numpy as np
2 np.set_printoptions(precision=2)
3 plt.figure(figsize=(60,60), dpi=300)
4 plot_confusion_matrix(cm, classes=birds, normalize=True)
5 plt.savefig("E:/backup/sem 6/Kecerdasan Buatan/KB3C - Copy/
figures/1174083/figures3/ganti.png")
```

Listing 3.17 Plot hasil perubahan label

Hasilnya seperti berikut:



Gambar 3.67 Hasil dari listing 3.17

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

Kita coba menggunakan Decission tree

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

Listing 3.18 Mencoba klasifikasi dengan decission tree dengan dataset yang sama

Hasilnya seperti berikut:

```
In [23]: from sklearn import tree
.... clftree = tree.DecisionTreeClassifier()
.... clftree.fit(df_train_att, df_train_label)
.... clftree.score(df_test_att, df_test_label)
Out[23]: 0.2732312565997888
```

Gambar 3.68 Hasil dari listing 3.18

Kita coba menggunakan SVM

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

Listing 3.19 Mencoba klasifikasi dengan SVM dengan dataset yang sama

Hasilnya seperti berikut:

```
In [24]: from sklearn import svm
.... clfsvm = svm.SVC()
.... clfsvm.fit(df_train_att, df_train_label)
.... clfsvm.score(df_test_att, df_test_label)
C:\ProgramData\Anaconda3\lib\site-packages\sklearn\svm\base.py:193: FutureWarning: The
default value of gamma will change from 'auto' to 'scale' in version 0.22 to account better
for unscaled features. Set gamma explicitly to 'auto' or 'scale' to avoid this warning.
    "avoid this warning.", FutureWarning)
Out[24]: 0.27666314677930304
```

Gambar 3.69 Hasil dari listing 3.19

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

Pengecekan Cross Validation untuk Random Forest

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

Listing 3.20 Hasil Cross Validation random forest

Hasilnya seperti berikut:

```
In [25]: from sklearn.model_selection import cross_val_score
.... scores = cross_val_score(clf, df_train_att, df_train_label, cv=5)
.... # show average score and +/- two standard deviations away (covering 95% of scores)
.... print("Accuracy: %0.2f (+/- %0.2f)" % (scores.mean(), scores.std() * 2))
Accuracy: 0.44 (+/- 0.02)
```

Gambar 3.70 Hasil dari listing 3.20

untuk decision tree

```

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

```

Listing 3.21 Hasil Cross Validation Random Forest

Hasilnya seperti berikut:

```

In [26]: scorestree = cross_val_score(clftree, df_train_att, df_train_label, cv=5)
...: print("Accuracy: %0.2f (+/- %0.2f)" % (scorestree.mean(), scorestree.std() * 2))
Accuracy: 0.25 (+/- 0.02)

```

Gambar 3.71 Hasil dari listing 3.21

untuk SVM

```

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

```

Listing 3.22 Hasil Cross Validation SVM

Hasilnya seperti berikut:

```

In [27]: scoressvm = cross_val_score(clfsvm, df_train_att, df_train_label, cv=5)
...: print("Accuracy: %0.2f (+/- %0.2f)" % (scoressvm.mean(), scoressvm.std() * 2))
Accuracy: 0.26 (+/- 0.01)

```

Gambar 3.72 Hasil dari listing 3.22

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

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

```

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

```

```

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

```

Listing 3.23 Melakukan Pengamatan komponen informasi

Hasilnya seperti berikut:

```

(max_features, n_estimators, scores.mean(), scores.std() * 2))
Max features: 5, num estimators: 10, accuracy: 0.26 (+/- 0.02)
Max features: 5, num estimators: 30, accuracy: 0.36 (+/- 0.04)
Max features: 5, num estimators: 50, accuracy: 0.39 (+/- 0.02)
Max features: 5, num estimators: 70, accuracy: 0.40 (+/- 0.02)
Max features: 5, num estimators: 90, accuracy: 0.42 (+/- 0.02)
Max features: 5, num estimators: 110, accuracy: 0.42 (+/- 0.02)
Max features: 5, num estimators: 130, accuracy: 0.43 (+/- 0.03)
Max features: 5, num estimators: 150, accuracy: 0.43 (+/- 0.02)
Max features: 5, num estimators: 170, accuracy: 0.44 (+/- 0.02)
Max features: 5, num estimators: 190, accuracy: 0.44 (+/- 0.03)
Max features: 10, num estimators: 10, accuracy: 0.28 (+/- 0.03)
Max features: 10, num estimators: 30, accuracy: 0.38 (+/- 0.03)
Max features: 10, num estimators: 50, accuracy: 0.40 (+/- 0.03)
Max features: 10, num estimators: 70, accuracy: 0.42 (+/- 0.03)
Max features: 10, num estimators: 90, accuracy: 0.43 (+/- 0.03)
Max features: 10, num estimators: 110, accuracy: 0.43 (+/- 0.02)
Max features: 10, num estimators: 130, accuracy: 0.44 (+/- 0.03)
Max features: 10, num estimators: 150, accuracy: 0.44 (+/- 0.04)
Max features: 10, num estimators: 170, accuracy: 0.45 (+/- 0.02)
Max features: 10, num estimators: 190, accuracy: 0.45 (+/- 0.03)
Max features: 15, num estimators: 10, accuracy: 0.30 (+/- 0.03)
Max features: 15, num estimators: 30, accuracy: 0.38 (+/- 0.03)
Max features: 15, num estimators: 50, accuracy: 0.42 (+/- 0.03)
Max features: 15, num estimators: 70, accuracy: 0.43 (+/- 0.03)
Max features: 15, num estimators: 90, accuracy: 0.44 (+/- 0.04)
Max features: 15, num estimators: 110, accuracy: 0.44 (+/- 0.02)
Max features: 15, num estimators: 130, accuracy: 0.44 (+/- 0.02)
Max features: 15, num estimators: 150, accuracy: 0.45 (+/- 0.02)
Max features: 15, num estimators: 170, accuracy: 0.45 (+/- 0.03)
Max features: 15, num estimators: 190, accuracy: 0.45 (+/- 0.02)
Max features: 20, num estimators: 10, accuracy: 0.32 (+/- 0.01)
Max features: 20, num estimators: 30, accuracy: 0.39 (+/- 0.03)
Max features: 20, num estimators: 50, accuracy: 0.42 (+/- 0.02)
Max features: 20, num estimators: 70, accuracy: 0.43 (+/- 0.03)
Max features: 20, num estimators: 90, accuracy: 0.43 (+/- 0.02)
Max features: 20, num estimators: 110, accuracy: 0.44 (+/- 0.03)
Max features: 20, num estimators: 130, accuracy: 0.45 (+/- 0.03)

```

Gambar 3.73 Hasil dari listing 3.23

```

Max features: 30, num estimators: 10, accuracy: 0.32 (+/- 0.02)
Max features: 30, num estimators: 30, accuracy: 0.40 (+/- 0.02)
Max features: 30, num estimators: 50, accuracy: 0.42 (+/- 0.02)
Max features: 30, num estimators: 70, accuracy: 0.43 (+/- 0.01)
Max features: 30, num estimators: 90, accuracy: 0.44 (+/- 0.03)
Max features: 30, num estimators: 110, accuracy: 0.44 (+/- 0.02)
Max features: 30, num estimators: 130, accuracy: 0.45 (+/- 0.03)
Max features: 30, num estimators: 150, accuracy: 0.45 (+/- 0.03)
Max features: 30, num estimators: 170, accuracy: 0.45 (+/- 0.02)
Max features: 30, num estimators: 190, accuracy: 0.45 (+/- 0.03)
Max features: 35, num estimators: 10, accuracy: 0.33 (+/- 0.02)
Max features: 35, num estimators: 30, accuracy: 0.39 (+/- 0.02)
Max features: 35, num estimators: 50, accuracy: 0.42 (+/- 0.03)
Max features: 35, num estimators: 70, accuracy: 0.43 (+/- 0.02)
Max features: 35, num estimators: 90, accuracy: 0.44 (+/- 0.03)
Max features: 35, num estimators: 110, accuracy: 0.44 (+/- 0.02)
Max features: 35, num estimators: 130, accuracy: 0.44 (+/- 0.02)
Max features: 35, num estimators: 150, accuracy: 0.45 (+/- 0.03)
Max features: 35, num estimators: 170, accuracy: 0.45 (+/- 0.02)
Max features: 35, num estimators: 190, accuracy: 0.45 (+/- 0.02)
Max features: 40, num estimators: 10, accuracy: 0.33 (+/- 0.02)
Max features: 40, num estimators: 30, accuracy: 0.40 (+/- 0.01)
Max features: 40, num estimators: 50, accuracy: 0.42 (+/- 0.02)
Max features: 40, num estimators: 70, accuracy: 0.43 (+/- 0.02)
Max features: 40, num estimators: 90, accuracy: 0.44 (+/- 0.03)
Max features: 40, num estimators: 110, accuracy: 0.44 (+/- 0.02)
Max features: 40, num estimators: 130, accuracy: 0.45 (+/- 0.03)
Max features: 40, num estimators: 150, accuracy: 0.45 (+/- 0.03)
Max features: 40, num estimators: 170, accuracy: 0.45 (+/- 0.02)
Max features: 40, num estimators: 190, accuracy: 0.45 (+/- 0.03)
Max features: 45, num estimators: 10, accuracy: 0.33 (+/- 0.02)
Max features: 45, num estimators: 30, accuracy: 0.40 (+/- 0.03)
Max features: 45, num estimators: 50, accuracy: 0.42 (+/- 0.02)
Max features: 45, num estimators: 70, accuracy: 0.43 (+/- 0.03)
Max features: 45, num estimators: 90, accuracy: 0.44 (+/- 0.03)
Max features: 45, num estimators: 110, accuracy: 0.44 (+/- 0.02)
Max features: 45, num estimators: 130, accuracy: 0.44 (+/- 0.02)
Max features: 45, num estimators: 150, accuracy: 0.45 (+/- 0.02)
Max features: 45, num estimators: 170, accuracy: 0.45 (+/- 0.02)
Max features: 45, num estimators: 190, accuracy: 0.45 (+/- 0.02)
Max features: 45, num estimators: 190, accuracy: 0.45 (+/- 0.01)

```

Gambar 3.74 Hasil dari listing 3.23(2)

Dan kita bisa melakukan plot informasi ini dengan kode

```

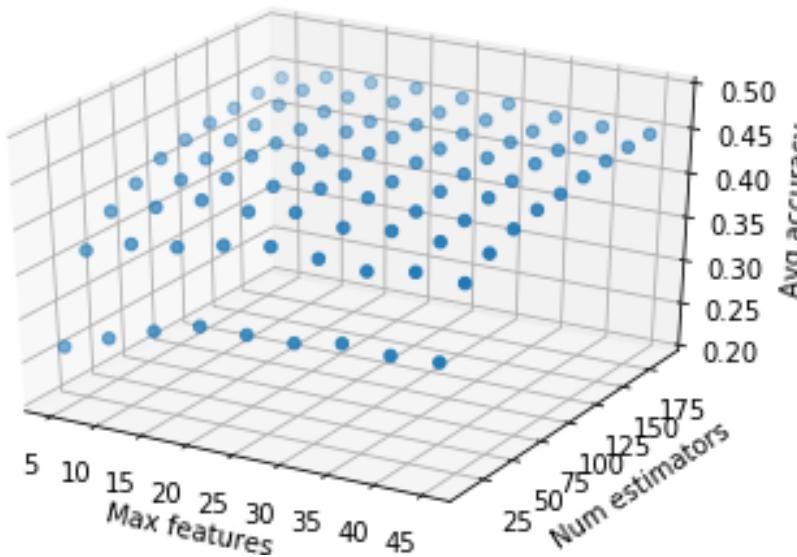
1 import matplotlib.pyplot as plt
2 from mpl_toolkits.mplot3d import Axes3D
3 from matplotlib import cm
4 fig = plt.figure()
5 fig.clf()
6 ax = fig.gca(projection='3d')
7 x = rf_params[:,0]
8 y = rf_params[:,1]
9 z = rf_params[:,2]
10 ax.scatter(x, y, z)
11 ax.set_zlim(0.2, 0.5)
12 ax.set_xlabel('Max features')

```

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

Listing 3.24 Plot Komponen informasi agar bisa dibaca

Hasilnya seperti berikut:



Gambar 3.75 Hasil dari listing 3.24

3.3.3 Penanganan Error

3.3.3.1 skrinsut Error

Untuk Error, saya tidak menemukannya. tetapi saya mengalami not responding beberapa kali. seperti pada gambar berikut, dimana not responding terjadi karena laptop dengan spek yang minim melakukan kinerja yang sangat berat.

The screenshot shows the Spyder Python IDE interface. The code editor contains a script named 'bird-identifier.py' with several print statements and plotting commands. The IPython console at the bottom shows the output of these commands, including a confusion matrix plot and a task manager window. The task manager shows various processes running, with Python (9) highlighted.

```

131     fm = ".2f" if normalize else 'd'
132     thresh = cm.max() / 2.
133
134     plt.tight_layout()
135     plt.ylabel('True label')
136     plt.xlabel('Predicted label')
137
138 # In[5,4]: Confusion Matrix
139
140 birds = pd.read_csv("E:/Backup/sem 6/Kecerdasan Buatan/CB_200_2011/classes.txt",
141 sep=',', header=None, usecols=[1], names=['birdname'])
142 birds
143 birds
144
145 # In[5,4]: Confusion Matrix
146
147 import numpy as np
148 np.set_printoptions(precision=2)
149 plt.figure(figsize=(60,60), dpi=300)
150 plot_confusion_matrix(cm, classes=birds, normalize=True)
151 plt.show()
152
153 # In[6]: Decision Tree dan SVM
154
155 from sklearn import tree
156 clftree = tree.DecisionTreeClassifier()
157 clftrain = clftree.fit(df_train_att, df_train_label)
158 clftree.score(df_test_att, df_test_label)
159 plt.savefig("E:/Backup/sem 6/Kecerdasan Buatan/X83C - Copy/figures/1174083/figures3/ga
160 # In[6]: Decision Tree dan SVM
161
162 from sklearn import svm
163 clfsvm = svm.SVC()
164 clfsvm.fit(df_train_att, df_train_label)
165 clfsvm.score(df_test_att, df_test_label)
166
167 # In[7]: Cross Validation
168
169

```

Gambar 3.76 Before Disaster

This screenshot shows the same Spyder environment as the previous one, but with a large red circle highlighting the Task Manager window. The Task Manager lists several processes, with Python (9) again highlighted. The IPython console at the bottom shows the continuation of the code execution, including the printing of a confusion matrix.

```

141
142 birds = birds[['birdname']]
143
144 # In[5,4]: Confusion Matrix
145
146 import numpy as np
147 np.set_printoptions(precision=2)
148 plt.figure(figsize=(60,60), dpi=300)
149 plot_confusion_matrix(cm, classes=birds, normalize=True)
150 plt.show()
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169

```

Gambar 3.77 After Disaster

3.3.3.2 Tuliskan Kode Error dan Jenis Error

The screenshot shows the Spyder Python IDE interface. The title bar says "Spyder (Python 3.7) (Not Responding)". The main area displays a code editor with several files open, including "bird-identifier.py", "praktikum.py", "Student performance.py", and "1174083.py". A "Variable explorer" window is open, showing a table with columns "Name", "Type", "Size", and "Value". The "Value" column contains numerical data. An "IPython console" window is also visible, showing some command history and output. The status bar at the bottom right shows "Permissions: RW", "End-of-line: LF", "Encoding: UTF-8", "Line: 156", "Column: 40", "Message: 275", and the date "18/03/2020".

```

# In[5.4]: Confusin Matrix
import numpy as np
np.set_printoptions(precision=2)
plt.figure(figsize=(60,60), dpi=300)
plot_confusion_matrix(cm, classes=birds, normalize=True)
plt.show()

```

Gambar 3.78 Not Responding pun terjadi

3.3.3.3 Cara Penanganan Error

Untuk mengatasinya, saya endtask terlebih dahulu spydernya. dan memodifikasi codingannya. yang asalnya

```
# In[5.4]: Confusin Matrix
```

```

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

```

Gambar 3.79 Sebelum diubah

```
# In[5.4]: Confusin Matrix
```

```

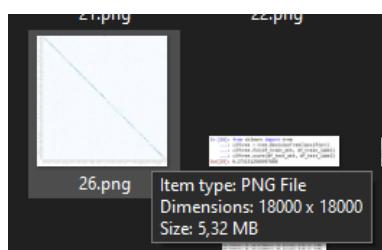
import numpy as np
np.set_printoptions(precision=2)
plt.figure(figsize=(60,60), dpi=300)
plot_confusion_matrix(cm, classes=birds, normalize=True)
plt.savefig("E:/backup/sem 6/Kecerdasan Buatan/KB3C - Copy/figures/1174083/figures3/ganti.png")

```

Gambar 3.80 Setelah diubah

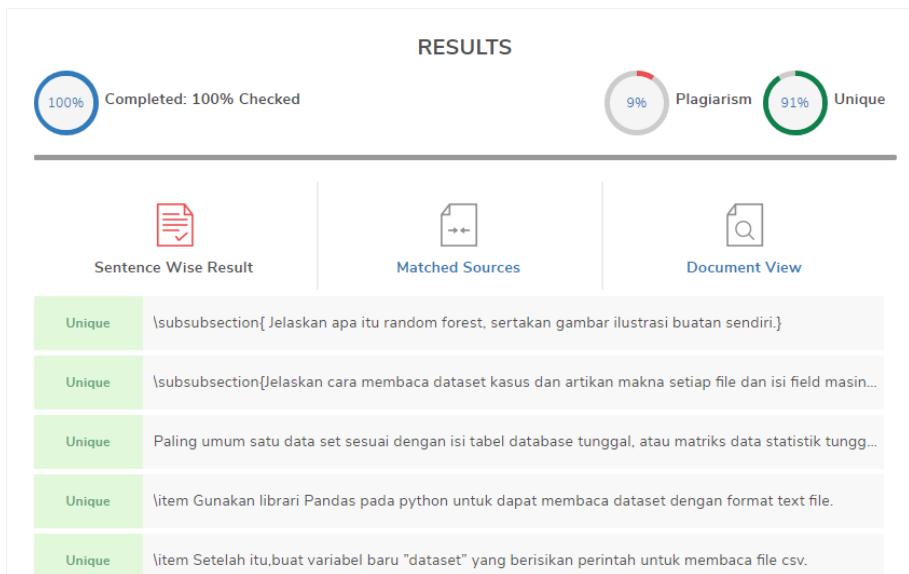
yang asalnya pada baris terakhir itu plt.show() yaitu untuk menampilkan-nya di console, saya ganti menjadi plt.savefig yang berfungsi menyimpan langsung gambar dari grafik tersebut. dan not respinding ini terjadi karena laptop berusaha menampilkan gambar dengan ukuran yang besar, yaitu 18000x18000px

tetapi speknya hanya tidak mampu maka terjadilah not responding. tetapi jika kita hanya langsung men save figurenya maka proses akan berjalan lancar meskipun agak lama.



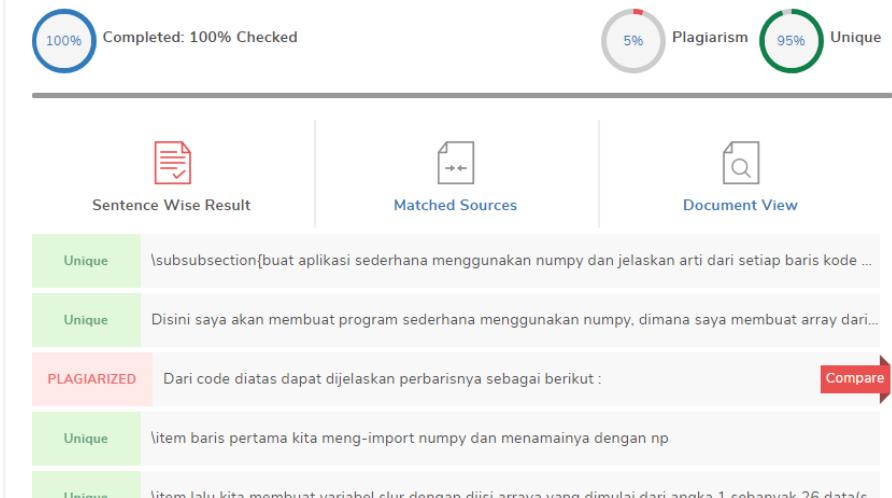
Gambar 3.81 Detail Gambar Confusion Matrix

3.3.3.4 *Bukti Tidak Plagiat*



Gambar 3.82 Cek Plagiarism

RESULTS



Gambar 3.83 Cek Plagiarism(2)

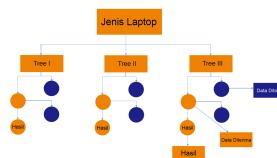
3.3.4 Link Video Youtube

<https://youtu.be/DyJRzPrFxQ>

3.4 1174069 - Fanny Shafira

3.4.1 Soal Teori

1. Jelaskan apa itu random forest, sertakan gambar ilustrasi buatan sendiri. Random Forest merupakan algoritma yang digunakan terhadap klasifikasi data dalam jumlah yang besar. Klasifikasi pada random forest dilakukan dengan penggabungan dicision tree dengan melakuakn training terhadap sempel data yang dimiliki. Semakin banyak dicision tree maka data yang di dapat akan semakin akurat. Untuk gambar Random Forest dapat dilihat dibawah ini :



Gambar 3.84 Random Forest.

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

Cross Validation merupakan sebuah teknik validasi model yang berfungsi untuk melakukan penilaian bagaimana hasil analisis statistik akan digeneralisasi ke data yang lebih independen. Cross validation digunakan dengan tujuan prediksi, dan bila kita ingin memperkirakan seberapa akurat model prediksi yang dilakukan dalam sebuah praktik. Tujuan dari cross validation yaitu untuk mendefinisikan dataset guna menguji dalam fase pelatihan untuk membatasi masalah seperti overfitting dan underfitting serta mendapatkan wawasan tentang bagaimana model akan digeneralisasikan ke set data independen.
4. Jelaskan apa arti score 44 % pada random forest, 27% pada decision tree dan 29 % dari SVM.

Dimana Score 44 % diperoleh dari hasil pengelahan dataset jenis burung. Dimana akan dilakukan proses pembagian data testing dan data training lalu diproses dan menghasilkan score sebanyak 44 % dimana menjelaskan bahwa score tersebut digunakan sebagai pembanding dalam tingkat keakuratannya. Pada decision tree akan memperoleh data lebih kecil yaitu sebanyak 27 % hal ini dikarenakan data yang diolah menggunakan decision tree dibagi menjadi beberapa tree dan lalu disimpulkan untuk mendapatkan data yang akurat. Pada SVM akan memperoleh score sebanyak 29 % hal ini dikarenakan data yang dimiliki masih bernilai netral sehingga tingkat keakuratannya masih belum jelas.
5. Jelaskan bagaimana cara membaca confusion matriks dan contohnya memakai gambar atau ilustrasi sendiri.

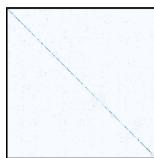
Untuk membaca confusion matriks dapat menggunakan source code sebagai berikut :

```

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

```

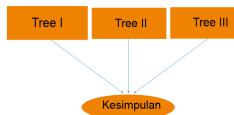
Dimana numpy akan mengurus semua data yang berhubungan dengan matrix. Pada source code tersebut digunakan dalam melakukan read pada dataset burung dengan menggunakan metode confusion matrix. Dalam confusion matrix memiliki 4 istilah yaitu True Positive yang merupakan data posotif yang terditeksi benar,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



Gambar 3.85 Confusion Matriks.

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

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



Gambar 3.86 Decision Tree.

Dimana ditunjukkan pada gambar diatas terdapat 3 tree. Dalam tree tersebut akan dilakukan proses voting. Saya akan memberikan contoh kasus, dimana akan diadakan voting untuk menentukan sebuah laptop. Dalam tree akan diberikan sejumlah data misalnya saja data tersebut berupa gambar, yang dimana data tersebut akan dipilih dengan cara voting. Hasil voting akhir dari setiap tree menunjukkan laptop asus,

yang berarti kesimpulan dari data yang telah diberikan menyatakan gambar tersebut adalah laptop asus. Bagaimana apabila terjadi perbedaan data misalnya saja pada tree 1 dan 2 menunyatakan laptop asus sedangkan pada tree 3 menyatakan laptop HP, maka kesimpulan yang di ambil adalah laptop asus dikarenakan hasil voting terbanyak adalah laptop asus.

3.4.2 Praktek Program

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

```

1 # In [44]: Soal1
2
3 import pandas as fanny #melakukan import pada library pandas
4          sebagai fanny
5 boyband = {"Boyband" : [ 'EXO' , 'SEVENTEEN' , 'DAY6' , 'IKON' ]} #
6          membuat varibel yang bernama boyband , dan mengisi
7          dataframe nama2 boyband

```

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

	Boyband
0	Boyband kesukaan Fanny EXO
1	Boyband kesukaan Fanny SEVENTEEN
2	Boyband kesukaan Fanny DAY6
3	Boyband kesukaan Fanny IKON

Gambar 3.87 Hasil Soal 1.

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

```

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

```

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

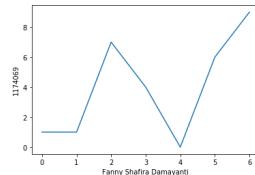
```
[[1. 0. 0. 0. 0. 0. 0. 0. 0.]
 [0. 1. 0. 0. 0. 0. 0. 0. 0.]
 [0. 0. 1. 0. 0. 0. 0. 0. 0.]
 [0. 0. 0. 1. 0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 1. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0. 1. 0. 0. 0.]
 [0. 0. 0. 0. 0. 0. 1. 0. 0.]
 [0. 0. 0. 0. 0. 0. 0. 1. 0.]
 [0. 0. 0. 0. 0. 0. 0. 0. 1.]
 [0. 0. 0. 0. 0. 0. 0. 0. 1.]]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

Gambar 3.88 Hasil Soal 2.

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

```
1 import matplotlib.pyplot as fanny #import matplotlib sebagai
   fanny
2
3 fanny.plot([1,1,7,4,0,6,9]) #memberikan nilai plot atau
   grafik pada fanny
4 fanny.xlabel('Fanny Shafira Damayanti') #memberikan label
   pada x
5 fanny.ylabel('1174069') #memberikan label pada y
6 fanny.show() #print hasil plot berbentuk grafik
```

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

**Gambar 3.89** Hasil Soal 3.

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

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

```
1
2 import pandas as pd #import library pandas sebagai
3
4 imgatt = pd.read_csv("data/CUB_200_2011/attributes/
   image_attribute_labels.txt",
```

```
5             sep='\\s+', header=None,  
6             error_bad_lines=False, warn_bad_lines=False,  
7             usecols=[0,1,2], names=['imgid', '  
attid', 'present']) #library imgatt sebagai membaca  
file csv dari dataset, dengan ketentuan yang ada.
```

Hasilnya adalah seperti ini :

Gambar 3.90 Hasil Soal 4 - 1

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

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

Hasilnya adalah seperti ini :

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

Gambar 3.91 Hasil Soal 4 - 2

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

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

Hasilnya adalah seperti ini :

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

Gambar 3.92 Hasil Soal 4 - 3

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

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

Hasilnya adalah seperti ini :

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

Gambar 3.93 Hasil Soal 4 - 4

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

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

Hasilnya adalah seperti ini :

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

[5 rows x 312 columns]

Gambar 3.94 Hasil Soal 4 - 5

- Menghasilkan jumlah kolom dan baris pada DataFrame imgatt2.

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

Hasilnya adalah seperti ini :

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

Gambar 3.95 Hasil Soal 4 - 6

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

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

Hasilnya adalah seperti ini :

```
In [8]: imglabels = pd.read_csv('data/tin/tin2011/image_class_labels.txt',
                           sep=' ', header=None, names=['imgid', 'label'])

In [8]: imglabels
Out[8]:
```

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

Gambar 3.96 Hasil Soal 4 - 7

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

```
In [9]: imglabels.head()
Out[9]:
```

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

Hasilnya adalah seperti ini :

```
In [9]: imglabels.head()
Out[9]:
```

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

Gambar 3.97 Hasil Soal 4 - 8

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

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

Hasilnya adalah seperti ini :

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

Gambar 3.98 Hasil Soal 4 - 9

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

```

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

```

Hasilnya adalah seperti ini :

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

Gambar 3.99 Hasil Soal 4 - 10

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

```

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

```

Hasilnya adalah seperti ini :

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

Gambar 3.100 Hasil Soal 4 - 11

- Mengecek isi 5 data teratas pada df att.

```

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

```

Hasilnya adalah seperti ini :

In [13]: df_att.head()														
Out[13]:														
1 2 3 4 5 6 7 ... 306 307 308 309 310 311 312														
imgatt	0	0	0	0	0	1	...	0	0	0	0	0	0	1
2449	0	0	0	0	0	0	...	0	0	0	0	0	0	0
820	0	0	0	0	0	0	...	0	0	0	0	0	0	0
7472	0	0	0	0	0	0	...	0	0	0	0	0	0	0
7368	0	0	0	0	0	0	...	0	0	0	0	0	0	0
9854	0	0	0	0	0	1	...	0	0	1	0	0	0	0

Gambar 3.101 Hasil Soal 4 - 12

- Mengecek isi data teratas dari df label.

```

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

```

Hasilnya adalah seperti ini :

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

Gambar 3.102 Hasil Soal 4 - 13

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

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

Hasilnya adalah seperti ini :

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

Gambar 3.103 Hasil Soal 4 - 14

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

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

Hasilnya adalah seperti ini :

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

Gambar 3.104 Hasil Soal 4 - 15

- Menunjukkan hasil prediksi dari Random Forest.

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

Hasilnya adalah seperti ini :

```
In [17]: clf.fit(df_train_att, df_train_label)
Out[17]: RandomForestClassifier(bootstrap=True, ccp_alpha=0.0, class_weight=None,
                                criterion='gini', max_depth=None, max_features='auto', max_leaf_nodes=None, max_samples=None,
                                min_impurity_decrease=0.0, min_impurity_split=None,
                                min_leaf_node_size=1, min_weight_fraction_in_leaf=0.0,
                                min_weight_fraction_leaf=0.0, n_estimators=100,
                                n_jobs=None, oob_score=False, random_state=42, verbose=0,
                                warm_start=False)
```

Gambar 3.105 Hasil Soal 4 - 16

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

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

Hasilnya adalah seperti ini :

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

Gambar 3.106 Hasil Soal 4 - 17

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

```
1
2 clf.score(df_test_att, df_test_label) #print clf sebagai
   testing yang sudah di training tadi
```

Hasilnya adalah seperti ini :

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

Gambar 3.107 Hasil Soal 4 - 18

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

- Melakukan import confusion matrix pada library sklearn matriks.

```
1
2 from sklearn.metrics import confusion_matrix #import
   Confusion Matrix
```

```

3 pred_labels = clf.predict(df_test_att) #sebagai data
   testing
4 cm = confusion_matrix(df_test_label, pred_labels) #cm
   sebagai variabel data label

```

Hasilnya adalah seperti ini :

```

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

```

Gambar 3.108 Hasil Soal 5 - 1

- Menampilkan isi cm dalam bentuk matrix yang berupa array.

```

1
2 cm #menampilkan data label berbentuk array

```

Hasilnya adalah seperti ini :

```

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

```

Gambar 3.109 Hasil Soal 5 - 2

- Membuat dan menampilkan hasil plot.

```

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

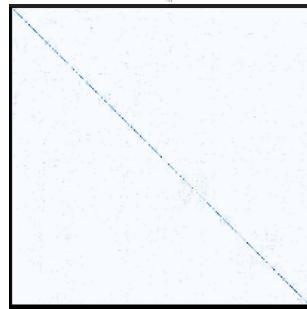
```

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

Hasilnya adalah seperti dibawah ini :

```
In [20]: import numpy as np
... import intervals
... def plot_confusion_matrix(cm, classes,
...                         normalize=False,
...                         title='Confusion matrix',
...                         cmap=plt.cm.Blues):
...     """
...     This function prints and plots the confusion matrix.
...     Normalization can be applied by setting `normalize=True`.
...     """
...     if normalize:
...         cm = cm.astype('float') / cm.sum(axis=1)[:, np.newaxis]
...     print("Normalized confusion matrix")
...     print(cm)
...
...     plt.imshow(cm, interpolation='nearest', cmap=cmap)
...     plt.title(title)
...     plt.colorbar()
...     tick_marks = np.arange(len(classes))
...     plt.xticks(tick_marks, classes, rotation=45)
...     plt.yticks(tick_marks, classes)
...
...     if normalize:
...         thresh = cm.max() / 2.
...     else:
...         thresh = cm[0][0] / 2.
...
...     for i, j in product(range(cm.shape[0]), range(cm.shape[1])):
...         plt.text(j, i, cm[i, j],
...                  horizontalalignment="center",
...                  verticalalignment="center",
...                  color="white" if cm[i, j] > thresh else "black")
...
...     plt.tight_layout()
...     plt.ylabel('True label')
...     plt.xlabel('Predicted label')
...
... 
```

```
In [21]: birds = pd.read_csv("data/CUB_200_2011/classes.txt",
...                         header=None, usecols=[1], names=['birdname'])
...
... bird_ids = birds['birdname'].index
...
... Out[21]:
0    001.Black_Faced_Albatross
1    002.Brown_Albatross
2    003.Sooty_Albatross
3    004.Blue_Face_Albatross
4    005.Crested_Albatross
...
195   158.Mouse_bird
196   159.Tropic_Bird
197   150.Rock_bird
198   151.Wirebird
199   200.Common_Valley_Wrent
Name: birdname, Length: 200, dtype: object
```



Gambar 3.110 Menampilkan hasil plot

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

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

```
1 from sklearn import tree #import library tree
2 clftree = tree.DecisionTreeClassifier() #clftrree sebagai
3 variabel untuk decision tree
4 clftrree.fit(df_train_att , df_train_label) #sebagai data
5 training
6 clftrree.score(df_test_att , df_test_label) #sebagai data
testing
```

Hasilnya adalah seperti ini :

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

Gambar 3.111 Hasil Soal 6 - 1

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

```

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

```

Hasilnya adalah seperti ini :

```

In [28]: from sklearn import svm
...: clfsvm = svm.SVC()
...: clfsvm.fit(df_train_att, df_train_label)
...: clfsvm.score(df_test_att, df_test_label)
Out[28]: 0.47729672690475186

```

Gambar 3.112 Hasil Soal 6 - 2

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

- Menunjukkan hasil cross validation pada Random Forest.

```

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

```

Hasilnya adalah seperti ini :

```

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

```

Gambar 3.113 Hasil Soal 7 - 1

- Menunjukkan hasil cross validation pada Desicion Tree.

```

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

```

Hasilnya adalah seperti ini :

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

Gambar 3.114 Hasil Soal 7 - 2

- Menunjukkan hasil cross validation pada SVM.

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

Hasilnya adalah seperti ini :

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

Gambar 3.115 Hasil Soal 7 - 3

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

- Menunjukkan informasi-informasi tree yang dibuat.

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

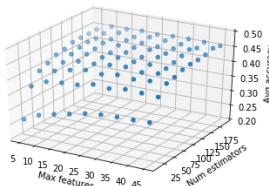
Hasilnya adalah seperti ini :

Gambar 3.116 Hasil Soal 8 - 1

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

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

Hasilnya adalah seperti ini :



Gambar 3.117 Hasil Soal 8 - 2

3.4.3 Penanganan Error

1. ScreenShoot Error



```
F:\Data\RandomForest> (from: 2) F:\Data\RandomForest>attribute_label.txt
F:\Data\RandomForest>attribute_label.txt
```

Gambar 3.118 FileNotFoundError

2. Tuliskan Kode Error dan Jenis Error

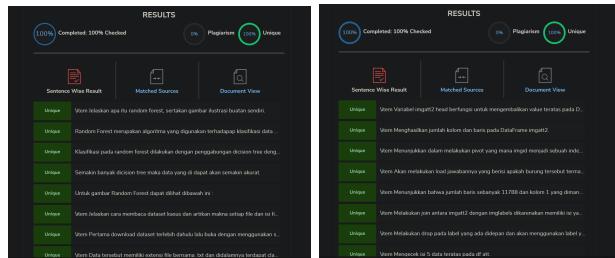
- FileNotFoundError

3. Cara Penangan Error

- FileNotFoundError

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

3.4.4 Bukti Tidak Plagiat



Gambar 3.119 Bukti Tidak Melakukan Plagiat Chapter 3

3.4.5 Link Youtube

<https://youtu.be/0B2-XZUiylk>

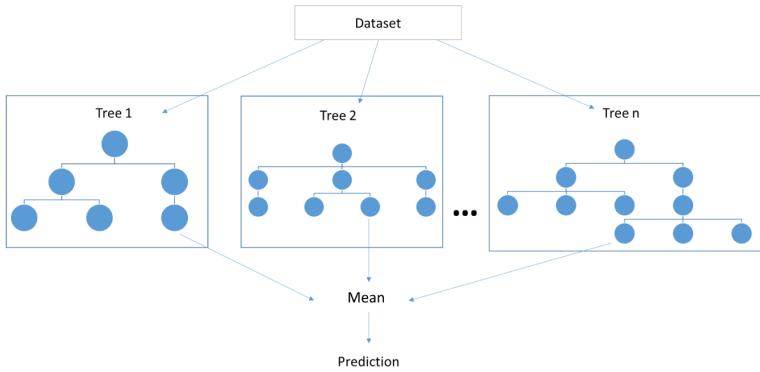
3.5 1174087 - Ilham Muhammad Ariq

Chapter 3 - Prediksi dengan Random Forest

3.5.1 Teori

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

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



Gambar 3.120 Random forest

3.5.1.2 Jelaskan cara membaca dataset kasus dan artikan makna setiap file dan isi field masing masing file.

1. cara membaca dataset

- Gunakan librari Pandas pada python untuk dapat membaca dataset dengan format text file.
- Setelah itu,buat variabel baru ”dataset” yang berisikan perintah untuk membaca file csv. seperti berikut

```

1 import pandas as pd
2 dataset = pd.read_csv("E:/Kecerdasan Buatan/chapter3/KB3C/
  src/1174087/3/dataset.csv", sep=',')
3 dataset.head()
  
```

Pada kodingan diatas dapat dijelaskan bahwa : Memanggil Librari Panda untuk membaca dataset Membuat variabel ”Dataset” yang berisikan pd.read_csv untuk membaca dataset.

- setelah di run maka hasilnya seperti berikut

Index	Kecamatan	Jumlah Makam	Kapasitas
0	KECAMATAN BATU	27	132082
1	Desa Sumberjo	4	12130
2	Kelurahan Songgorerto	3	29900
3	Desa Pesanggrahan	4	18751
4	Kelurahan Ngaglik	4	20500
5	Kelurahan Sirih	2	14800
6	Desa Sidomulyo	4	13800
7	Kelurahan Tempe	3	7601
8	Desa Oro Oro Gede	3	16800
9	KECAMATAN BUMIAJI	45	142125
10	Desa Giripurno	3	17500
11	Desa Pandanrejo	2	10700
12	Desa Bumiaji	5	21200
13	Desa Bulukerto	5	18250
14	Desa Punter	2	9000
15	Desa Tulungrejo	9	25800
16	Desa Sumbergondo	7	7075
17	Desa Gununganari	10	18400
18	Desa Sumberbrantas	2	15000
19	KECAMATAN JUNREJO	23	94145
20	Desa Beji	2	15000

Gambar 3.121 contoh dataset

Gambar diatas merupakan dataset Jumlah Tempat Pemakaman Umum (TPU) Kota Batu 2019. Datasetnya didapat dari laman <https://data.go.id/dataset-pemakaman-umum-tpu-kota-batu-2019>. Penjelasan dari isi field diatas adalah sebagai berikut :

- Atribut Index merupakan atribut otomatis untuk penomoran data yang ada.
- Atribut kecamatan merupakan kecamatan yang ada pada kota batu
- Atribut jumlah merupakan jumlah makam yang ada pada kecamatan
- Atribut kapasitas merupakan daya tampung pemakaman pada kecamatan

3.5.1.3 Jelaskan apa itu cross validation

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

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

Dimana Score 44 % didapatkan dari hasil pengelohan dataset jenis burung.

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

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

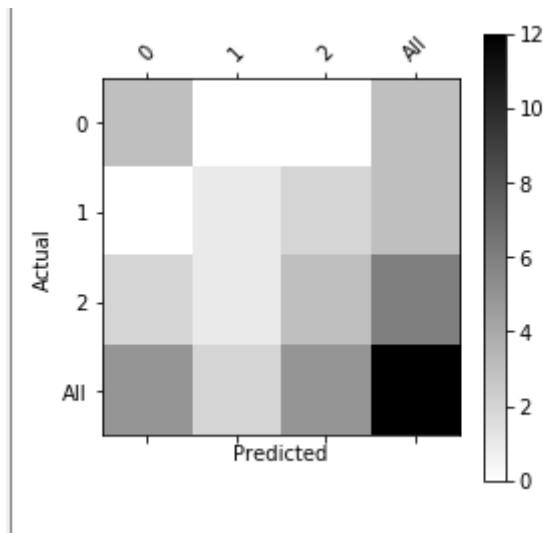
Perhitungan Confusion Matriks dapat dilakukan sebagai berikut. Disini saya menggunakan data yang dibuat sendiri untuk menampilkan data aktual dan prediksi.

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

```
1 import pandas as pd
2 import numpy as np
3 import matplotlib.pyplot as plt
4
5 y_actu = pd.Series([2, 0, 2, 2, 0, 1, 1, 2, 2, 0, 1, 2], name
6                   ='Actual')
7 y_pred = pd.Series([0, 0, 2, 1, 0, 2, 1, 0, 2, 0, 2, 2], name
8                   ='Predicted')
9 df_confusion = pd.crosstab(y_actu, y_pred, rownames=[ 'Actual' ,
10                           ], colnames=[ 'Predicted' ], margins=True)
11
12 def plot_confusion_matrix(df_confusion , title='Confusion
13                           matrix' , cmap=plt.cm.gray_r):
14     plt.matshow(df.confusion , cmap=cmap) # imshow
15     #plt.title(title)
16     plt.colorbar()
17     tick_marks = np.arange(len(df.confusion.columns))
18     plt.xticks(tick_marks , df.confusion.columns , rotation=45)
19     plt.yticks(tick_marks , df.confusion.index)
```

```
16     plt.ylabel(df_confusion.index.name)
17     plt.xlabel(df_confusion.columns.name)
18
19 plot_confusion_matrix(df_confusion)
```

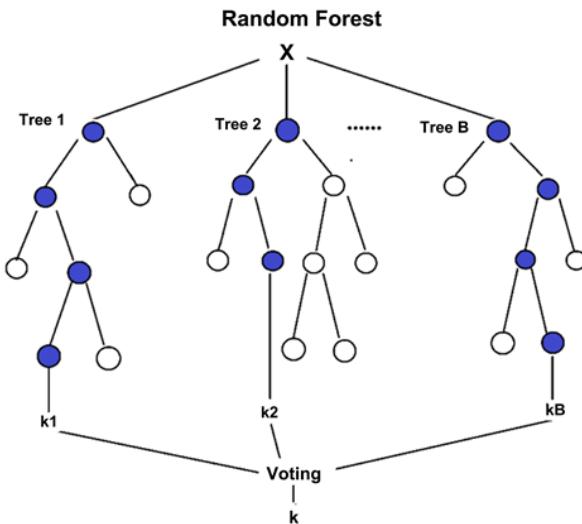
Hasil dari kode diatas akan menghasilkan confusion matrix seperti berikut:



Gambar 3.122 hasil confusion matrix

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

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



Gambar 3.123 voting random forest

3.5.2 Praktek

3.5.2.1 Nomor 1

```

1 import pandas as pd # Melakukan import library pandas sebagai pd
2
3 biodata = { 'Nama': [ 'Ariq' , 'Reza' , 'Alvan' ] , 'hobi': [ 'Berenang' , 'Tidur' , 'Game' ] , 'Umur': [17,18,19] }
4 #Membuat varibel yang bernama biodata , dan mengisi dataframanya dengan 3 field yaitu nama hobi dan umur
5 data = pd.DataFrame(biodata) # Membuat variabel data yang akan membuat DataFrame biodata menggunakan library pandas.
6 print(data) #print hasil dari data
  
```

	Nama	hobi	Umur
0	Ariq	Berenang	17
1	Reza	Tidur	18
2	Alvan	Game	19

Gambar 3.124 Membuat Aplikasi pakai pandas

3.5.2.2 Nomor 2

```

1 import numpy as np #Melakukan import library numpy sebaaai np
2 matrix = np.eye(5) #Membuat sebuah matriks identitas 5x5 pake
    numpy dengan menggunakan fungsi eye
3 print(matrix) #print matrix

```

```

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

```

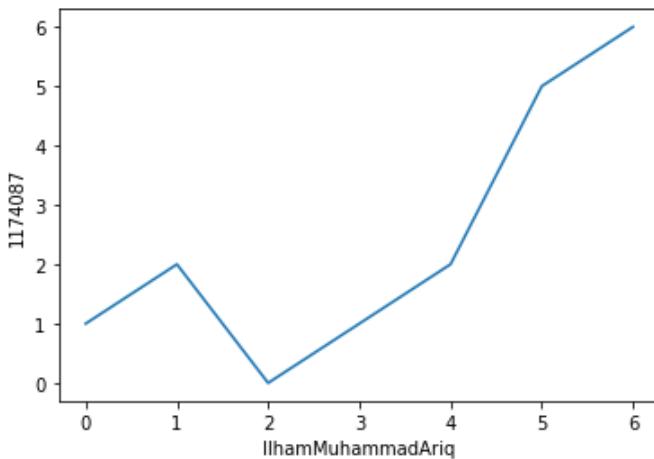
Gambar 3.125 Membuat Aplikasi pakai numpy

3.5.2.3 Nomor 3

```

1 import matplotlib.pyplot as pt #Melakukan import library numpy
    menjadi nama sendiri yaitu dirga
2 pt.plot([1,2,0,1,2,5,6]) #Memasukkan nilai pada plot
3 pt.xlabel('IlhamMuhammadAriq') #Menambahkan label pada x
4 pt.ylabel('1174087') #Menambahkan label pada y
5 pt.show() #Menampilkan grafik plot

```



Gambar 3.126 Membuat Aplikasi pakai matplotlib

3.5.2.4 Nomor 4

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

```
1 import pandas as pd #Melakukan import library numpy menjadi pd
2 imgatt = pd.read_csv("E:/Kecerdasan Buatan/CUB_200_2011/
    attributes/image_attribute_labels.txt",
3                         sep='\s+', header=None, error_bad_lines=
    False, warn_bad_lines=False,
4                         usecols=[0,1,2], names=['imgid', 'attid',
    , 'present']) #Membuat variable imgatt untuk membaca file
    csv dari dataset
```

```
In [4]: import pandas as pd #Melakukan import library numpy menjadi pd
...
...: imgatt = pd.read_csv("E:/Kecerdasan Buatan/CUB_200_2011/attributes/image_attribute_labels.txt",
...:                         sep='\s+', header=None, error_bad_lines=False, warn_bad_lines=False,
...:                         usecols=[0,1,2], names=['imgid', 'attid', 'present']) #Membuat variable imgatt untuk
membaca file csv dari dataset, dengan ketentuan yang ada.
```

Gambar 3.127 Hasil 4 Bagian 1

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

```
1 imgatt.head() #Menampilkan data yang sudah dibaca, data
    paling atas
```

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

Gambar 3.128 Hasil 4 Bagian 2

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

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

```
In [6]: imgatt.shape #Menampilkan jumlah seluruh data, kolom-nya
Out[6]: (3677856, 3)
```

Gambar 3.129 Hasil 4 Bagian 3

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

```
1 imgatt2 = imgatt.pivot(index='imgid', columns='attid', values
                         ='present') #Membuat sebuah variabel baru dari fungsi
                           imgatt, dengan mengganti index menjadi kolom dan kolom
                           menjadi index
```

```
In [7]: imgatt2 = imgatt.pivot(index='imgid', columns='attid', values='present') #Membuat sebuah variabel baru
dari fungsi imgatt, dengan mengganti index menjadi kolom dan kolom menjadi index
```

Gambar 3.130 Hasil 4 Bagian 4

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

```
1 imgatt2.head() #Menampilkan data yang sudah dibaca tadi tapi
                  cuman data paling atas
```

```

attid 1 2 3 4 5 6 7 ... 306 307 308 309 310 311 312
imgid
1 0 0 0 0 1 0 0 ...
2 0 0 0 0 0 0 0 ...
3 0 0 0 0 1 0 0 ...
4 0 0 0 0 0 1 0 0 ...
5 0 0 0 0 1 0 0 ...

```

[5 rows x 312 columns]

Gambar 3.131 Hasil 4 Bagian 5

- Menghasilkan jumlah kolom dan baris pada DataFrame imgatt2.

```
1 imgatt2.shape #Menampilkan jumlah seluruh data , kolom-nya
```

```
In [15]: imgatt2.shape #Menampilkan jumlah seluruh data, kolom-nya
Out[15]: (11788, 312)
```

Gambar 3.132 Hasil 4 Bagian 6

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

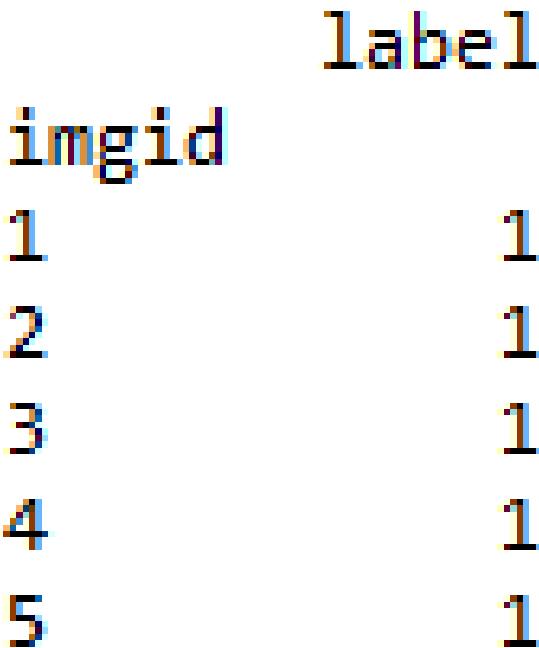
```
1 imglabels = pd.read_csv("E:/Kecerdasan Buatan/CUB_200_2011/
    image_class_labels.txt",
2                         sep=',', header=None, names=['imgid',
3                                     'label']) #Membaca data dimasukkan ke variable imglabels
4 imglabels = imglabels.set_index('imgid') #Variable imglabels
    dan set index (imgid)
```

```
In [16]: imglabels = pd.read_csv("E:/Kecerdasan Buatan/CUB_200_2011/image_class_labels.txt",
...:                         sep=',', header=None, names=['imgid', 'label']) #Membaca data dimasukkan
ke variable imglabels
...: imglabels = imglabels.set_index('imgid') #Variable imglabels dan set index (imgid)
```

Gambar 3.133 Hasil 4 Bagian 7

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

```
1 imglabels.head() #Menampilkan data yang sudah dibaca tadi
    tapi cuman data paling atas
```

**Gambar 3.134** Hasil 4 Bagian 8

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

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

```
In [18]: imglabels.shape #Menampilkan jumlah seluruh data, kolom-nya
Out[18]: (11788, 1)
```

Gambar 3.135 Hasil 4 Bagian 9

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

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

```
In [19]: df = imgatt2.join(imglabels) #Varibel df dimasukkan fungsi join dari data imgatt2 ke variabel
imglabels
...: df = df.sample(frac=1) #Varibel df sebagai sample dengan ketentuan frac=1
```

Gambar 3.136 Hasil 4 Bagian 10

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

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

```
In [20]: df_att = df.iloc[:, :312] #Membuat kolom dengan ketentuan 312
...: df_label = df.iloc[:, 312:] #Membuat kolom dengan ketentuan 312
```

Gambar 3.137 Hasil 4 Bagian 11

- Mengecek isi 5 data teratas pada df_att.

```
1 df_att.head() #Menampilkan data yang sudah dibaca tadi tapi
cuman data paling atas
```

```
In [26]: df_att.head() #Menampilkan data yang sudah dibaca tadi tapi cuman data paling atas
Out[26]:
```

1	2	3	4	5	6	7	...	306	307	308	309	310	311	312	
imgid							...								
10397	0	0	0	0	0	0	1	...	0	0	0	0	0	0	1
11088	0	0	0	0	0	0	1	...	0	0	0	1	0	0	0
2382	0	0	0	0	0	0	1	...	0	1	0	1	0	0	0
9374	0	0	0	0	0	0	1	...	0	0	0	0	0	0	0
9371	0	0	0	0	0	0	0	...	0	0	0	1	0	0	0

[5 rows x 312 columns]

Gambar 3.138 Hasil 4 Bagian 12

- Mengecek isi data teratas dari df_label.

```
1 df_label.head() #Menampilkan data yang sudah dibaca tadi tapi
cuman data paling atas
```

label	imgid
62	3574
149	8745
187	10967
199	11719
185	10878

Gambar 3.139 Hasil 4 Bagian 13

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

```

1 df_train_att = df_att[:8000] #Data akan dibagi dari 8000 row
   pertama menjadi data training dan sisanya adalah data
   testing
2 df_train_label = df_label[:8000] #Data akan dibagi dari 8000
   row pertama menjadi data training dan sisanya adalah data
   testing
3 df_test_att = df_att[8000:] #Berbalik dari sebelumnya data
   akan dibagi mulai dari 8000 row pertama menjadi data
   training dan sisanya adalah data testing
4 df_test_label = df_label[8000:] #Berbalik dari sebelumnya
   data akan dibagi mulai dari 8000 row pertama menjadi data
   training dan sisanya adalah data testing
5
6 df_train_label = df_train_label['label'] #Menambahkan label
7 df_test_label = df_test_label['label'] #Menambahkan label

```

```

In [23]: df_train_att = df_att[:8000] #Data akan dibagi dari 8000 row pertama menjadi data training dan
sisanya adalah data testing
...: df_train_label = df_label[:8000] #Data akan dibagi dari 8000 row pertama menjadi data training dan
sisanya adalah data testing
...: df_test_att = df_att[8000:] #Berbalik dari sebelumnya data akan dibagi mulai dari 8000 row pertama
menjadi data training dan sisanya adalah data testing
...: df_test_label = df_label[8000:] #Berbalik dari sebelumnya data akan dibagi mulai dari 8000 row
pertama menjadi data training dan sisanya adalah data testing
...:
...: df_train_label = df_train_label['label'] #Menambahkan Label
...: df_test_label = df_test_label['label'] #Menambahkan Label

```

Gambar 3.140 Hasil 4 Bagian 14

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

```

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

```

```

In [24]: from sklearn.ensemble import RandomForestClassifier #Import fungsi randomforestclassifier
...: clf = RandomForestClassifier(max_features=50, random_state=0, n_estimators=100) #clf sebagai variabel untuk klafifikasi random forest

```

Gambar 3.141 Hasil 4 Bagian 15

- Menunjukkan hasil prediksi dari Random Forest.

```

1 clf.fit(df_train_att, df_train_label) #Variabnle clf untuk fit yaitu menjadi data training

```

```

In [25]: clf.fit(df_train_att, df_train_label) #Variabnle clf untuk fit yaitu menjadi data training
Out[25]:
RandomForestClassifier(bootstrap=True, class_weight=None, criterion='gini',
   max_depth=None, max_features=50, max_leaf_nodes=None,
   min_impurity_decrease=0.0, min_impurity_split=None,
   min_samples_leaf=1, min_samples_split=2,
   min_weight_fraction_leaf=0.0, n_estimators=100,
   n_jobs=None, oob_score=False, random_state=0, verbose=0,
   warm_start=False)

```

Gambar 3.142 Hasil 4 Bagian 16

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

```

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

```

```

In [26]: print(clf.predict(df_train_att.head())) #Print clf yang di sudah prediksi dari training tetapi hanya menampilkan data paling atas
[ 62 149 187 199 185]

```

Gambar 3.143 Hasil 4 Bagian 17

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

```

1 clf.score(df_test_att, df_test_label) #Memunculkan clf sebagai testing yang sudah di training tadi

```

```

In [28]: clf.score(df_test_att, df_test_label) #Memunculkan clf sebagai testing yang sudah di training tadi
Out[28]: 0.4390179514255544

```

Gambar 3.144 Hasil 4 Bagian 18

3.5.2.5 Nomor 5

```

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

```

```

In [29]: from sklearn.metrics import confusion_matrix #Mengimport Confusion Matrix
...: pred_labels = clf.predict(df_test_att) #Membuat variable pred_labels dari data testing
...: cm = confusion_matrix(df_test_label, pred_labels) #cm sebagai variabel data label

```

Gambar 3.145 Hasil 5 Bagian 1

```

1 cm #Memunculkan data label berbentuk array

```

```

array([[ 6,  2,  0, ...,  0,  1,  0],
       [ 0, 15,  0, ...,  0,  0,  0],
       [ 5,  1,  7, ...,  0,  0,  0],
       ...,
       [ 0,  0,  1, ...,  2,  0,  0],
       [ 0,  0,  0, ...,  0,  9,  0],
       [ 0,  0,  0, ...,  0,  0, 16]], dtype=int64)

```

Gambar 3.146 Hasil 5 Bagian 2

```

1 import matplotlib.pyplot as plt #Mengimport library matplotlib
    sebagai plt
2 import itertools #Mengimport library itertools
3 def plot_confusion_matrix(cm, classes,
                           normalize=False,
                           title='Confusion matrix',
                           cmap=plt.cm.Blues): #Membuat fungsi
    dengan ketentuan data yang ada pada cm
7
8     if normalize:
9         cm = cm.astype('float') / cm.sum(axis=1)[:, np.newaxis]
10        print("Normalized confusion matrix") #Jika normalisasi
11        sebagai ketentuan yang ada maka print normalized confusion
12        matrix
13    else:
14        print('Confusion matrix, without normalization') #Jika
15        tidak memenuhi kondisi if maka pring else
16
17    print(cm) #Print data cm
18
19    plt.imshow(cm, interpolation='nearest', cmap=cmap) #plt
20    sebagai fungsi untuk membuat plot

```

```

17 plt.title(title) #Membuat title pada plot
18 #plt.colorbar()
19 tick_marks = np.arange(len(classes)) #Membuat marks pada plot
20 plt.xticks(tick_marks, classes, rotation=90) #Membuat ticks
21 pada x
22 plt.yticks(tick_marks, classes) #Membuat ticks pada y
23
24 fmt = '.2f' if normalize else 'd' #fmt sebagai normalisasi
25 thresh = cm.max() / 2. #Variable thresh menambil data max
26 pada cm kemudian dibagi 2
27
28 plt.tight_layout() #Mengatur layout pada plot
29 plt.ylabel('True label') #Menambahkan nama label pada sumbu y
30 plt.xlabel('Predicted label') #Menambahkan nama label pada
31 sumbu x

```

```

In [32]: import matplotlib.pyplot as plt #Mengimport library matplotlib sebagai plt
...: import itertools #Mengimport library itertools
...: def plot_confusion_matrix(cm, classes,
...:                          normalize=False,
...:                          title='Confusion matrix',
...:                          cmap=plt.cm.Blues): #Membuat fungsi dengan ketentuan data yang ada pada
cm
...:
...:     if normalize:
...:         cm = cm.astype('float') / cm.sum(axis=1)[:, np.newaxis]
...:         print("Normalized confusion matrix") #Jika normalisasi sebagai ketentuan yang ada maka
print normalized confusion matrix
...:     else:
...:         print('Confusion matrix, without normalization') #Jika tidak memenuhi kondisi if maka
print else
...:
...:     print(cm) #Print data cm
...:
...:     plt.imshow(cm, interpolation='nearest', cmap=cmap) #plt sebagai fungsi untuk membuat plot
...:     plt.title(title) #Membuat title pada plot
...:     #plt.colorbar()
...:     tick_marks = np.arange(len(classes)) #Membuat marks pada plot
...:     plt.xticks(tick_marks, classes, rotation=90) #Membuat ticks pada x
...:     plt.yticks(tick_marks, classes) #Membuat ticks pada y
...:
...:     fmt = '.2f' if normalize else 'd' #fmt sebagai normalisasi
...:     thresh = cm.max() / 2. #Variable thresh menambil data max pada cm kemudian dibagi 2
...:
...:     plt.tight_layout() #Mengatur layout pada plot
...:     plt.ylabel('True label') #Menambahkan nama label pada sumbu y
...:     plt.xlabel('Predicted label') #Menambahkan nama label pada sumbu x

```

Gambar 3.147 Hasil 5 Bagian 3

```

1 birds = pd.read_csv("E:/Kecerdasan Buatan/CUB_200_2011/classes .
txt",
2                                     sep='\s+', header=None, usecols=[1], names=[ 'birdname']) #membaca csv dengan ketentuan nama birdname
3 birds = birds['birdname'] #nama birds dengan ketentuan birdname
4 birds #Menampilkan data birds

```

```

187          188.Pileated_Woodpecker
188          189.Red_bellied_Woodpecker
189          190.Red_cockaded_Woodpecker
190          191.Red_headed_Woodpecker
191          192.Downy_Woodpecker
192          193.Bewick_Wren
193          194.Cactus_Wren
194          195.Carolina_Wren
195          196.House_Wren
196          197.Marsh_Wren
197          198.Rock_Wren
198          199.Winter_Wren
199          200.Common_Yellowthroat
Name: birdname, Length: 200, dtype: object

```

Gambar 3.148 Hasil 5 Bagian 4

```

1 import numpy as np #Mengimport library numpy sebagai np
2 np.set_printoptions(precision=2) #np sebagai variabel yang
   membuat set precision=2
3 plt.figure(figsize=(60,60), dpi=300) #Plot sebagai figure dengan
   ketentuan sizw 60,60 dan dpi 300
4 plot_confusion_matrix(cm, classes=birds, normalize=True) #Data cm
   dan clas birds dibuat sebagai plot
5 plt.show()
6 #plt.savefig('hasil.png')

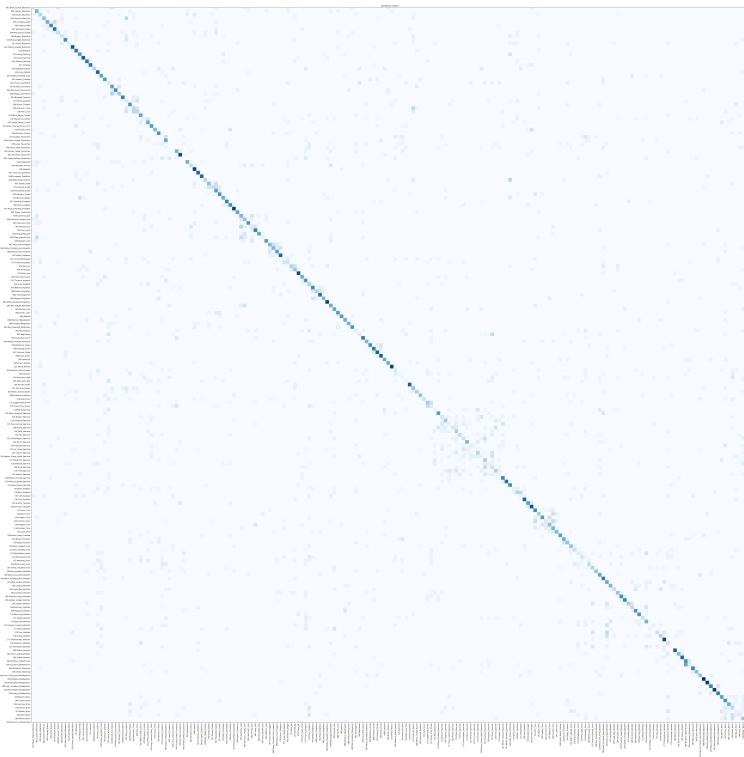
```

```

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

```

Gambar 3.149 Hasil 5 Bagian 5

**Gambar 3.150** Plot Hasil 5 Bagian 5

3.5.2.6 Nomor 6

```

1 from sklearn import tree #Mengimport library tree
2 clftree = tree.DecisionTreeClassifier() #clftree sebagai variabel
   untuk decision tree
3 clftree.fit(df_train_att, df_train_label) #Mengatur data training
4 clftree.score(df_test_att, df_test_label) #Mengatur data testing

In [26]: from sklearn import tree #Mengimport Library tree
...: clftree = tree.DecisionTreeClassifier() #clftree sebagai variabel untuk decision tree
...: clftree.fit(df_train_att, df_train_label) #Mengatur data training
...: clftree.score(df_test_att, df_test_label) #Mengatur data testing
Out[26]: 0.264519535374868

```

Gambar 3.151 Hasil 6 Bagian 1

```

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

```

```
In [27]: from sklearn import svm #Mengimport library svm
...: clfsvm = svm.SVC() #clfsvm sebagai variabel untuk mengatur fungsi SVC
...: clfsvm.fit(df_train_att, df_train_label) #Mengatur data training
...: clfsvm.score(df_test_att, df_test_label) #Mengatur data testing
C:\Users\Panda23\Anaconda3\lib\site-packages\sklearn\svm\base.py:193: FutureWarning: The default value of gamma will change from 'auto' to 'scale' in version 0.22 to account better for unscaled features. Set gamma explicitly to 'auto' or 'scale' to avoid this warning.
"avoid this warning.", FutureWarning)

Out[27]: 0.2819429778247096C:\Users\Panda23\Anaconda3\lib\site-packages\sklearn\svm\base.py:193:
FutureWarning: The default value of gamma will change from 'auto' to 'scale' in version 0.22 to account
better for unscaled features. Set gamma explicitly to 'auto' or 'scale' to avoid this warning.
"avoid this warning.", FutureWarning)
```

Gambar 3.152 Hasil 6 Bagian 2

3.5.2.7 Nomor 7

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

```
In [28]: from sklearn.model_selection import cross_val_score #Mengimport cross_val_score
...: scores = cross_val_score(clf, df_train_att, df_train_label, cv=5) #Membuat variable scores sebagai
variabel prediksi dari data training
...: print("Accuracy: %0.2f (+/- %0.2f)" % (scores.mean(), scores.std() * 2)) #Print data scores dengan
ketentuan akurasi
Out[28]: 0.2819429778247096Accuracy: 0.44 (+/- 0.03)
```

Gambar 3.153 Hasil 7 Bagian 1

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

```
In [30]: scorestree = cross_val_score(clftree, df_train_att, df_train_label, cv=5) #Membuat variable
prediksi menggunakan scores dan metode tree
...: print("Accuracy: %0.2f (+/- %0.2f)" % (scorestree.mean(), scorestree.std() * 2)) #Menampilkan
dengan ketentuan yang ada
Accuracy: 0.25 (+/- 0.02)
```

Gambar 3.154 Hasil 7 Bagian 2

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

```
In [31]: scoressvm = cross_val_score(clfsvm, df_train_att, df_train_label, cv=5) #Membuat variable data
training
...: print("Accuracy: %.2f (+/- %.2f)" % (scoressvm.mean(), scoressvm.std() * 2)) #Menampilkan data
testing dan output akurasi
C:\Users\Panda23\Anaconda3\lib\site-packages\sklearn\svm\base.py:193: FutureWarning: The default value of
gamma will change from 'auto' to 'scale' in version 0.22 to account better for unscaled features. Set gamma
explicitly to 'auto' or 'scale' to avoid this warning.
"avoid this warning.", FutureWarning)
C:\Users\Panda23\Anaconda3\lib\site-packages\sklearn\svm\base.py:193: FutureWarning: The default value of
gamma will change from 'auto' to 'scale' in version 0.22 to account better for unscaled features. Set gamma
explicitly to 'auto' or 'scale' to avoid this warning.
"avoid this warning.", FutureWarning)
C:\Users\Panda23\Anaconda3\lib\site-packages\sklearn\svm\base.py:193: FutureWarning: The default value of
gamma will change from 'auto' to 'scale' in version 0.22 to account better for unscaled features. Set gamma
explicitly to 'auto' or 'scale' to avoid this warning.
"avoid this warning.", FutureWarning)
C:\Users\Panda23\Anaconda3\lib\site-packages\sklearn\svm\base.py:193: FutureWarning: The default value of
gamma will change from 'auto' to 'scale' in version 0.22 to account better for unscaled features. Set gamma
explicitly to 'auto' or 'scale' to avoid this warning.
"avoid this warning.", FutureWarning)
C:\Users\Panda23\Anaconda3\lib\site-packages\sklearn\svm\base.py:193: FutureWarning: The default value of
gamma will change from 'auto' to 'scale' in version 0.22 to account better for unscaled features. Set gamma
explicitly to 'auto' or 'scale' to avoid this warning.
"avoid this warning.", FutureWarning)
Accuracy: 0.26 (+/- 0.02)
```

Gambar 3.155 Hasil 7 Bagian 3

3.5.2.8 Nomor 8

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

```

....      ..._param[4] = scores.mean() + scores.std() * 2
....      rf_params[i,3] = scores.std() * 2 #index 3
....      i += 1 #Dengan ketentuan i += 1
....      print("Max features: %d, num estimators: %d, acc
n_estimators, scores.mean(), scores.std() * 2))
....      #Print hasil pengulangan yang sudah ditentukan
Max features: 5, num estimators: 10, accuracy: 0.26 (+/- 0.03)
Max features: 5, num estimators: 30, accuracy: 0.36 (+/- 0.03)
Max features: 5, num estimators: 50, accuracy: 0.39 (+/- 0.03)
Max features: 5, num estimators: 70, accuracy: 0.41 (+/- 0.02)
Max features: 5, num estimators: 90, accuracy: 0.42 (+/- 0.02)
Max features: 5, num estimators: 110, accuracy: 0.43 (+/- 0.03)
Max features: 5, num estimators: 130, accuracy: 0.43 (+/- 0.02)
Max features: 5, num estimators: 150, accuracy: 0.44 (+/- 0.02)
Max features: 5, num estimators: 170, accuracy: 0.44 (+/- 0.03)
Max features: 5, num estimators: 190, accuracy: 0.45 (+/- 0.01)
Max features: 10, num estimators: 10, accuracy: 0.29 (+/- 0.02)
Max features: 10, num estimators: 30, accuracy: 0.38 (+/- 0.02)
Max features: 10, num estimators: 50, accuracy: 0.41 (+/- 0.03)
Max features: 10, num estimators: 70, accuracy: 0.43 (+/- 0.02)
Max features: 10, num estimators: 90, accuracy: 0.43 (+/- 0.02)
Max features: 10, num estimators: 110, accuracy: 0.44 (+/- 0.02)
Max features: 10, num estimators: 130, accuracy: 0.45 (+/- 0.03)
Max features: 10, num estimators: 150, accuracy: 0.45 (+/- 0.03)
Max features: 10, num estimators: 170, accuracy: 0.45 (+/- 0.02)
Max features: 10, num estimators: 190, accuracy: 0.45 (+/- 0.03)
Max features: 15, num estimators: 10, accuracy: 0.31 (+/- 0.03)
Max features: 15, num estimators: 30, accuracy: 0.40 (+/- 0.03)
Max features: 15, num estimators: 50, accuracy: 0.42 (+/- 0.03)
Max features: 15, num estimators: 70, accuracy: 0.43 (+/- 0.02)
Max features: 15, num estimators: 90, accuracy: 0.43 (+/- 0.03)
Max features: 15, num estimators: 110, accuracy: 0.44 (+/- 0.02)
Max features: 15, num estimators: 130, accuracy: 0.44 (+/- 0.02)
Max features: 15, num estimators: 150, accuracy: 0.45 (+/- 0.02)
Max features: 15, num estimators: 170, accuracy: 0.45 (+/- 0.02)
Max features: 15, num estimators: 190, accuracy: 0.45 (+/- 0.02)

```

Gambar 3.156 Hasil 8 Bagian 1

```

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

```

Gambar 3.157 Hasil 8 Bagian 1 Akhir kode

```

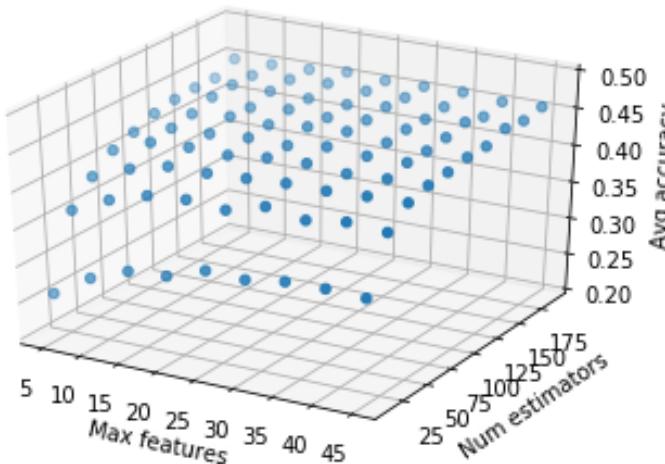
1 import matplotlib.pyplot as plt #Mengimport library matplotlib
sebagai plt

```

```

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

```



Gambar 3.158 Hasil 8 Bagian 2

3.5.3 Penanganan Error

1. ScreenShoot Error

```

File "<ipython-input-2-442c23aa66173>", line 1, in <module>
    import pandas as pd # Melakukan import library pandas sebagai pd
ModuleNotFoundError: No module named 'pandas'

```

Gambar 3.159 NoModuleNotFoundError

2. Tuliskan Kode Error dan Jenis Error

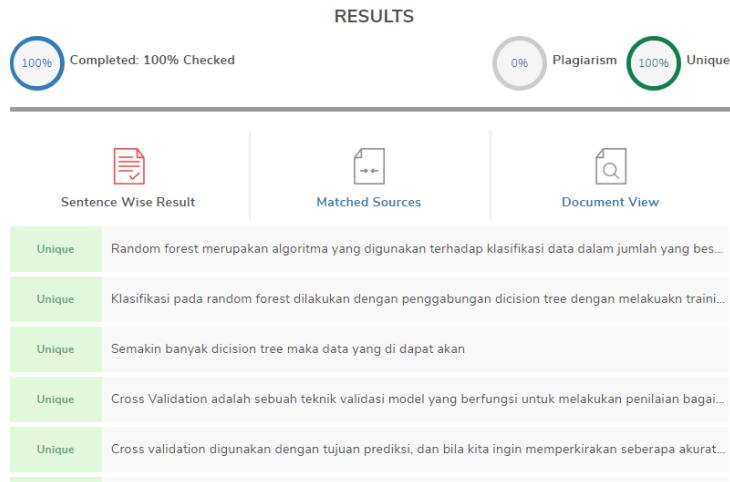
- NoModuleNotFoundError

3. Cara Penangan Error

- NoModuleNotFoundError

Penulisan nama modul harus sesuai atau menginstall module yang diinginkan

3.5.4 Bukti Tidak Plagiat



Gambar 3.160 Bukti Tidak Plagiat

3.5.5 Link Youtube:

<https://www.youtube.com/watch?v=f9v0KKU8JJ8>

3.6 1174079 -Chandra Kirana Poetra

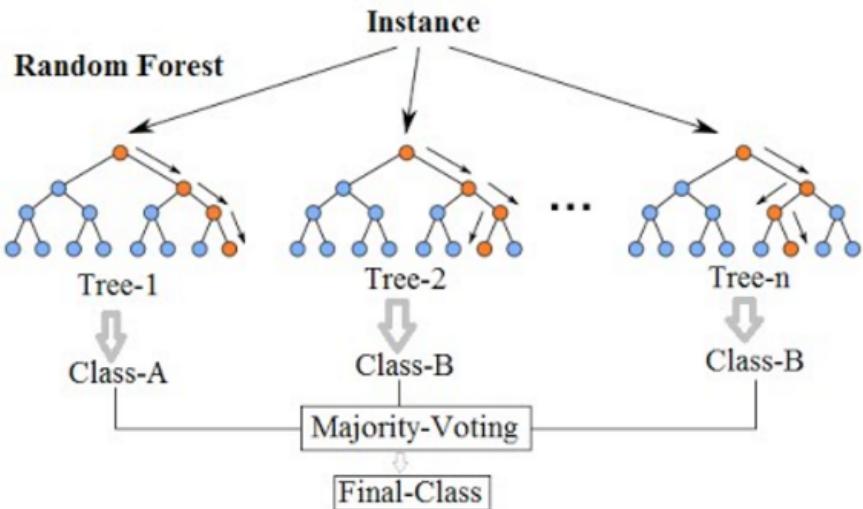
Chapter 3 - Prediksi dengan Random Forest

3.6.1 Teori

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

Random forest merupakan algoritma klasifikasi yang meliputi banyak decisions trees. random forest menggunakan bagging (salah satu algoritma machine learning) serta fitur pengacakan(randomness) ketika membangun setiap satu tree untuk mencoba membuat suatu hutan yang berisi pohon yang tidak berkorelasi

Random Forest Simplified



Gambar 3.161 contoh random forest

3.6.1.2 Jelaskan cara membaca dataset kasus dan artikan makna setiap file dan isi field masing masing file.

Dataset adalah suatu koleksi dari banyak data, biasanya dalam bentuk tabel dan penggambarannya kurang lebih mirip seperti table yang ada di database.

1. cara membaca dataset

```

1 # -*- coding: utf-8 -*-
2 """
3 Created on Wed Mar 18 19:42:42 2020
4
5 @author: Chandra
6 """
7
8 import pandas as pd
9 dataset = pd.read_csv('F:/ Poltekpos/D4 TI 3C/Semester 6/
10   Kecerdasan Buatan/Github/Upload 18 Maret 2020 github tugas
11   3/src/1174079/3/data.csv', sep=',')
12 dataset.head()
  
```

- import library panda dan namai sebagai pd agar memudahkan pemanggilan

- buat variable dataset dan panggil perintah untuk membaca file yang dimaksud
- berikut hasilnya

Index	Men Pertanian (20)	Unnamed: 1	Unnamed: 2	Unnamed: 3	Unnamed: 4	Unnamed: 5
0	No.	Komoditas	Tahun	nan	nan	Satuan
1	nan	nan	2016	2017	2018	nan
2	A	Pertanian Tanaman Pang...	nan	nan	nan	nan
3	I	Luas Panen Padi dan Pal...	nan	nan	nan	nan
4	1	Padi Sawah	675,00	469,00	490,00	Hektar
5	2	Padi Gogo	4,00	0	0	Hektar
6	3	Jagung panen Tua	224,00	161,00	299,00	Hektar
7	4	Jagung Panen Muda	232,00	433,00	349,00	Hektar
8	5	Kacang Tanah	60,00	39,00	1,90	Hektar
9	6	Ubi Kayu	62,00	48,00	23,00	Hektar
10	7	Ubi Jalar	94,00	57,00	4,90	Hektar
11	B	Pertanian Hortikultura	nan	nan	nan	nan
12	I	Luas Panen Tanaman Sayur...	nan	nan	nan	nan
13	1	Bawang Merah	284	373	379	Hektar
14	2	Bawang Putih	41	31	22	Hektar
15	3	Bawang Daun	281	285	373	Hektar
16	4	Kentang	466	490	475	Hektar
17	5	Kubis	410	404	257	Hektar
18	6	Kembang Kol	398	391	361	Hektar
19	7	Petsai/Sawi	311	314	342	Hektar
20	8	Wortel	371	402	505	Hektar
21	9	Kacang Merah	30	24	25	Hektar
		Kacang				

Gambar 3.162 contoh dataset

3.6.1.3 Jelaskan apa itu cross validation

Cross-validation (CV) adalah Merupakan metode statistic untuk meng-evaluasi dan membandingkan akurasi Learning algorithm dengan cara membagi dataset menjadi 2 bagian: Satu bagian digunakan untuk training model, Bagian yang lain untuk mem-validasi model Suatu dataset akan dibagi sesuai dengan banyaknya k, dan akan di test bergantian hingga seluruh bagian terpenuhi.

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

Score dari masing masing itu merupakan hasil dari tingkat keakuratan predikasi yang digunakan, jika menggunakan random forest anda mendapatkan tingkat keakuratan mencapai 44%, decision tree sekitar 27% dan svm 29%.

Kelas	Terklasifikasi Positif	Terklasifikasi Negatif
Positif	TP (True Positive)	FN (False Negative)
Negatif	FP (False Positive)	TN (True Negative)

Gambar 3.163 Contoh confusion matrix

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

1. TP adalah True Positive, jumlah data positif terdeteksi benar
2. TN adalah True Negative, jumlah data negatif terdeteksi benar.
3. FN adalah False Negative, jumlah data negatif terdeteksi salah.
4. FP adalah False Positive, jumlah data positif terdeteksi salah

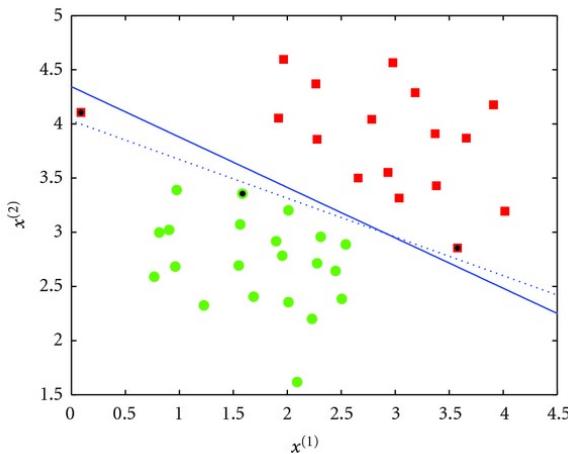
		Actual values	
		Apple	Orange
Predicted values	Apple	5 True positives	2 False positives
	Orange	3 False negatives	3 True negatives

Gambar 3.164 Contoh confusion matrix

Penjelasan dari gambar diatas adalah mesin mendeteksi 5 apel sebagai 5 apel dan 2 apel sisanya terdeteksi sebagai jeruk, lalu 3 jeruk terdeteksi sebagai apel dan 3 jeruk terdeteksi sebagai jeruk.

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

Voting di random forest kurang lebih sama seperti ketika kita ingin bertanya atau survey apakah suatu makanan masuk ke kategori enak atau tidak, jadi bisa dibilang voting itu adalah suara atau pendapat, yang akan diklasifikasikan.



Gambar 3.165 contoh random forest tentang kualitas barang yang semakin mahal semakin tahan lama

3.6.2 Praktek

3.6.2.1 Nomor 1

```

1 # In [0]
2 import pandas as pd # import library
3 dataset = pd.read_csv('F:/Poltekpos/D4 TI 3C/Semester 6/
   Kecerdasan Buatan/Github/Upload 18 Maret 2020 github tugas 3/
   src/1174079/3/data.csv', sep=',')
4 # buat variable berisi file csv
5 dataset.head() # membaca data csv

```

	Luas Panen Pertanian (2016-2018)			... Unnamed: 5
0	No.	...	Satuan	
1	NaN	...	NaN	
2	A	...	NaN	
3	I	...	NaN	
4	1	...	Hektar	

Gambar 3.166 Membuat Aplikasi pakai pandas

3.6.2.2 Nomor 2

```

1 import numpy as np # import numpy
2 np.arange(1, 14, 4) # mengatur value awal dan akhir dilompati
   dengan beda 4

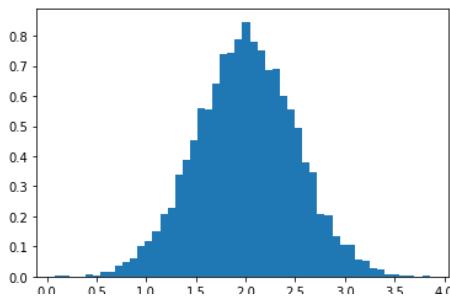
```

```
In [2]: import numpy as np # imp
.... np.arange(1, 14, 4) # me
ditampati dengan beda 4
Out[2]: array([ 1,  5,  9, 13])
```

Gambar 3.167 Membuat Aplikasi pakai numpy

3.6.2.3 Nomor 3

```
1 import numpy as np #import library
2 import matplotlib.pyplot as plt #import library
3 mu, sigma = 2, 0.5 # Build a vector of 10000 normal deviates with
        variance 0.5^2 and mean 2
4 v = np.random.normal(mu,sigma,10000) #Mengisi histogram dengan
        data
5 # Plot a normalized histogram with 50 bins
6 plt.hist(v, bins=50, density=1)          # matplotlib version (plot)
7 plt.show() # tampilkan
```



Gambar 3.168 Membuat Aplikasi pakai matplotlib

3.6.2.4 Nomor 4

- Membaca dataset dengan variable budi. variable ini digunakan untuk mengisi data

```
1 import pandas as pd #Melakukan import library numpy menjadi
pd
2
3 budi = pd.read_csv("D:/Pictures/image_attribute_labels.txt",
4                     sep='\s+', header=None, error_bad_lines=
False, warn_bad_lines=False,
5                     usecols=[0,1,2], names=['imgid', 'attid',
, 'present']) #Membuat variable budi untuk membaca file
csv dari dataset, dengan ketentuan yang ada.
```

```
In [9]: import pandas as pd #Melakukan import library numpy menjadi pd
....:
....: budi = pd.read_csv("D:/Pictures/
image_attribute_labels.txt",
....: sep='\s+', header=None,
error_bad_lines=False, warn_bad_lines=False,
....: usecols=[0,1,2], names=['imgid',
'attid', 'present']) #Membuat variable budi untuk membaca file
csv dari dataset, dengan ketentuan yang ada.
```

Gambar 3.169 Hasil 4 Bagian 1

- Menampilkan data teratas dari variable budi

```
1 budi.head() #Menampilkan data yang sudah dibaca tadi tapi
cuman data paling atas
```

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

Gambar 3.170 Hasil 4 Bagian 2

- menampilkan jumlah kolumn dan baris dari variable budi

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

In [11]: budi.shape #	nya
Out[11]:	(3677856, 3)

Gambar 3.171 Hasil 4 Bagian 3

- Mengubah kolumn jadi baris dengan fungsi pivot yang disimpan pada variable budi2 yang datanya berasal dari variable budi sebelumnya.

```
1 budi2 = budi.pivot(index='imgid', columns='attid', values='
present') #Membuat sebuah variabel baru dari fungsi budi ,
dengan mengganti index menjadi kolom dan kolom menjadi
index
```

```
In [12]: budi2 = budi.pivot(index='imgid', columns='attid',
values='present') #Membuat sebuah variabel baru dari fungsi
budi, dengan mengganti index menjadi kolom dan kolom menjadi
index
```

Gambar 3.172 Hasil 4 Bagian 4

- Menampilkan data teratas dari budi2

```
1 budi2.head() #Menampilkan data yang sudah dibaca tadi tapi
cuman data paling atas
```

```
Out[13]:
attid 1 2 3 4 5 6 7 ... 306 307 308
309 310 311 312
imgid
1 0 0 0 0 1 0 0 ... 0 0 1
0 0 0 0 0 0 0 0 ... 0 0 0
2 0 0 0 0 0 0 0 ... 0 0 0
0 0 0 0 0 0 0 0 ... 0 0 1
3 0 0 0 0 1 0 0 ... 0 0 0
0 0 1 0 0 0 0 0 ... 1 0 0
4 0 0 0 0 0 1 0 0 ... 1 0 0
1 0 0 0 0 0 0 1 0 0 0 ...
5 0 0 0 0 0 0 0 0 0 0
0 0 0 0
```

[5 rows x 312 columns]

Gambar 3.173 Hasil 4 Bagian 5

- menampilkan jumlah kolumn dan baris variable budi2

```
1 budi2.shape #Menampilkan jumlah seluruh data , kolom-nya
```

Out[25]: (11788, 312)

Gambar 3.174 Hasil 4 Bagian 6

- membaca data csv dan membuat index baru bernama imgid

```
1 imglabels = pd.read_csv("D:/Pictures/image_class_labels.txt",
2 sep=',', header=None, names=['imgid',
3 'label']) #Membaca data dimasukkan ke variable imglabels
4 imglabels = imglabels.set_index('imgid') #Variable imglabels
dan set index (imgid)
```

```
In [26]: imglabels = pd.read_csv("D:/Pictures/
image_class_labels.txt",
...
sep=',', header=None,
names=['imgid', 'label']) #Membaca data dimasukkan ke variable
imglabels
```

```
...
...: imglabels = imglabels.set_index('imgid') #Variable
imglabels dan set index (imgid)
```

```
In [27]: imglabels = pd.read_csv("D:/Pictures/
image_class_labels.txt",
...
sep=',', header=None,
names=['imgid', 'label']) #Membaca data dimasukkan ke variable
imglabels
```

```
...
...: imglabels = imglabels.set_index('imgid') #Variable
imglabels dan set index (imgid)
```

Gambar 3.175 Hasil 4 Bagian 7

- Menampilkan data teratas

```
1 imglabels.head() #Menampilkan data yang sudah dibaca tadi  
    tapi cuman data paling atas
```

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

Gambar 3.176 Hasil 4 Bagian 8

- mengeluarkan output jumlah kolumn dan baris

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

Out[31]: (11788, 1)

Gambar 3.177 Hasil 4 Bagian 9

- Menggabungkan variable budi2 dan imglabels karena isi yang mirip sehingga bisa dikategorikan sebagai supervised learning.

```
1 df = budi2.join(imglabels) #Varibel df dimasukkan fungsi join  
    dari data budi2 ke variabel imglabels  
2 df = df.sample(frac=1) #Varibel df sebagai sample dengan  
    ketentuan frac=1
```

```
In [32]: df = budi2.join(imglabels) #1  
fungsi join dari data budi2 ke variabel  
....: df = df.sample(frac=1) #Varibel  
dengan ketentuan frac=1
```

Gambar 3.178 Hasil 4 Bagian 10

- membuat column

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

```
In [33]: df_att = df.iloc[:, :312] #Membuat kolom dengan
      ketentuan 312
      ...: df_label = df.iloc[:, 312:] #Membuat kolom dengan
      ketentuan 312
```

Gambar 3.179 Hasil 4 Bagian 11

- Mengecek data teratas variable df_att.

```
1 df_att.head() #Menampilkan data yang sudah dibaca tadi tapi
   cuman data paling atas
```

	1	2	3	4	5	6	7	...	306	307	308
309	310	311	312								
imgid	0	1	0	0	0	0	0	...	0	0	1
5911	0	0	1	0	0	0	0	...	0	0	0
68	0	0	0	0	0	1	0	0	...	1	0
1	0	0	0	0	0	0	0	1	...	0	0
10028	0	0	0	0	0	0	0	1	...	0	0
5002	0	1	0	0	0	0	0	0	...	0	0
0	0	0	1	0	0	0	0	0	...	0	1
10530	0	0	0	0	0	0	0	1	...	0	0
0	0	0	1	0	0	0	0	0	...	0	0

15 rows × 312 columns

Gambar 3.180 Hasil 4 Bagian 12

- Mengeceki data teratas dari df_label.

```
1 df_label.head() #Menampilkan data yang sudah dibaca tadi tapi
   cuman data paling atas
```

```
In [36]: df_label.head()
tadi tapi cuman data pal
Out[36]:
label
imgid
5911      101
68          2
10028     171
5002      86
10530     179
```

Gambar 3.181 Hasil 4 Bagian 13

- Membagi 8000 baris awal sebagai data training dan yang lainnya sebagai data testing.

```

1 df_train_att = df_att[:8000] #Data akan dibagi dari 8000 row
   pertama menjadi data training dan sisanya adalah data
   testing
2 df_train_label = df_label[:8000] #Data akan dibagi dari 8000
   row pertama menjadi data training dan sisanya adalah data
   testing
3 df_test_att = df_att[8000:] #Berbalik dari sebelumnya data
   akan dibagi mulai dari 8000 row pertama menjadi data
   training dan sisanya adalah data testing
4 df_test_label = df_label[8000:] #Berbalik dari sebelumnya
   data akan dibagi mulai dari 8000 row pertama menjadi data
   training dan sisanya adalah data testing
5
6 df_train_label = df_train_label['label'] #Menambahkan label
7 df_test_label = df_test_label['label'] #Menambahkan label

```

```

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

```

Gambar 3.182 Hasil 4 Bagian 14

- memanggil class RandomForestClassifier. dengan maks kolumn 50

```

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

```

```

In [38]: from sklearn.ensemble import RandomForestClassifier
#Import fungsi randomforestclassifier
...: clf = RandomForestClassifier(max_features=50,
random_state=0, n_estimators=100) #clf sebagai variabel untuk
klasifikasi random forest

```

Gambar 3.183 Hasil 4 Bagian 15

- output hasil prediksi dari Random Forest.

```

1 clf.fit(df_train_att, df_train_label) #Variabne clf untuk
   fit yaitu menjadi data training

```

```
In [39]: clf.fit(dt_train_att, dt_train_label) #Variabne untuk fit yaitu menjadi data training
Out[39]:
RandomForestClassifier(bootstrap=True, class_weight=None,
criterion='gini',
max_depth=None, max_features=50,
max_leaf_nodes=None, min_impurity_decrease=0.0,
min_impuity_split=None, min_samples_leaf=1,
min_samples_split=2, min_weight_fraction_leaf=0.0,
n_estimators=100, n_jobs=None, oob_score=False,
random_state=0, verbose=0,
warm_start=False)
```

Gambar 3.184 Hasil 4 Bagian 16

- Memerlihatkan score akurasi.

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

```
In [40]: print(clf.predict(df_train_att.head()))
yang di sudah prediksi dari training tetapi hanya
data paling atas
[101  2 171  86 179]
```

Gambar 3.185 Hasil 4 Bagian 17

- Menampilkan tingkat akurasi

```
1 clf.score(df_test_att, df_test_label) #Memunculkan clf
    sebagai testing yang sudah di training tadi
```

```
In [41]: clf.score(df_test_att, df_test_label)
clf sebagai testing yang sudah di training tadi
Out[41]: 0.45248152059134106
```

Gambar 3.186 Hasil 4 Bagian 18

3.6.2.5 Nomor 5

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

```
In [42]: from sklearn.metrics import confusion_matrix
#Mengimport Confusion Matrix
...; pred_labels = clf.predict(df_test_att) #Membuat
variable pred_labels dari data testing
...; cm = confusion_matrix(df_test_label, pred_labels)
sebagai variabel data label
```

Gambar 3.187 Hasil 5 Bagian 1

```
1 cm #Memunculkan data label berbentuk array
```

```
In [43]: cm #Memunculkan data label berbentuk array
Out[43]:
array([[ 2,  3,  3, ...,  0,  0,  0],
       [ 0, 11,  0, ...,  0,  1,  0],
       [ 0,  0, 12, ...,  0,  0,  0],
       ...,
       [ 0,  0,  0, ...,  2,  0,  0],
       [ 0,  0,  0, ...,  0, 10,  0],
       [ 0,  0,  0, ...,  0,  0, 12]], dtype=int64)
...
```

Gambar 3.188 Hasil 5 Bagian 2

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

```
In [44]: import matplotlib.pyplot as plt #Mengimport library
matplotlib sebagai plt
.... import itertools #Mengimport library itertools
.... def plot_confusion_matrix(cm, classes,
....                          normalize=False,
....                          title='Confusion matrix',
....                          cmap=plt.cm.Blues):
#Membuat fungsi dengan ketentuan data yang ada pada cm
.... if normalize:
....     cm = cm.astype('float') / cm.sum(axis=1)[:, np.newaxis]
....     print("Normalized confusion matrix") #Jika
normalisasi sebagai ketentuan yang ada maka print normalized
confusion matrix
.... else:
....     print('Confusion matrix, without
normalization') #Jika tidak memenuhi kondisi tf maka print
else:
....     print(cm) #Print data cm
.... plt.imshow(cm, interpolation='nearest',
cmap=cmap) #plt sebagai fungsi untuk membuat plot
.... plt.title(title) #Membuat title pada plot
.... plt.colorbar()
.... tick_marks = np.arange(len(classes)) #Membuat
marks pada plot
.... plt.xticks(tick_marks, classes, rotation=90)
#Membuat ticks pada x
```

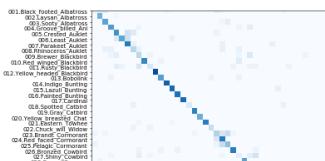
Gambar 3.189 Hasil 5 Bagian 3

```
1 birds = pd.read_csv("D:/ Pictures/classes.txt",
2                      sep='\s+', header=None, usecols=[1], names=[],
3                      birdname']) #membaca csv dengan ketentuan nama birdname
4 birds = birds['birdname'] #nama birds dengan ketentuan birdname
5 birds #Menampilkan data birds
```

```
Out[45]:
0      001.Black_footed_Albatross
1      002.Laysan_Albatross
2      003.Sooty_Albatross
3      004.Groove_billed_Ani
4      005.Crested_Auklet
...
195          196.House_Wren
196          197.Marsh_Wren
197          198.Rock_Wren
198          199.Winter_Wren
199          200.Common_Yellowthroat
Name: birdname, Length: 200, dtype: object
```

Gambar 3.190 Hasil 5 Bagian 4

```
1 import numpy as np #Mengimport library numpy sebagai np
2 np.set_printoptions(precision=2) #np sebagai variabel yang
   membuat set precision=2
3 plt.figure(figsize=(30,30), dpi=50) #Plot sebagai figure dengan
   ketentuan sizew 60,60 dan dpi 300
4 plot_confusion_matrix(cm, classes=birds, normalize=True) #Data cm
   dan clas birds dibuat sebagai plot
5 #plt.show
6 #plt.savefig ('hasil.png') #
```



Gambar 3.191 Hasil 5 Bagian 5

3.6.2.6 Nomor 6

```

1 from sklearn import tree #Mengimport library tree
2 clftree = tree.DecisionTreeClassifier() #clftree sebagai variabel
   untuk decision tree
3 clftree.fit(df_train_att, df_train_label) #Mengatur data training
4 clftree.score(df_test_att, df_test_label) #Mengatur data testing

```

| Out[49]: 0.2732312565997888

Gambar 3.192 Hasil 6 Bagian 1

```

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

```

Out[50]: 0.28880675818373813

Gambar 3.193 Hasil 6 Bagian 2

3.6.2.7 Nomor 7

```

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

```

```

In [51]: from sklearn.model_selection import cross_val_score
#Mengimport cross_val_score
...: scores = cross_val_score(clf, df_train_att,
df_train_label, cv=5) #Membuat variable scores sebagai
variabel prediksi dari data training
...: print("Accuracy: %.2f (+/- %.2f)" % (scores.mean(),
scores.std() * 2)) #Print data scores dengan ketentuan akurasi
Accuracy: 0.44 (+/- 0.01)

```

Gambar 3.194 Hasil 7 Bagian 1

```

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

```

```

In [52]: scorestree = cross_val_score(clftree, df_train_att,
df_train_label, cv=5) #Membuat variable prediksi menggunakan
scores dan metode tree
...: print("Accuracy: %0.2f (+/- %0.2f)" %
(scorestree.mean(), scorestree.std() * 2)) #Menampilkan dengan
ketentuan yang ada
Accuracy: 0.26 (+/- 0.02)

```

Gambar 3.195 Hasil 7 Bagian 2

```

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

```

| Accuracy: 0.27 (+/- 0.01)

Gambar 3.196 Hasil 7 Bagian 3

3.6.2.8 Nomor 8

```

1 max_features_opts = range(5, 50, 5) #Variable max_features_opts
    sebagai variabel untuk membuat range 5,50,5
2 n_estimators_opts = range(10, 200, 20) #Variablen_estimators_opts
    sebagai variabel untuk membuat range 10,200,20
3 rf_params = np.empty((len(max_features_opts)*len(
    n_estimators_opts),4), float) #Variablerf_params sebagai
    variabel untuk menjumlahkan yang sudah di tentukan sebelumnya
4 i = 0
5 for max_features in max_features_opts: #Perulangan
6     for n_estimators in n_estimators_opts: #Perulangan
7         clf = RandomForestClassifier(max_features=max_features ,
n_estimators=n_estimators) #Menampilkan variabel csf
8         scores = cross_val_score(clf , df_train_att ,
df_train_label , cv=5) #Variable scores sebagai variabel
    training
9         rf_params[i,0] = max_features #index 0
10        rf_params[i,1] = n_estimators #index 1
11        rf_params[i,2] = scores.mean() #index 2
12        rf_params[i,3] = scores.std() * 2 #index 3
13        i += 1 #Dengan ketentuan i += 1
14        print("Max features: %d, num estimators: %d, accuracy:
    %0.2f (+/- %0.2f)" % (max_features , n_estimators ,
    scores.mean(), scores.std() * 2))
15        #Print hasil pengulangan yang sudah ditentukan

```

```

Max features: 5, num estimators: 10, accuracy: 0.27 (+/- 0.01)
Max features: 5, num estimators: 30, accuracy: 0.36 (+/- 0.03)
Max features: 5, num estimators: 50, accuracy: 0.39 (+/- 0.02)
Max features: 5, num estimators: 70, accuracy: 0.40 (+/- 0.01)
Max features: 5, num estimators: 90, accuracy: 0.41 (+/- 0.01)

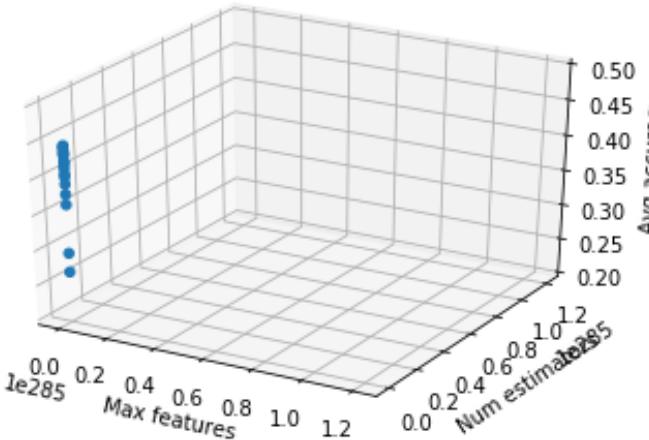
```

Gambar 3.197 Hasil 8 Bagian 1

```

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

```



Gambar 3.198 Hasil 8 Bagian 2

3.6.3 Penanganan Error

NameError: name 'dataset' is not defined : Biasanya karena typo, cek lagi penulisan kode

3.6.3.1 Tuliskan Kode Error dan Jenis Error NameError: name 'dataset' is not defined

3.6.3.2 Cara Penanganan Error Pastikan untuk tidak menulis function secara typo dan ceklagi

3.6.3.3 *Bukti Tidak Plagiat*



Gambar 3.199 Cek Plagiarism

3.6.4 Link Video Youtube

<https://youtu.be/hWCR0dXYagk>

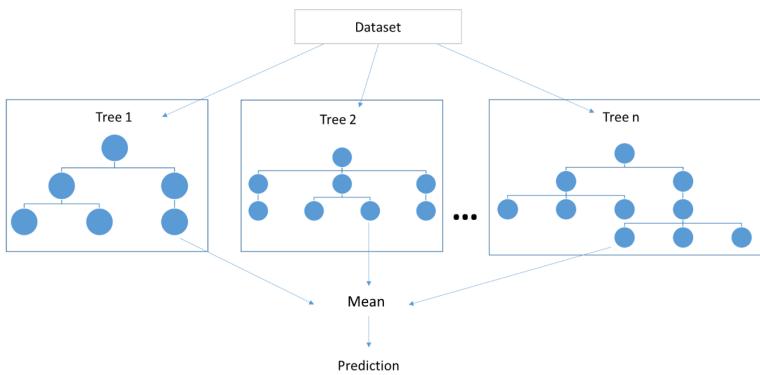
3.7 1174077 - Alvan Alvanzah

Chapter 3 - Prediksi dengan Random Forest

3.7.1 Teori

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

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



Gambar 3.200 Random forest

3.7.1.2 Jelaskan cara membaca dataset kasus dan artikan makna setiap file dan isi field masing masing file.

1. cara membaca dataset

- Gunakan librari Pandas pada python untuk dapat membaca dataset dengan format text file.
- Setelah itu,buat variabel baru ”dataset” yang berisikan perintah untuk membaca file csv. seperti berikut

```

1 import pandas as pd
2 dataset = pd.read_csv("C:/Users/ASUS/Downloads/KB3C-master
   (2)/KB3C-master/src/1174077/3/dataset.csv", sep=',')
3 dataset.head()

```

Pada kodingan diatas dapat dijelaskan bahwa : Memanggil Librari Panda untuk membaca dataset Membuat variabel ”Dataset” yang berisikan pd.read_csv untuk membaca dataset.

- setelah di run maka hasilnya seperti berikut

Index	Kecamatan	Jumlah Makam	Kapasitas
0	KECAMATAN BATU	27	132082
1	Desa Sumberjo	4	12130
2	Kelurahan Songgorerto	3	29900
3	Desa Pesanggrahan	4	18751
4	Kelurahan Ngaglik	4	20500
5	Kelurahan Sirih	2	14800
6	Desa Sidomulyo	4	13800
7	Kelurahan Tempe	3	7601
8	Desa Oro Oro Gede	3	16800
9	KECAMATAN BUMIAI	45	142125
10	Desa Giripurno	3	17500
11	Desa Pandanrejo	2	10700
12	Desa Bumiaji	5	21200
13	Desa Bulukerto	5	18250
14	Desa Punter	2	9000
15	Desa Tulungrejo	9	25800
16	Desa Sumbergondo	7	7075
17	Desa Gununganari	10	18400
18	Desa Sumberbrantas	2	15000
19	KECAMATAN JUNREJO	23	94145
20	Desa Beji	2	15000

Gambar 3.201 contoh dataset

Gambar diatas merupakan dataset Jumlah Tempat Pemakaman Umum (TPU) Kota Batu 2019. Datasetnya didapat dari laman <https://data.go.id/dataset-pemakaman-umum-tpu-kota-batu-2019>. Penjelasan dari isi field diatas adalah sebagai berikut :

- Atribut Index merupakan atribut otomatis untuk penomoran data yang ada.
- Atribut kecamatan merupakan kecamatan yang ada pada kota batu
- Atribut jumlah merupakan jumlah makam yang ada pada kecamatan
- Atribut kapasitas merupakan daya tampung pemakaman pada kecamatan

3.7.1.3 Jelaskan apa itu cross validation

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

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

Dimana Score 44 % didapatkan dari hasil pengelohan dataset jenis burung.

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

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

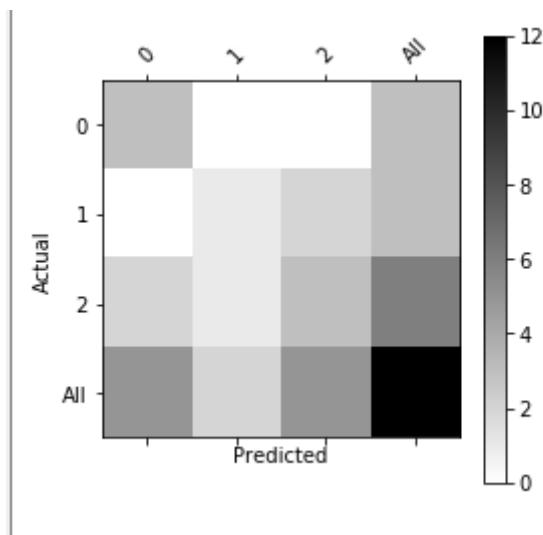
Perhitungan Confusion Matriks dapat dilakukan sebagai berikut. Disini saya menggunakan data yang dibuat sendiri untuk menampilkan data aktual dan prediksi.

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

```
1 import pandas as pd
2 import numpy as np
3 import matplotlib.pyplot as plt
4
5 y_actu = pd.Series([2, 0, 2, 2, 0, 1, 1, 2, 2, 0, 1, 2], name
6                   ='Actual')
7 y_pred = pd.Series([0, 0, 2, 1, 0, 2, 1, 0, 2, 0, 2, 2], name
8                   ='Predicted')
9 df_confusion = pd.crosstab(y_actu, y_pred, rownames=[ 'Actual' ,
10                           ], colnames=[ 'Predicted' ], margins=True)
11
12 def plot_confusion_matrix(df_confusion , title='Confusion
13                           matrix' , cmap=plt.cm.gray_r):
14     plt.matshow(df.confusion , cmap=cmap) # imshow
15     #plt.title(title)
16     plt.colorbar()
17     tick_marks = np.arange(len(df.confusion.columns))
18     plt.xticks(tick_marks , df.confusion.columns , rotation=45)
19     plt.yticks(tick_marks , df.confusion.index)
```

```
16     plt.ylabel(df_confusion.index.name)
17     plt.xlabel(df_confusion.columns.name)
18
19 plot_confusion_matrix(df_confusion)
```

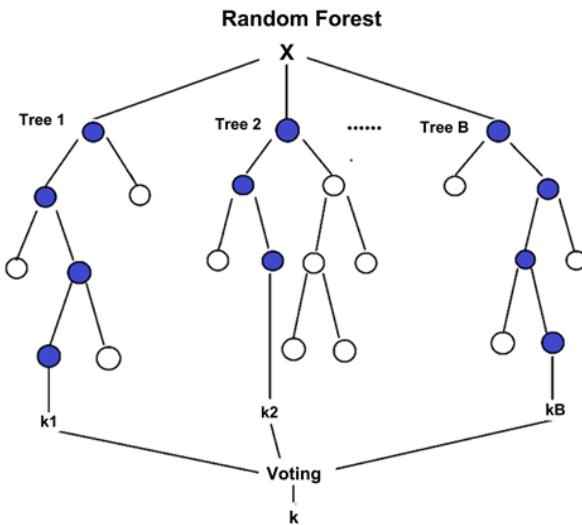
Hasil dari kode diatas akan menghasilkan confusion matrix seperti berikut:



Gambar 3.202 hasil confusion matrix

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

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



Gambar 3.203 voting random forest

3.7.2 Praktek

3.7.2.1 Nomor 1

```

1 import pandas as pd # Melakukan import library pandas sebagai pd
2
3 biodata = { 'Nama': [ 'Alvan', 'Ariq', 'Advent' ], 'hobi': [ 'Game', 'Nonton', 'Tidur' ], 'Umur': [ 20, 19, 21 ] }
4 #Membuat varibel yang bernama biodata , dan mengisi dataframanya dengan 3 field yaitu nama hobi dan umur
5 data = pd.DataFrame(biodata) # Membuat variabel data yang akan membuat DataFrame biodata menggunakan library pandas.
6 print(data) #print hasil dari data
  
```

	Nama	hobi	Umur
0	Ariq	Berenang	17
1	Reza	Tidur	18
2	Alvan	Game	19

Gambar 3.204 Membuat Aplikasi pakai pandas

3.7.2.2 Nomor 2

```

1 import numpy as np #Melakukan import library numpy sebaaai np
2 matrix = np.eye(5) #Membuat sebuah matriks identitas 5x5 pake
    numpy dengan menggunakan fungsi eye
3 print(matrix) #print matrix

```

```

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

```

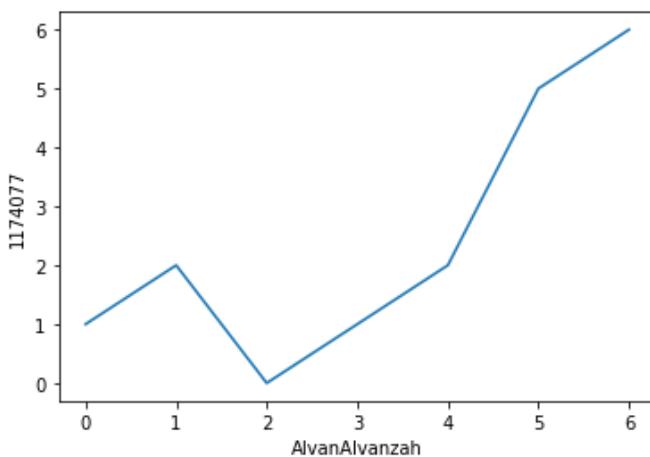
Gambar 3.205 Membuat Aplikasi pakai numpy

3.7.2.3 Nomor 3

```

1 import matplotlib.pyplot as pt #Melakukan import library numpy
    menjadi nama sendiri yaitu dirga
2 pt.plot([1,2,0,1,2,5,6]) #Memasukkan nilai pada plot
3 pt.xlabel('AlvanAlvanzah') #Menambahkan label pada x
4 pt.ylabel('1174077') #Menambahkan label pada y
5 pt.show() #Menampilkan grafik plot

```



Gambar 3.206 Membuat Aplikasi pakai matplotlib

3.7.2.4 Nomor 4

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

```
1 import pandas as pd #Melakukan import library numpy menjadi
2 pd
3 imgatt = pd.read_csv("E:/Kecerdasan Buatan/CUB_200_2011/
4 attributes/image_attribute_labels.txt",
5 sep='\s+', header=None, error_bad_lines=
6 False, warn_bad_lines=False,
7 usecols=[0,1,2], names=['imgid', 'attid',
8 , 'present']) #Membuat variable imgatt untuk membaca file
9 csv dari dataset
```

```
In [4]: import pandas as pd #Melakukan import library numpy menjadi pd
...
... imgatt = pd.read_csv("Kecadasan_Butan/CUB_200_2011/attributes/image_attribute_labels.txt",
... sep="+", header=None, error_bad_lines=False, warn_bad_lines=False,
... usecols=[0,1,2], names=['imgid', 'attid', 'present']) #Membuat variable imgatt untuk membaca file csv dari dataset, dengan ketentuan yang ada.
```

Gambar 3.207 Hasil 4 Bagian 1

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

```
1 imgatt.head() #Menampilkan data yang sudah dibaca , data  
    paling atas
```

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

Gambar 3.208 Hasil 4 Bagian 2

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

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

```
In [6]: imgatt.shape #Menampilkan jumlah seluruh data, kolom-nya
Out[6]: (3677856, 3)
```

Gambar 3.209 Hasil 4 Bagian 3

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

```
1 imgatt2 = imgatt.pivot(index='imgid', columns='attid', values='present') #Membuat sebuah variabel baru dari fungsi imgatt, dengan mengganti index menjadi kolom dan kolom menjadi index
```

```
In [7]: imgatt2 = imgatt.pivot(index='imgid', columns='attid', values='present') #Membuat sebuah variabel baru dari fungsi imgatt, dengan mengganti index menjadi kolom dan kolom menjadi index
```

Gambar 3.210 Hasil 4 Bagian 4

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

```
1 imgatt2.head() #Menampilkan data yang sudah dibaca tadi tapi cuman data paling atas
```

```

attid 1 2 3 4 5 6 7 ... 306 307 308 309 310 311 312
imgid
1 0 0 0 0 1 0 0 ...
2 0 0 0 0 0 0 0 ...
3 0 0 0 0 1 0 0 ...
4 0 0 0 0 0 1 0 0 ...
5 0 0 0 0 1 0 0 ...

```

[5 rows x 312 columns]

Gambar 3.211 Hasil 4 Bagian 5

- Menghasilkan jumlah kolom dan baris pada DataFrame imgatt2.

```
1 imgatt2.shape #Menampilkan jumlah seluruh data , kolom-nya
```

```
In [15]: imgatt2.shape #Menampilkan jumlah seluruh data, kolom-nya
Out[15]: (11788, 312)
```

Gambar 3.212 Hasil 4 Bagian 6

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

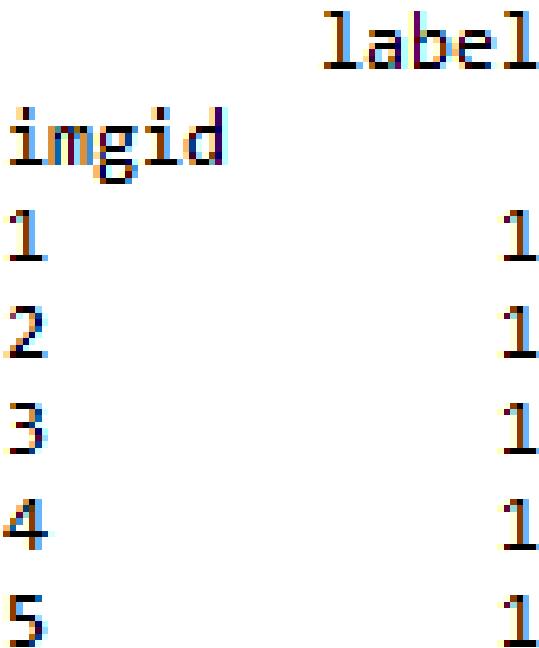
```
1 imglabels = pd.read_csv("E:/Kecerdasan Buatan/CUB_200_2011/
    image_class_labels.txt",
2                         sep=',', header=None, names=['imgid',
3                                     'label']) #Membaca data dimasukkan ke variable imglabels
4 imglabels = imglabels.set_index('imgid') #Variable imglabels
    dan set index (imgid)
```

```
In [16]: imglabels = pd.read_csv("E:/Kecerdasan Buatan/CUB_200_2011/image_class_labels.txt",
...:                         sep=',', header=None, names=['imgid', 'label']) #Membaca data dimasukkan
ke variable imglabels
...: imglabels = imglabels.set_index('imgid') #Variable imglabels dan set index (imgid)
```

Gambar 3.213 Hasil 4 Bagian 7

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

```
1 imglabels.head() #Menampilkan data yang sudah dibaca tadi
    tapi cuman data paling atas
```

**Gambar 3.214** Hasil 4 Bagian 8

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

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

```
In [18]: imglabels.shape #Menampilkan jumlah seluruh data, kolom-nya
Out[18]: (11788, 1)
```

Gambar 3.215 Hasil 4 Bagian 9

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

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

```
In [19]: df = imgatt2.join(imglabels) #Varibel df dimasukkan fungsi join dari data imgatt2 ke variabel
imglabels
...: df = df.sample(frac=1) #Varibel df sebagai sample dengan ketentuan frac=1
```

Gambar 3.216 Hasil 4 Bagian 10

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

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

```
In [20]: df_att = df.iloc[:, :312] #Membuat kolom dengan ketentuan 312
...: df_label = df.iloc[:, 312:] #Membuat kolom dengan ketentuan 312
```

Gambar 3.217 Hasil 4 Bagian 11

- Mengecek isi 5 data teratas pada df_att.

```
1 df_att.head() #Menampilkan data yang sudah dibaca tadi tapi
cuman data paling atas
```

```
In [26]: df_att.head() #Menampilkan data yang sudah dibaca tadi tapi cuman data paling atas
Out[26]:
```

1	2	3	4	5	6	7	...	306	307	308	309	310	311	312	
imgid							...								
10397	0	0	0	0	0	0	1	...	0	0	0	0	0	0	1
1108	0	0	0	0	0	0	1	...	0	0	0	1	0	0	0
2382	0	0	0	0	0	0	1	...	0	1	0	1	0	0	0
9374	0	0	0	0	0	0	1	...	0	0	0	0	0	0	0
9371	0	0	0	0	0	0	0	...	0	0	0	1	0	0	0

[5 rows x 312 columns]

Gambar 3.218 Hasil 4 Bagian 12

- Mengecek isi data teratas dari df_label.

```
1 df_label.head() #Menampilkan data yang sudah dibaca tadi tapi
cuman data paling atas
```

label	imgid
62	3574
149	8745
187	10967
199	11719
185	10878

Gambar 3.219 Hasil 4 Bagian 13

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

```

1 df_train_att = df_att[:8000] #Data akan dibagi dari 8000 row
   pertama menjadi data training dan sisanya adalah data
   testing
2 df_train_label = df_label[:8000] #Data akan dibagi dari 8000
   row pertama menjadi data training dan sisanya adalah data
   testing
3 df_test_att = df_att[8000:] #Berbalik dari sebelumnya data
   akan dibagi mulai dari 8000 row pertama menjadi data
   training dan sisanya adalah data testing
4 df_test_label = df_label[8000:] #Berbalik dari sebelumnya
   data akan dibagi mulai dari 8000 row pertama menjadi data
   training dan sisanya adalah data testing
5
6 df_train_label = df_train_label['label'] #Menambahkan label
7 df_test_label = df_test_label['label'] #Menambahkan label

```

```

In [23]: df_train_att = df_att[:8000] #Data akan dibagi dari 8000 row pertama menjadi data training dan
sisanya adalah data testing
...: df_train_label = df_label[:8000] #Data akan dibagi dari 8000 row pertama menjadi data training dan
sisanya adalah data testing
...: df_test_att = df_att[8000:] #Berbalik dari sebelumnya data akan dibagi mulai dari 8000 row pertama
menjadi data training dan sisanya adalah data testing
...: df_test_label = df_label[8000:] #Berbalik dari sebelumnya data akan dibagi mulai dari 8000 row
pertama menjadi data training dan sisanya adalah data testing
...:
...: df_train_label = df_train_label['label'] #Menambahkan Label
...: df_test_label = df_test_label['label'] #Menambahkan Label

```

Gambar 3.220 Hasil 4 Bagian 14

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

```

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

```

```
In [24]: from sklearn.ensemble import RandomForestClassifier #Import fungsi randomforestclassifier
...: clf = RandomForestClassifier(max_features=50, random_state=0, n_estimators=100) #clf sebagai variabel untuk klafifikasi random forest
```

Gambar 3.221 Hasil 4 Bagian 15

- Menunjukkan hasil prediksi dari Random Forest.

```

1 clf.fit(df_train_att, df_train_label) #Variabnle clf untuk fit yaitu menjadi data training

```

```
In [25]: clf.fit(df_train_att, df_train_label) #Variabnle clf untuk fit yaitu menjadi data training
Out[25]:
RandomForestClassifier(bootstrap=True, class_weight=None, criterion='gini',
   max_depth=None, max_features=50, max_leaf_nodes=None,
   min_impurity_decrease=0.0, min_impurity_split=None,
   min_samples_leaf=1, min_samples_split=2,
   min_weight_fraction_leaf=0.0, n_estimators=100,
   n_jobs=None, oob_score=False, random_state=0, verbose=0,
   warm_start=False)
```

Gambar 3.222 Hasil 4 Bagian 16

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

```

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

```

```
In [26]: print(clf.predict(df_train_att.head())) #Print clf yang di sudah prediksi dari training tetapi hanya menampilkan data paling atas
[ 62 149 187 199 185]
```

Gambar 3.223 Hasil 4 Bagian 17

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

```

1 clf.score(df_test_att, df_test_label) #Memunculkan clf sebagai testing yang sudah di training tadi

```

```
In [28]: clf.score(df_test_att, df_test_label) #Memunculkan clf sebagai testing yang sudah di training tadi
Out[28]: 0.4390179514255544
```

Gambar 3.224 Hasil 4 Bagian 18

3.7.2.5 Nomor 5

```

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

```

```

In [29]: from sklearn.metrics import confusion_matrix #Mengimport Confusion Matrix
...: pred_labels = clf.predict(df_test_att) #Membuat variable pred_labels dari data testing
...: cm = confusion_matrix(df_test_label, pred_labels) #cm sebagai variabel data label

```

Gambar 3.225 Hasil 5 Bagian 1

```

1 cm #Memunculkan data label berbentuk array

```

```

array([[ 6,  2,  0, ...,  0,  1,  0],
       [ 0, 15,  0, ...,  0,  0,  0],
       [ 5,  1,  7, ...,  0,  0,  0],
       ...,
       [ 0,  0,  1, ...,  2,  0,  0],
       [ 0,  0,  0, ...,  0,  9,  0],
       [ 0,  0,  0, ...,  0,  0, 16]], dtype=int64)

```

Gambar 3.226 Hasil 5 Bagian 2

```

1 import matplotlib.pyplot as plt #Mengimport library matplotlib
    sebagai plt
2 import itertools #Mengimport library itertools
3 def plot_confusion_matrix(cm, classes,
                           normalize=False,
                           title='Confusion matrix',
                           cmap=plt.cm.Blues): #Membuat fungsi
    dengan ketentuan data yang ada pada cm
7
8     if normalize:
9         cm = cm.astype('float') / cm.sum(axis=1)[:, np.newaxis]
10        print("Normalized confusion matrix") #Jika normalisasi
11        sebagai ketentuan yang ada maka print normalized confusion
12        matrix
13    else:
14        print('Confusion matrix, without normalization') #Jika
15        tidak memenuhi kondisi if maka pring else
16
17    print(cm) #Print data cm
18
19    plt.imshow(cm, interpolation='nearest', cmap=cmap) #plt
20    sebagai fungsi untuk membuat plot

```

```

17 plt.title(title) #Membuat title pada plot
18 #plt.colorbar()
19 tick_marks = np.arange(len(classes)) #Membuat marks pada plot
20 plt.xticks(tick_marks, classes, rotation=90) #Membuat ticks
21 pada x
22 plt.yticks(tick_marks, classes) #Membuat ticks pada y
23
24 fmt = '.2f' if normalize else 'd' #fmt sebagai normalisasi
25 thresh = cm.max() / 2. #Variable thresh menambil data max
26 pada cm kemudian dibagi 2
27
28 plt.tight_layout() #Mengatur layout pada plot
29 plt.ylabel('True label') #Menambahkan nama label pada sumbu y
30 plt.xlabel('Predicted label') #Menambahkan nama label pada
31 sumbu x

```

```

In [32]: import matplotlib.pyplot as plt #Mengimport library matplotlib sebagai plt
...: import itertools #Mengimport library itertools
...: def plot_confusion_matrix(cm, classes,
...:                          normalize=False,
...:                          title='Confusion matrix',
...:                          cmap=plt.cm.Blues): #Membuat fungsi dengan ketentuan data yang ada pada
cm
...:
...:     if normalize:
...:         cm = cm.astype('float') / cm.sum(axis=1)[:, np.newaxis]
...:         print("Normalized confusion matrix") #Jika normalisasi sebagai ketentuan yang ada maka
print normalized confusion matrix
...:     else:
...:         print('Confusion matrix, without normalization') #Jika tidak memenuhi kondisi if maka
print else
...:
...:     print(cm) #Print data cm
...:
...:     plt.imshow(cm, interpolation='nearest', cmap=cmap) #plt sebagai fungsi untuk membuat plot
...:     plt.title(title) #Membuat title pada plot
...:     #plt.colorbar()
...:     tick_marks = np.arange(len(classes)) #Membuat marks pada plot
...:     plt.xticks(tick_marks, classes, rotation=90) #Membuat ticks pada x
...:     plt.yticks(tick_marks, classes) #Membuat ticks pada y
...:
...:     fmt = '.2f' if normalize else 'd' #fmt sebagai normalisasi
...:     thresh = cm.max() / 2. #Variable thresh menambil data max pada cm kemudian dibagi 2
...:
...:     plt.tight_layout() #Mengatur layout pada plot
...:     plt.ylabel('True label') #Menambahkan nama label pada sumbu y
...:     plt.xlabel('Predicted label') #Menambahkan nama label pada sumbu x

```

Gambar 3.227 Hasil 5 Bagian 3

```

1 birds = pd.read_csv("E:/Kecerdasan Buatan/CUB_200_2011/classes .
txt",
2                                     sep='\s+', header=None, usecols=[1], names=[ 'birdname']) #membaca csv dengan ketentuan nama birdname
3 birds = birds['birdname'] #nama birds dengan ketentuan birdname
4 birds #Menampilkan data birds

```

```

187          188.Pileated_Woodpecker
188          189.Red_bellied_Woodpecker
189          190.Red_cockaded_Woodpecker
190          191.Red_headed_Woodpecker
191          192.Downy_Woodpecker
192          193.Bewick_Wren
193          194.Cactus_Wren
194          195.Carolina_Wren
195          196.House_Wren
196          197.Marsh_Wren
197          198.Rock_Wren
198          199.Winter_Wren
199          200.Common_Yellowthroat
Name: birdname, Length: 200, dtype: object

```

Gambar 3.228 Hasil 5 Bagian 4

```

1 import numpy as np #Mengimport library numpy sebagai np
2 np.set_printoptions(precision=2) #np sebagai variabel yang
   membuat set precision=2
3 plt.figure(figsize=(60,60), dpi=300) #Plot sebagai figure dengan
   ketentuan sizw 60,60 dan dpi 300
4 plot_confusion_matrix(cm, classes=birds, normalize=True) #Data cm
   dan clas birds dibuat sebagai plot
5 plt.show()
6 #plt.savefig('hasil.png')

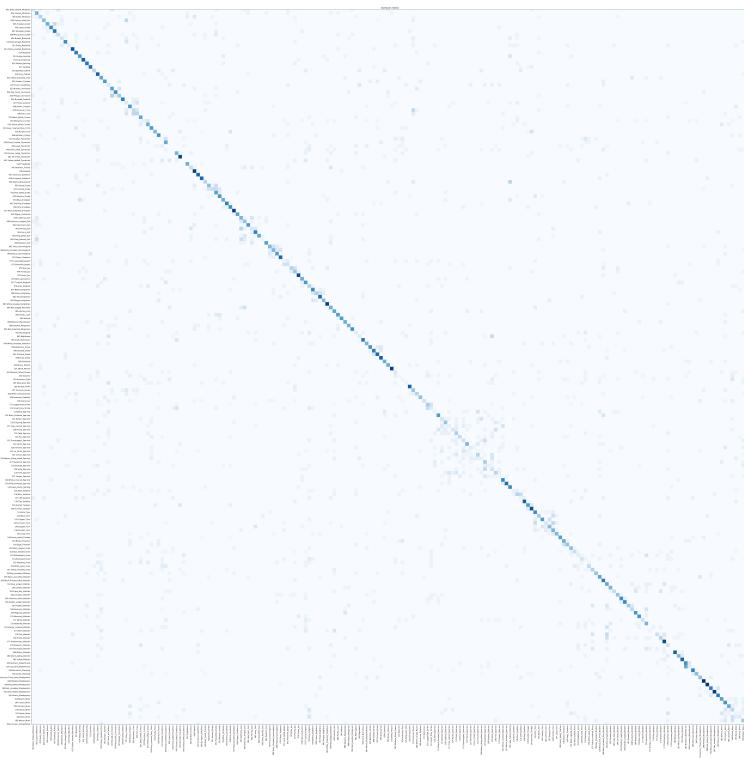
```

```

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

```

Gambar 3.229 Hasil 5 Bagian 5



Gambar 3.230 Plot Hasil 5 Bagian 5

3.7.2.6 Nomor 6

```

1 from sklearn import tree #Mengimport library tree
2 clftree = tree.DecisionTreeClassifier() #clftree sebagai variabel
   untuk decision tree
3 clftree.fit(df_train_att, df_train_label) #Mengatur data training
4 clftree.score(df_test_att, df_test_label) #Mengatur data testing

In [26]: from sklearn import tree #Mengimport Library tree
...: clftree = tree.DecisionTreeClassifier() #clftree sebagai variabel untuk decision tree
...: clftree.fit(df_train_att, df_train_label) #Mengatur data training
...: clftree.score(df_test_att, df_test_label) #Mengatur data testing
Out[26]: 0.264519535374868

```

Gambar 3.231 Hasil 6 Bagian 1

```

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

```

```
In [27]: from sklearn import svm #Mengimport library svm
...: clfsvm = svm.SVC() #clfsvm sebagai variabel untuk mengatur fungsi SVC
...: clfsvm.fit(df_train_att, df_train_label) #Mengatur data training
...: clfsvm.score(df_test_att, df_test_label) #Mengatur data testing
C:\Users\Panda23\Anaconda3\lib\site-packages\sklearn\svm\base.py:193: FutureWarning: The default value of gamma will change from 'auto' to 'scale' in version 0.22 to account better for unscaled features. Set gamma explicitly to 'auto' or 'scale' to avoid this warning.
"avoid this warning.", FutureWarning)

Out[27]: 0.2819429778247096C:\Users\Panda23\Anaconda3\lib\site-packages\sklearn\svm\base.py:193:
FutureWarning: The default value of gamma will change from 'auto' to 'scale' in version 0.22 to account
better for unscaled features. Set gamma explicitly to 'auto' or 'scale' to avoid this warning.
"avoid this warning.", FutureWarning)
```

Gambar 3.232 Hasil 6 Bagian 2

3.7.2.7 Nomor 7

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

```
In [28]: from sklearn.model_selection import cross_val_score #Mengimport cross_val_score
...: scores = cross_val_score(clf, df_train_att, df_train_label, cv=5) #Membuat variable scores sebagai
variabel prediksi dari data training
...: print("Accuracy: %0.2f (+/- %0.2f)" % (scores.mean(), scores.std() * 2)) #Print data scores dengan
ketentuan akurasi
Out[28]: 0.2819429778247096Accuracy: 0.44 (+/- 0.03)
```

Gambar 3.233 Hasil 7 Bagian 1

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

```
In [30]: scorestree = cross_val_score(clftree, df_train_att, df_train_label, cv=5) #Membuat variable
prediksi menggunakan scores dan metode tree
...: print("Accuracy: %0.2f (+/- %0.2f)" % (scorestree.mean(), scorestree.std() * 2)) #Menampilkan
dengan ketentuan yang ada
Accuracy: 0.25 (+/- 0.02)
```

Gambar 3.234 Hasil 7 Bagian 2

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

```
In [31]: scoressvm = cross_val_score(clfsvm, df_train_att, df_train_label, cv=5) #Membuat variable data
training
...: print("Accuracy: %.2f (+/- %.2f)" % (scoressvm.mean(), scoressvm.std() * 2)) #Menampilkan data
testing dan output akurasi
C:\Users\Panda23\Anaconda3\lib\site-packages\sklearn\svm\base.py:193: FutureWarning: The default value of
gamma will change from 'auto' to 'scale' in version 0.22 to account better for unscaled features. Set gamma
explicitly to 'auto' or 'scale' to avoid this warning.
"avoid this warning.", FutureWarning)
C:\Users\Panda23\Anaconda3\lib\site-packages\sklearn\svm\base.py:193: FutureWarning: The default value of
gamma will change from 'auto' to 'scale' in version 0.22 to account better for unscaled features. Set gamma
explicitly to 'auto' or 'scale' to avoid this warning.
"avoid this warning.", FutureWarning)
C:\Users\Panda23\Anaconda3\lib\site-packages\sklearn\svm\base.py:193: FutureWarning: The default value of
gamma will change from 'auto' to 'scale' in version 0.22 to account better for unscaled features. Set gamma
explicitly to 'auto' or 'scale' to avoid this warning.
"avoid this warning.", FutureWarning)
C:\Users\Panda23\Anaconda3\lib\site-packages\sklearn\svm\base.py:193: FutureWarning: The default value of
gamma will change from 'auto' to 'scale' in version 0.22 to account better for unscaled features. Set gamma
explicitly to 'auto' or 'scale' to avoid this warning.
"avoid this warning.", FutureWarning)
C:\Users\Panda23\Anaconda3\lib\site-packages\sklearn\svm\base.py:193: FutureWarning: The default value of
gamma will change from 'auto' to 'scale' in version 0.22 to account better for unscaled features. Set gamma
explicitly to 'auto' or 'scale' to avoid this warning.
"avoid this warning.", FutureWarning)
Accuracy: 0.26 (+/- 0.02)
```

Gambar 3.235 Hasil 7 Bagian 3

3.7.2.8 Nomor 8

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

```

....      ..._param[4] = scores.mean() + scores.std() * 2
....      rf_params[i,3] = scores.std() * 2 #index 3
....      i += 1 #Dengan ketentuan i += 1
....      print("Max features: %d, num estimators: %d, acc
n_estimators, scores.mean(), scores.std() * 2))
....      #Print hasil pengulangan yang sudah ditentukan
Max features: 5, num estimators: 10, accuracy: 0.26 (+/- 0.03)
Max features: 5, num estimators: 30, accuracy: 0.36 (+/- 0.03)
Max features: 5, num estimators: 50, accuracy: 0.39 (+/- 0.03)
Max features: 5, num estimators: 70, accuracy: 0.41 (+/- 0.02)
Max features: 5, num estimators: 90, accuracy: 0.42 (+/- 0.02)
Max features: 5, num estimators: 110, accuracy: 0.43 (+/- 0.03)
Max features: 5, num estimators: 130, accuracy: 0.43 (+/- 0.02)
Max features: 5, num estimators: 150, accuracy: 0.44 (+/- 0.02)
Max features: 5, num estimators: 170, accuracy: 0.44 (+/- 0.03)
Max features: 5, num estimators: 190, accuracy: 0.45 (+/- 0.01)
Max features: 10, num estimators: 10, accuracy: 0.29 (+/- 0.02)
Max features: 10, num estimators: 30, accuracy: 0.38 (+/- 0.02)
Max features: 10, num estimators: 50, accuracy: 0.41 (+/- 0.03)
Max features: 10, num estimators: 70, accuracy: 0.43 (+/- 0.02)
Max features: 10, num estimators: 90, accuracy: 0.43 (+/- 0.02)
Max features: 10, num estimators: 110, accuracy: 0.44 (+/- 0.02)
Max features: 10, num estimators: 130, accuracy: 0.45 (+/- 0.03)
Max features: 10, num estimators: 150, accuracy: 0.45 (+/- 0.03)
Max features: 10, num estimators: 170, accuracy: 0.45 (+/- 0.02)
Max features: 10, num estimators: 190, accuracy: 0.45 (+/- 0.03)
Max features: 15, num estimators: 10, accuracy: 0.31 (+/- 0.03)
Max features: 15, num estimators: 30, accuracy: 0.40 (+/- 0.03)
Max features: 15, num estimators: 50, accuracy: 0.42 (+/- 0.03)
Max features: 15, num estimators: 70, accuracy: 0.43 (+/- 0.02)
Max features: 15, num estimators: 90, accuracy: 0.43 (+/- 0.03)
Max features: 15, num estimators: 110, accuracy: 0.44 (+/- 0.02)
Max features: 15, num estimators: 130, accuracy: 0.44 (+/- 0.02)
Max features: 15, num estimators: 150, accuracy: 0.45 (+/- 0.02)
Max features: 15, num estimators: 170, accuracy: 0.45 (+/- 0.02)
Max features: 15, num estimators: 190, accuracy: 0.45 (+/- 0.02)

```

Gambar 3.236 Hasil 8 Bagian 1

```

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

```

Gambar 3.237 Hasil 8 Bagian 1 Akhir kode

```

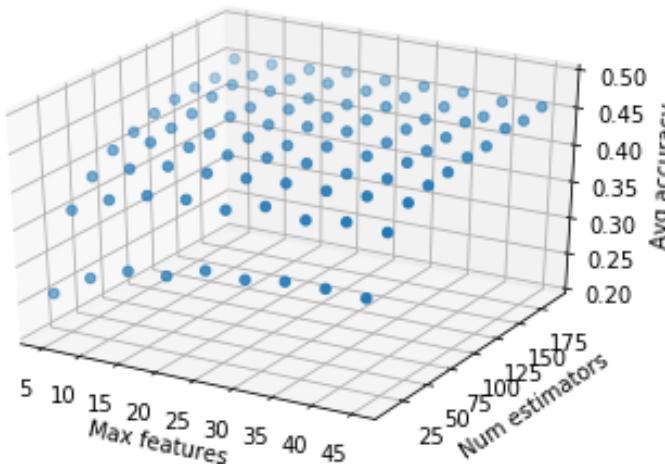
1 import matplotlib.pyplot as plt #Mengimport library matplotlib
sebagai plt

```

```

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

```



Gambar 3.238 Hasil 8 Bagian 2

3.7.3 Penanganan Error

1. ScreenShoot Error

```

File "<ipython-input-2-442c23aa66173>", line 1, in <module>
    import pandas as pd # Melakukan import library pandas sebagai pd
ModuleNotFoundError: No module named 'pandas'

```

Gambar 3.239 NoModuleNotFoundError

2. Tuliskan Kode Error dan Jenis Error

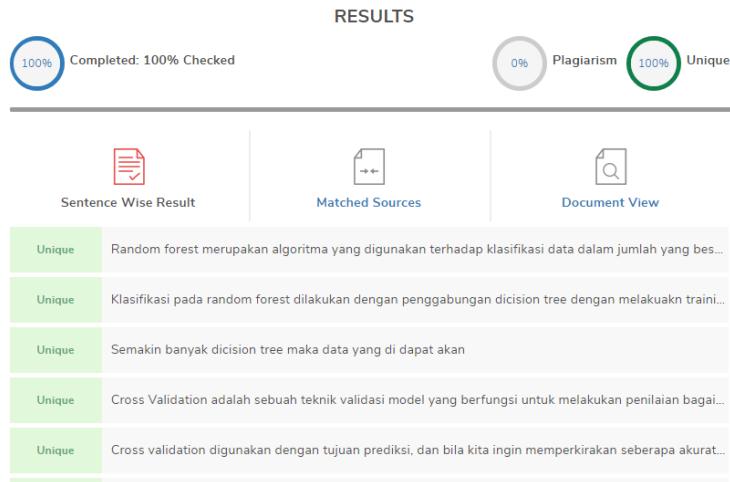
- NoModuleNotFoundError

3. Cara Penangan Error

- NoModuleNotFoundError

Penulisan nama modul harus sesuai atau menginstall module yang diinginkan

3.7.4 Bukti Tidak Plagiat



Gambar 3.240 **Bukti Tidak Plagiat**

3.8 1174054 - Aulyardha Anindita

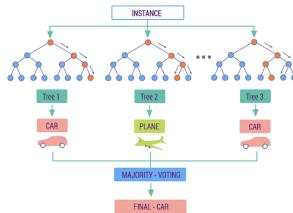
3.8.1 Teori

1. Random Forest

Random Forest adalah salah satu metode yang ada di Decision Tree. Decision Tree yaitu suatu diagram atau yang biasa dikenal dengan pohon pengambil keputusan merupakan suatu diagram alir yang memiliki bentuk seperti pohon yang digunakan untuk mengumpulkan data atau root node. Random Forest dapat juga diartikan sebagai kombinasi dari setiap tree yang kemudian dikombinasikan dalam suatu model yang bergantung pada sebuah nilai vektor random dari distribusi yang sama pada semua tree yang masing-masing memiliki descision tree tergantung dari kedalamannya yang maksimal.

Random Forest juga merupakan algoritma yang biasa digunakan untuk meklasifikasikan data dalam jumlah yang besar. klasifikasi tersebut di-

lakukan dengan penggabungan pohon yang akan mempengaruhi akurasi yang didapatkan.



Gambar 3.241 Random Forest

2. Cara Membaca Dataset

- Untuk langkah pertama, kita mendownload terlebih dahulu file dataset nya, disini saya mengambil dataset burung yang ada pada teori
- Setelah download selesai, buka software spyder untuk melihat isi dari file dataset yang telah kita download
- Isi dari data tersebut memiliki extensi file yang bernama .txt yang didalamnya terdapat class dari field.
- Contohnya pada data jenis burung memiliki file index dan angka, dimana index tersebut berisi angka yang memiliki arti jenis burung dan nama burung
- Sedangkan pada field memiliki isi nilai yang berupa 0 dan 1 dimana memiliki sifat boolean yaitu true or false
- Hal ini terjadi karena komputer hanya bisa membaca bilangan biner sehingga field tersebut berupa angka, angka yang dimaksud disini adalah angka 0 yang berarti false dan angka 1 yang berarti true

3. Cross Validation

Cross Validation (CV) merupakan suatu metode statistik yang digunakan untuk mengevaluasi kinerja model atau algoritma dengan cara data dipisahkan menjadi dua subset yaitu data proses pembelajaran dan data validasi atau evaluasi. suatu model atau algoritma tersebut dilatih oleh subset untuk pembelajaran kemudian divalidasi oleh subset validasi. sehingga pemilihan jenis cross validation dapat didasarkan pada ukuran dataset, biasanya cross validation k-fold berguna untuk mengurangi waktu komputasi dengan menjaga keakuratan estimasi. Tujuan dari cross validation yaitu untuk mendefinisikan dataset guna menguji dalam fase pelatihan untuk membatasi masalah seperti overfitting dan underfitting serta mendapatkan wawasan tentang bagaimana model akan digeneralisasikan ke set data independen.

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

Score 44 % diperoleh dari hasil pengolahan dataset jenis burung. Hal ini dilakukan untuk proses pembagian data testing dan data training kemandirian diproses dan menghasilkan score sebanyak 44 % yang artinya bahwa score tersebut digunakan sebagai pembanding dalam tingkat keakuratannya.

Pada Decision Tree memperoleh data yang lebih kecil yaitu sebanyak 27 % hal ini disebabkan karena data yang diolah menggunakan decision tree dibagi menjadi beberapa tree, sehingga dapat disimpulkan hal ini dilakukan untuk mendapatkan data yang akurat.

Pada SVM memperoleh score sebanyak 29 % hal ini disebabkan karena data yang dimiliki masih bernilai netral sehingga tingkat keakuratannya masih belum jelas.

5. Cara Membaca Confusion Matriks

Untuk membaca confusion matriks dapat menggunakan source code sebagai berikut :

```

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

```

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

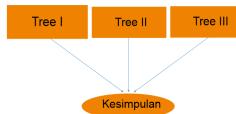


Gambar 3.242 Confusion Matriks

6. Voting pada random forest

Voting pada random forest ialah sebuah proses pemilihan dari tree yang

dimana akan dimunculkan hasilnya dan disimpulkan menjadi informasi yang pasti. Untuk lebih jelasnya saya akan memberikan sebuah contoh bagaimana voting bekerja :



Gambar 3.243 Decision Tree.

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

3.8.2 Praktek

1. Aplikasi Sederhana Menggunakan Pandas

```

1 import pandas as pd #digunakan untuk mengimport library pandas
2 dita = pd.read_csv('D:/Mata Kuliah/Tingkat 3/Semester 6/Kecerdasan
 3          Buatan/Chapter 3/drakor.csv',sep=',') # digunakan untuk memanggil file csv dan dipisahkan dengan ;
4 len(dita) #untuk mengetahui jumlah data
5 print(dita.juduldrakor) #digunakan untuk mengetahui kolom
 6          nama dari variabel dita
  
```

Hasilnya :

```

.... Import pandas as pd #digunakan untuk mengimport library pandas
.... dita = pd.read_csv('D:/Mata Kuliah/Tingkat 3/Semester 6/Kecerdasan
Buatn/Chapter 3/drakor.csv',sep=',') #digunakan untuk memanggil file csv dan
dipisahkan dengan ;
.... len(dita) #untuk mengetahui jumlah data
.... print(dita.juduldrakor) #digunakan untuk mengetahui kolom nama dari
variabel dita
0   Crash Landing On You
1   Chocolate
2   Forest
3   Queen's Gambit
4   Haikili
5   Extraordinary You
6   Romantic Doctor Kim
7   Goong
8   School
Name: JudulDrakor, dtype: object
  
```

Gambar 3.244 Hasil Nomor 1

2. Nomor 2

```

1 # In [2]: Soal2
2 import numpy as np #digunakan untuk mengimport library numpy
3 episode = np.sum(dita.jumlahepisode) # menggunakan fungsi sum
   bawaan dari numpy
4 print(episode) #menampilkan jumlah episode dari variabel
   episode

```

Hasilnya :

```

In [2]: import numpy as np #digunakan untuk mengimport library numpy
...: episode = np.sum(dita.jumlahepisode) # menggunakan fungsi sum bawaan dari
...: numpy
...: print(episode) menampilkan jumlah episode dari variabel episode
168

```

Gambar 3.245 Hasil Nomor 2

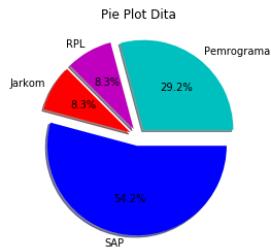
3. Nomor 3

```

1 # In [3]: Soal3
2 from matplotlib import pyplot as plt #digunakan untuk
   mengimport library matplotlib
3 Pemrograman = [1,2,3,4,5,6,7] #membuat variabel dengan nama
   pemrograman
4 RPL =[7,8,6,11,7,8,9] ##membuat variabel dengan nama RPL
5 Jarkom = [2,3,4,3,2,7,5] ##membuat variabel dengan nama
   jarkom
6 MB =[7,8,7,2,2,4,2] #membuat variabel dengan nama MB
7 SAP = [8,5,7,8,13,7,15] #membuat variabel dengan nama SAP
8 slices = [7,2,2,13] #membuat objek irisan dari variabel yang
   sudah dispesifikasi
9 activities = [ 'Pemrograman' , 'RPL' , 'Jarkom' , 'SAP' ] #membuat
   judul dari masing-masing variabel
10 cols = [ 'c' , 'm' , 'r' , 'b' ] #mengatur warna atau memberikan
   warna pada setiap item
11 plt.pie(slices , #untuk mengatur plot pie
12 labels=activities ,
13 colors=cols ,
14 startangle=0,
15 shadow= True ,
16 explode=(0.1 ,0.1 ,0.1 ,0.1 ) ,
17 autopct='%.1f%%')
18 plt.title('Pie Plot Dita') #memberikan judul pada pie plot
19 plt.show() #menampilkan graph

```

Hasilnya :



Gambar 3.246 Hasil Nomor 3

4. Nomor 4

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

```

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

```

Hasilnya :

```

In [26]: import pandas as pd #import library pandas sebagai as
...
Out[26]: imgatt = pd.read_csv("D:/Mata Kuliah/Tingkat 3/Semester 6/Kecerdasan
Buatan/Chapter 3/CUB_200_2011/attributes/image_attribute_labels.txt",
sep='\s+', header=None, error_bad_lines=False,
warn_bad_lines=False,
usecols=[0,1,2], names=['imgid', 'attid',
'present']) #library imgatt sebagai membaca file csv dari dataset, dengan
ketentuan yang ada.

```

Gambar 3.247 Hasil Soal 4 - 1

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

```

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

```

Hasilnya :

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

Gambar 3.248 Hasil Soal 4 - 2

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

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

Hasilnya :

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

Gambar 3.249 Hasil Soal 4 - 3

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

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

Hasilnya :

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

Gambar 3.250 Hasil Soal 4 - 4

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

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

Hasilnya :

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

[5 rows x 312 columns]

Gambar 3.251 Hasil Soal 4 - 5

- Menghasilkan jumlah kolom dan baris pada DataFrame imgatt2.

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

Hasilnya :

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

Gambar 3.252 Hasil Soal 4 - 6

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

```
1 imglabels = pd.read_csv("D:/Mata Kuliah/Tingkat 3/Semester  
6/Kecerdasan Buatan/Chapter 3/CUB_200_2011/  
image_class_labels.txt",  
2 sep=',', header=None, names=['imgid',  
3 'label']) #baca data csv dengan ketentuan yang ada  
4 imglabels = imglabels.set_index('imgid') #variabel  
imglabels sebagai set index (imgid)
```

Hasilnya :

In [8]:	imglabels = pd.read_csv("D:/Mata Kuliah/Tingkat 3/Semester 6/Kecerdasan Buatan/Chapter 3/CUB_200_2011/ image_class_labels.txt", sep=',', header=None, names=['imgid', 'label']) #description from dataset README: # # ground truth class labels (bird species labels) for each image # # imgid is the image identifier, with each line corresponding to one image. # # imgid, label # # where imgid and class_label correspond to the line in images.txt and classes.txt, # respectively.
Out[8]:	imglabels.head()
	label
	1
	2
	3
	4
	5

Gambar 3.253 Hasil Soal 4 - 7

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

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

Hasilnya :

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

Gambar 3.254 Hasil Soal 4 - 8

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

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

Hasilnya :

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

Gambar 3.255 Hasil Soal 4 - 9

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

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

Hasilnya :

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

Gambar 3.256 Hasil Soal 4 - 10

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

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

Hasilnya :

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

Gambar 3.257 Hasil Soal 4 - 11

- Mengecek isi 5 data teratas pada df att.

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

Hasilnya:

```
In [13]: df_att.head()
Out[13]:
   1  2  3  4  5  6  7  ...  306 307 308 309 310 311 312
imgid  0  0  0  0  0  0  1  ...  0  0  0  0  0  0  0  1
988   0  0  0  0  0  0  0  ...  0  0  0  0  0  0  0  0
7472  0  0  0  0  0  0  0  ...  0  0  0  0  0  0  0  0
7084  0  0  0  0  0  1  0  ...  0  0  0  0  1  0  0  0
9834  0  0  0  0  0  0  1  ...  0  0  1  0  0  0  0  1
[5 rows x 312 columns]
```

Gambar 3.258 Hasil Soal 4 - 12

- Mengecek isi data teratas dari df_label.

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

Hasilnya:

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

Gambar 3.259 Hasil Soal 4 - 13

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

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

Hasilnya:

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

Gambar 3.260 Hasil Soal 4 - 14

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

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

Hasilnya:

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

Gambar 3.261 Hasil Soal 4 - 15

- Menunjukkan hasil prediksi dari Random Forest.

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

Hasilnya:

```
In [17]: clf.fit(df_train_att, df_train_label)
Out[17]:
RandomForestClassifier(bootstrap=True, class_weight=None,
criterion='gini', max_depth=None, max_features=50,
max_leaf_nodes=None, max_samples=None,
min_impurity_decrease=0.0, min_impurity_split=None,
min_samples_leaf=1, min_samples_split=2,
n_estimators=100, n_jobs=-1, oob_score=False, random_state=0,
verbose=0, warm_start=False)
```

Gambar 3.262 Hasil Soal 4 - 16

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

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

Hasilnya:

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

Gambar 3.263 Hasil Soal 4 - 17

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

Gambar 3.264 Hasil Soal 4 - 17

5. Nomor 5

- Melakukan import confusion matrix pada library sklearn matriks.

```

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

```

Hasilnya:

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

Gambar 3.265 Hasil Soal 5 - 1

- Menampilkan isi cm dalam bentuk matrix yang berupa array.

```
1 cm #menampilkan data label berbentuk array
```

Hasilnya:

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

Gambar 3.266 Hasil Soal 5 - 2

- Membuat dan menampilkan hasil plot.

```

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

```

```
19 tick_marks = np.arange(len(classes)) #membuat marks  
20 pada plot  
21 plt.xticks(tick_marks, classes, rotation=90) #membuat  
22 ticks pada x  
23 plt.yticks(tick_marks, classes) #membuat ticks pada y  
24  
25 fmt = '.2f' if normalize else 'd' #fmt sebagai  
26 normalisasi  
27 thresh = cm.max() / 2. #variabel thresh menambil data  
28 max pada cm kemudian dibagi 2  
29  
30 plt.tight_layout() #mengatur layout pada plot  
31 plt.ylabel('True label') #memberi nama label pada sumbu  
32 y  
33 plt.xlabel('Predicted label') #memberi nama label pada  
34 sumbu x  
35  
36 # In [22]:  
37  
38 birds = pd.read_csv("D:/Mata Kuliah/Tingkat 3/Semester 6/  
39 Kecerdasan Buatan/Chapter 3/CUB_200_2011/classes.txt",  
40 sep='\s+', header=None, usecols=[1],  
41 names=['birdname']) #membaca csv dengan ketentuan nama  
42 birdname  
43 birds = birds[['birdname']] #nama birds dengan ketentuan  
44 birdname  
45 birds #menampilkan data birds  
46  
47 # In [23]:  
48  
49 import numpy as np #import library numpy sebagai np  
50 np.set_printoptions(precision=2) #np sebagai variabel yang  
51 membuat set precision=2  
52 plt.figure(figsize=(60,60), dpi=300) #plot sebagai figure  
53 dengan ketentuan size 60,60 dan dpi 300  
54 plot_confusion_matrix(cm, classes=birds, normalize=True) #  
55 data cm dan clas birds dibuat sebagai plot
```

Hasilnya:

Gambar 3.267 Hasil Soal 5 - 3

6. Nomor 6

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

```

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

```

Hasilnya:

```

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

```

Gambar 3.268 Hasil Soal 6 - 1

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

```

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

```

Hasilnya:

```

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

```

Gambar 3.269 Hasil Soal 6 - 2

7. Nomor 7

- Menunjukkan hasil cross validation pada Random Forest.

```

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

```

Hasilnya:

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

Gambar 3.270 Hasil Soal 7 - 1

- Menunjukkan hasil cross validation pada Desicion Tree.

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

Hasilnya:

```
In [30]: scorestree = cross_val_score(clftree, df_train_att, df_train_label, cv=5)
...: print("Accuracy: %0.2f (+/- %0.2f)" % (scorestree.mean(), scorestree.std()))
...: Accuracy: 0.26 (+/- 0.03)
```

Gambar 3.271 Hasil Soal 7 - 2

- Menunjukkan hasil cross validation pada SVM.

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

Hasilnya:

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

Gambar 3.272 Hasil Soal 7 - 3

8. Nomor 8

- Menunjukkan informasi-informasi tree yang dibuat.

```
1 max_features_opts = range(5, 50, 5) #max_features_opts
...: sebagai variabel untuk membuat range 5,50,5
2 n_estimators_opts = range(10, 200, 20) #n_estimators_opts
...: sebagai variabel untuk membuat range 10,200,20
3 rf_params = np.empty((len(max_features_opts)*len(
...: n_estimators_opts),4), float) #rf_params sebagai
...: variabel untuk menjumlahkan yang sudah di tentukan
...: sebelumnya
4 i = 0
5 for max_features in max_features_opts: #pengulangan
...:     for n_estimators in n_estimators_opts: #pengulangan
```

```

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

```

Hasilnya:

Max features: 5, num estimators: 10, accuracy: 0.26 (+/- 0.02)	Max features: 25, num estimators: 100, accuracy: 0.46 (+/- 0.03)
Max features: 5, num estimators: 30, accuracy: 0.36 (+/- 0.02)	Max features: 30, num estimators: 10, accuracy: 0.33 (+/- 0.02)
Max features: 5, num estimators: 50, accuracy: 0.40 (+/- 0.03)	Max features: 30, num estimators: 30, accuracy: 0.40 (+/- 0.02)
Max features: 5, num estimators: 70, accuracy: 0.43 (+/- 0.03)	Max features: 30, num estimators: 50, accuracy: 0.44 (+/- 0.03)
Max features: 5, num estimators: 90, accuracy: 0.42 (+/- 0.02)	Max features: 30, num estimators: 70, accuracy: 0.44 (+/- 0.03)
Max features: 5, num estimators: 110, accuracy: 0.43 (+/- 0.02)	Max features: 30, num estimators: 90, accuracy: 0.44 (+/- 0.03)
Max features: 5, num estimators: 130, accuracy: 0.43 (+/- 0.03)	Max features: 30, num estimators: 110, accuracy: 0.45 (+/- 0.03)
Max features: 5, num estimators: 150, accuracy: 0.44 (+/- 0.03)	Max features: 30, num estimators: 130, accuracy: 0.46 (+/- 0.03)
Max features: 5, num estimators: 170, accuracy: 0.44 (+/- 0.03)	Max features: 30, num estimators: 150, accuracy: 0.46 (+/- 0.03)
Max features: 5, num estimators: 190, accuracy: 0.44 (+/- 0.03)	Max features: 30, num estimators: 170, accuracy: 0.46 (+/- 0.03)
Max features: 10, num estimators: 10, accuracy: 0.29 (+/- 0.01)	Max features: 30, num estimators: 190, accuracy: 0.46 (+/- 0.04)
Max features: 10, num estimators: 30, accuracy: 0.39 (+/- 0.01)	Max features: 35, num estimators: 10, accuracy: 0.41 (+/- 0.03)
Max features: 10, num estimators: 50, accuracy: 0.41 (+/- 0.03)	Max features: 35, num estimators: 30, accuracy: 0.41 (+/- 0.03)
Max features: 10, num estimators: 70, accuracy: 0.42 (+/- 0.03)	Max features: 35, num estimators: 50, accuracy: 0.42 (+/- 0.03)
Max features: 10, num estimators: 90, accuracy: 0.43 (+/- 0.03)	Max features: 35, num estimators: 70, accuracy: 0.43 (+/- 0.03)
Max features: 10, num estimators: 110, accuracy: 0.44 (+/- 0.03)	Max features: 35, num estimators: 90, accuracy: 0.44 (+/- 0.03)
Max features: 10, num estimators: 130, accuracy: 0.45 (+/- 0.04)	Max features: 35, num estimators: 110, accuracy: 0.45 (+/- 0.03)
Max features: 10, num estimators: 150, accuracy: 0.45 (+/- 0.03)	Max features: 35, num estimators: 130, accuracy: 0.45 (+/- 0.03)
Max features: 10, num estimators: 170, accuracy: 0.45 (+/- 0.03)	Max features: 35, num estimators: 150, accuracy: 0.46 (+/- 0.03)
Max features: 10, num estimators: 190, accuracy: 0.46 (+/- 0.03)	Max features: 35, num estimators: 170, accuracy: 0.46 (+/- 0.03)
Max features: 15, num estimators: 10, accuracy: 0.39 (+/- 0.02)	Max features: 35, num estimators: 190, accuracy: 0.46 (+/- 0.04)
Max features: 15, num estimators: 30, accuracy: 0.42 (+/- 0.03)	Max features: 40, num estimators: 10, accuracy: 0.41 (+/- 0.02)
Max features: 15, num estimators: 50, accuracy: 0.44 (+/- 0.02)	Max features: 40, num estimators: 30, accuracy: 0.43 (+/- 0.04)
Max features: 15, num estimators: 70, accuracy: 0.45 (+/- 0.02)	Max features: 40, num estimators: 50, accuracy: 0.44 (+/- 0.03)
Max features: 15, num estimators: 90, accuracy: 0.45 (+/- 0.02)	Max features: 40, num estimators: 70, accuracy: 0.45 (+/- 0.03)
Max features: 15, num estimators: 110, accuracy: 0.45 (+/- 0.02)	Max features: 40, num estimators: 90, accuracy: 0.45 (+/- 0.03)
Max features: 15, num estimators: 130, accuracy: 0.45 (+/- 0.02)	Max features: 40, num estimators: 110, accuracy: 0.45 (+/- 0.04)
Max features: 15, num estimators: 150, accuracy: 0.46 (+/- 0.02)	Max features: 40, num estimators: 130, accuracy: 0.46 (+/- 0.03)
Max features: 15, num estimators: 170, accuracy: 0.46 (+/- 0.02)	Max features: 40, num estimators: 150, accuracy: 0.46 (+/- 0.03)
Max features: 15, num estimators: 190, accuracy: 0.46 (+/- 0.02)	Max features: 40, num estimators: 170, accuracy: 0.46 (+/- 0.04)
Max features: 20, num estimators: 10, accuracy: 0.31 (+/- 0.02)	Max features: 45, num estimators: 10, accuracy: 0.34 (+/- 0.01)
Max features: 20, num estimators: 30, accuracy: 0.43 (+/- 0.03)	Max features: 45, num estimators: 30, accuracy: 0.43 (+/- 0.02)
Max features: 20, num estimators: 50, accuracy: 0.43 (+/- 0.03)	Max features: 45, num estimators: 50, accuracy: 0.43 (+/- 0.03)
Max features: 20, num estimators: 70, accuracy: 0.44 (+/- 0.03)	Max features: 45, num estimators: 70, accuracy: 0.44 (+/- 0.04)
Max features: 20, num estimators: 90, accuracy: 0.44 (+/- 0.03)	Max features: 45, num estimators: 90, accuracy: 0.45 (+/- 0.03)
Max features: 20, num estimators: 110, accuracy: 0.44 (+/- 0.03)	Max features: 45, num estimators: 110, accuracy: 0.45 (+/- 0.04)
Max features: 20, num estimators: 130, accuracy: 0.45 (+/- 0.03)	Max features: 45, num estimators: 130, accuracy: 0.45 (+/- 0.04)
Max features: 20, num estimators: 150, accuracy: 0.46 (+/- 0.03)	Max features: 45, num estimators: 150, accuracy: 0.45 (+/- 0.03)
Max features: 20, num estimators: 170, accuracy: 0.46 (+/- 0.03)	Max features: 45, num estimators: 170, accuracy: 0.46 (+/- 0.04)
Max features: 20, num estimators: 190, accuracy: 0.46 (+/- 0.04)	Max features: 45, num estimators: 190, accuracy: 0.46 (+/- 0.04)
Max features: 25, num estimators: 10, accuracy: 0.33 (+/- 0.02)	

Gambar 3.273 Hasil Soal 8 - 1

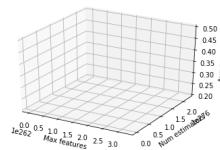
- Menunjukkan hasil dari plotting komponen informasi sehingga dapat kita baca sebagai grafik 3D.

```

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

```

Hasilnya:



Gambar 3.274 Hasil Soal 8 - 2

3.8.3 Penanganan Error

1. ScreenShoot Error

```
#fileNotFound: File 'D:/NetGuliah/Lingklat 3/Semester 6/Kecerdasan Buatan/Chapter 3/CUB_200_2011/attributes/image_attribute_labels.txt' does not exist
```

Gambar 3.275 File Not Found Error

2. Tuliskan Kode Error dan Jenis Error

- File Not Found Error

3. Cara Penanganan Error

- File Not Found Error

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

3.8.4 Bukti Tidak Plagiat



Gambar 3.276 Bukti Tidak Plagiat

3.8.5 Link Youtube

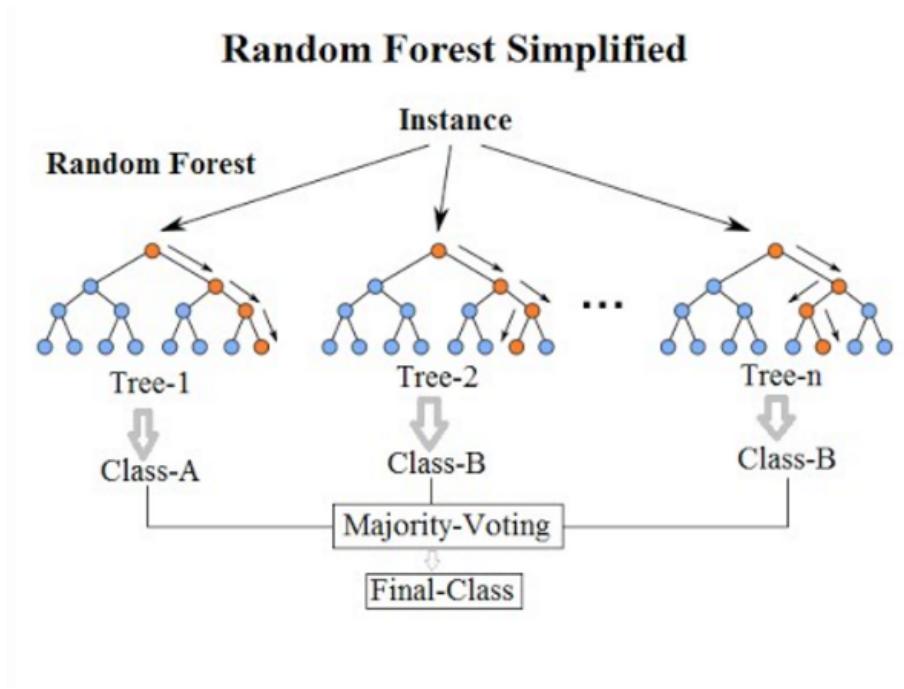
3.9 Advent Nopele Olansi Damiahan Sihite(1174089)

Chapter 3 - Prediksi dengan Random Forest

3.9.1 Teori

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

Random forest merupakan algoritma klasifikasi yang meliputi banyak decisions trees. random forest menggunakan bagging (salah satu algoritma machine learning) serta fitur pengacakan(randomness) ketika membangun setiap satu tree untuk mencoba membuat suatu hutan yang berisi pohon yang tidak berkorelasi



Gambar 3.277 contoh random forest

3.9.1.2 Jelaskan cara membaca dataset kasus dan artikan makna setiap file dan isi field masing masing file.

Dataset adalah suatu koleksi dari banyak data, biasanya dalam bentuk tabel dan penggambarannya kurang lebih mirip seperti table yang ada di database.

1. cara membaca dataset

```

1
2 import pandas as pd
3 dataset = pd.read_csv('F:/Poltekpos/D4 TI 3C/Semester 6/
  Kecerdasan Buatan/Github/Upload 18 Maret 2020 github tugas
  3/src/1174079/3/data.csv', sep=',')
4 dataset.head()
  
```

- import library panda dan namai sebagai pd agar memudahkan pemanggilan
- buat variable dataset dan panggil perintah untuk membaca file yang dimaksud
- berikut hasilnya

Index	ben Pertanian (20)	Unnamed: 1	Unnamed: 2	Unnamed: 3	Unnamed: 4	Unnamed: 5
0	No.	Komoditas	Tahun	nan	nan	Satuan
1	nan	nan	2016	2017	2018	nan
2	A	Pertanian Tanaman Pang...	nan	nan	nan	nan
3	I	Luas Panen Padi dan Pal...	nan	nan	nan	nan
4	1	Padi Sawah	675,00	469,00	490,00	Hektar
5	2	Padi Gogo	4,00	0	0	Hektar
6	3	Jagung panen Tua	224,00	161,00	299,00	Hektar
7	4	Jagung Panen Muda	232,00	433,00	349,00	Hektar
8	5	Kacang Tanah	60,00	39,00	1,90	Hektar
9	6	Ubi Kayu	62,00	48,00	23,00	Hektar
10	7	Ubi Jalar	94,00	57,00	4,90	Hektar
11	B	Pertanian Hortikultura	nan	nan	nan	nan
12	I	Luas Panen Tanaman Sayu...	nan	nan	nan	nan
13	1	Bawang Merah	284	373	379	Hektar
14	2	Bawang Putih	41	31	22	Hektar
15	3	Bawang Daun	281	285	373	Hektar
16	4	Kentang	466	490	475	Hektar
17	5	Kubis	410	404	257	Hektar
18	6	Kembang Kol	398	391	361	Hektar
19	7	Petsai/Sawi	311	314	342	Hektar
20	8	Wortel	371	402	505	Hektar
21	9	Kacang Merah	30	24	25	Hektar
		Kacang				

Gambar 3.278 contoh dataset

3.9.1.3 Jelaskan apa itu cross validation

Cross-validation (CV) adalah Merupakan metode statistic untuk meng-evaluasi dan membandingkan akurasi Learning algorithm dengan cara membagi dataset menjadi 2 bagian: Satu bagian digunakan untuk training model, Bagian yang lain untuk mem-validasi model Suatu dataset akan dibagi sesuai dengan banyaknya k, dan akan di test bergantian hingga seluruh bagian terpenuhi.

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

Score dari masing masing itu merupakan hasil dari tingkat keakuratan predikasi yang digunakan, jika menggunakan random forest anda mendapatkan tingkat keakuratan mencapai 44%, decision tree sekitar 27% dan svm 29%.

Kelas	Terklasifikasi Positif	Terklasifikasi Negatif
Positif	TP (True Positive)	FN (False Negative)
Negatif	FP (False Positive)	TN (True Negative)

Gambar 3.279 Contoh confusion matrix

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

1. TP adalah True Positive, jumlah data positif terdeteksi benar
2. TN adalah True Negative, jumlah data negatif terdeteksi benar.
3. FN adalah False Negative, jumlah data negatif terdeteksi salah.
4. FP adalah False Positive, jumlah data positif terdeteksi salah

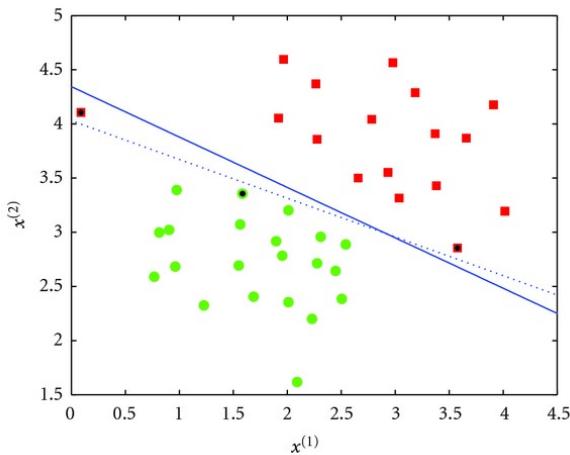
		Actual values	
		Apple	Orange
Predicted values	Apple	5 True positives	2 False positives
	Orange	3 False negatives	3 True negatives

Gambar 3.280 Contoh confusion matrix

Penjelasan dari gambar diatas adalah mesin mendeteksi 5 apel sebagai 5 apel dan 2 apel sisanya terdeteksi sebagai jeruk, lalu 3 jeruk terdeteksi sebagai apel dan 3 jeruk terdeteksi sebagai jeruk.

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

Voting di random forest kurang lebih sama seperti ketika kita ingin bertanya atau survey apakah suatu makanan masuk ke kategori enak atau tidak, jadi bisa dibilang voting itu adalah suara atau pendapat, yang akan diklasifikasikan.



Gambar 3.281 contoh random forest tentang kualitas barang yang semakin mahal semakin tahan lama

3.9.2 Praktek

3.9.2.1 Nomor 1

```

1 np.arange(1, 14, 4) # mengatur value awal dan akhir dilompati dengan beda 4
2
3 # In [2]
4 import numpy as np #import library
5 import matplotlib.pyplot as plt #import library

```

	Luas Panen Pertanian (2016-2018)			... Unnamed: 5
0	No.	...	Satuan	
1	NaN	...	NaN	
2	A	...	NaN	
3	I	...	NaN	
4	1	...	Hektar	

Gambar 3.282 Membuat Aplikasi pakai pandas

3.9.2.2 Nomor 2

```

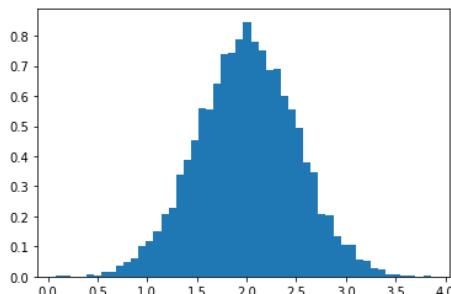
1 v = np.random.normal(mu,sigma,10000) #Mengisi histogram dengan data
2 # Plot a normalized histogram with 50 bins

```

```
In [2]: import numpy as np # imp
.... np.arange(1, 14, 4) # me
ditampati dengan beda 4
Out[2]: array([ 1,  5,  9, 13])
```

Gambar 3.283 Membuat Aplikasi pakai numpy

3.9.2.3 Nomor 3



Gambar 3.284 Membuat Aplikasi pakai matplotlib

3.9.2.4 Nomor 4

- Membaca dataset dengan variable budi. variable ini digunakan untuk mengisi data

```
1
2 budi.head() #Menampilkan data yang sudah dibaca tadi tapi
    cuman data paling atas
3
4 # In [5]
5 budi.shape #Menampilkan jumlah seluruh data , kolom-nya
```

```
In [9]: import pandas as pd #Melakukan import library numpy
menjadi pd
...
.... budi = pd.read_csv("D:/Pictures/
image_attribute_labels.txt",
    .... sep='\s+', header=None,
error_bad_lines=False, warn_bad_lines=False,
    .... usecols=[0,1,2], names=['imgid',
'atid', 'present']) #Membuat variable budi untuk membaca file
csv dari dataset, dengan ketentuan yang ada.
```

Gambar 3.285 Hasil 4 Bagian 1

- Menampilkan data teratas dari variable budi

```
1
2 # In [7]
```

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

Gambar 3.286 Hasil 4 Bagian 2

- menampilkan jumlah kolumn dan baris dari variable budi

```
In [11]: budi.shape #nya
Out[11]: (3677856, 3)
```

Gambar 3.287 Hasil 4 Bagian 3

- Mengubah kolumn jadi baris dengan fungsi pivot yang disimpan pada variabel budi2 yang datanya berasal dari variable budi sebelumnya.

```
1 # In [9]
```

```
In [12]: budi2 = budi.pivot(index='imgid', columns='attid',
                           values='present') #Membuat sebuah variabel baru dari fungsi
                           budi, dengan mengganti index menjadi kolom dan kolom menjadi
                           index
```

Gambar 3.288 Hasil 4 Bagian 4

- Menampilkan data teratas dari budi2

```
Out[13]:
attid 1 2 3 4 5 6 7 ... 306 307 308
309 310 311 312
imgid
1 0 0 0 0 1 0 0 ... 0 0 1
0 0 0 0 0 0 0 0 ... 0 0 0
2 0 0 0 0 0 0 0 ... 0 0 0
0 0 0 0 0 0 0 0 ... 0 0 0
3 0 0 0 0 1 0 0 0 ... 0 0 1
0 0 1 0 0 0 0 0 ... 0 0 0
4 0 0 0 0 0 1 0 0 0 ... 1 0 0
1 0 0 0 0 0 0 0 0 ... 1 0 0
5 0 0 0 0 1 0 0 0 ... 0 0 0
0 0 0 0 0 0 0 0 ... 0 0 0

[5 rows x 312 columns]
```

Gambar 3.289 Hasil 4 Bagian 5

- menampilkan jumlah kolumn dan baris variable budi2

```
1 # In [10]
```

```
Out[25]: (11788, 312)
```

Gambar 3.290 Hasil 4 Bagian 6

- membaca data csv dan membuat index baru bernama imgid

```
1
2 # In [11]
3 imglabels.shape #Menampilkan jumlah seluruh data , kolom-nya
```

```
In [26]: imglabels = pd.read_csv("D:/Pictures/
image_class_labels.txt",
...:
sep=',', header=None,
names=['imgid', 'label']) #Membaca data dimasukkan ke variable
imglabels
...:
...: imglabels = imglabels.set_index('imgid') #Variable
imglabels dan set index (imgid)

In [27]: imglabels = pd.read_csv("D:/Pictures/
image_class_labels.txt",
...:
sep=',', header=None,
names=['imgid', 'label']) #Membaca data dimasukkan ke variable
imglabels
...:
...: imglabels = imglabels.set_index('imgid') #Variable
imglabels dan set index (imgid)
```

Gambar 3.291 Hasil 4 Bagian 7

- Menampilkan data teratas

```
1 df = df.sample(frac=1) #Variabel df sebagai sample dengan
ketentuan frac=1
```

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

Gambar 3.292 Hasil 4 Bagian 8

- mengeluarkan output jumlah kolumn dan baris

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

Out[31]: (11788, 1)

Gambar 3.293 Hasil 4 Bagian 9

- Menggabungkan variable budi2 dan imglabels karena isi yang mirip sehingga bisa dikategorikan sebagai supervised learning.

```
1 # In [14]
2 df_att.head() #Menampilkan data yang sudah dibaca tadi tapi
    cuman data paling atas
```

```
In [32]: df = budi2.join(imglabels) #1
fungsi join dari data budi2 ke variabel
....: df = df.sample(frac=1) #Variabel
dengan ketentuan frac=1
```

Gambar 3.294 Hasil 4 Bagian 10

- membuat columnm

```
1 df_label.head() #Menampilkan data yang sudah dibaca tadi tapi
    cuman data paling atas
```

```
In [33]: df_att = df.iloc[:, :312] #Membuat kolom dengan
ketentuan 312
...: df_label = df.iloc[:, 312:] #Membuat kolom dengan
ketentuan 312
```

Gambar 3.295 Hasil 4 Bagian 11

- Mengecek data teratas variable df_att.

```
1 df_train_label = df_label[:8000] #Data akan dibagi dari 8000
row pertama menjadi data training dan sisanya adalah data
testing
```

	1	2	3	4	5	6	7	...	306	307	308
309	310	311	312								
imgid	0	1	0	0	0	0	0	...	0	0	1
5911	0	0	0	0	0	0	0	...	0	0	0
68	0	0	0	0	0	1	0	0	...	1	0
1	0	0	0	0	0	0	0	1	...	0	0
10028	0	0	0	0	0	0	0	0	1	...	0
5002	0	0	1	0	0	0	0	0	...	0	0
10530	0	0	0	0	0	0	0	1	...	0	0
0	0	0	1								

15 rows x 312 columns

Gambar 3.296 Hasil 4 Bagian 12

- Mengeceki data teratas dari df_label.

```
In [36]: df_label.head()
tadi tapi cuman data pal
Out[36]:
label
imgid
5911      101
68          2
10028     171
5002      86
10530     179
```

Gambar 3.297 Hasil 4 Bagian 13

- Membagi 8000 baris awal sebagai data training dan yang lainnya sebagai data testing.

```
1
2 # In [17]
3 from sklearn.ensemble import RandomForestClassifier #Import
fungsi randomforestclassifier
4 clf = RandomForestClassifier(max_features=50, random_state=0,
n_estimators=100) #clf sebagai variabel untuk klasifikasi
random forest
```

```

5
6 # In [18]
7 clf.fit(df_train_att, df_train_label) #Variabne clf untuk
    fit yaitu menjadi data training

```

```

#6000 row pertama menjadi data training dan sisanya adalah data
testing
....: df_train_label = df_label[:8000] #Data akan dibagi
dari 8000 row pertama menjadi data training dan sisanya adalah
data testing
....: df_test_att = df_att[8000:] #Berbalik dari sebelumnya
data akan dibagi mulai dari 8000 row pertama menjadi data
training dan sisanya adalah data testing
....: df_test_label = df_label[8000:] #Berbalik dari
sebelumnya data akan dibagi mulai dari 8000 row pertama
menjadi data training dan sisanya adalah data testing
....:
....: df_train_label = df_train_label['label'] #Menambahkan
label
....: df_test_label = df_test_label['label'] #Menambahkan
label

```

Gambar 3.298 Hasil 4 Bagian 14

- memanggil class RandomForestClassifier. dengan maks kolumn 50

```

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

```

```

In [38]: from sklearn.ensemble import RandomForestClassifier
#Import fungsi randomforestclassifier
....: clf = RandomForestClassifier(max_features=50,
random_state=0, n_estimators=100) #clf sebagai variabel untuk
klasifikasi random forest

```

Gambar 3.299 Hasil 4 Bagian 15

- output hasil prediksi dari Random Forest.

```

In [39]: clf.fit(df_train_att, df_train_label) #Variabne
untuk fit yaitu menjadi data training
Out[39]:
RandomForestClassifier(bootstrap=True, class_weight=None,
criterion='gini',
            max_depth=None, max_features=50,
            max_leaf_nodes=None,
            min_impurity_decrease=0.0,
            min_impurity_split=None,
            min_samples_leaf=1,
            min_samples_split=2,
            min_weight_fraction_leaf=0.0,
            n_estimators=100,
            n_jobs=None, oob_score=False,
            random_state=0, verbose=0,
            warm_start=False)

```

Gambar 3.300 Hasil 4 Bagian 16

- Memperlihatkan score akurasi.

```

In [40]: print(clf.predict(df_train_att.head()))
yang di sudah prediksi dari training tetapi hanya
data paling atas
[101  2 171  86 179]

```

Gambar 3.301 Hasil 4 Bagian 17

- Menampilkan tingkat akurasi

```
In [41]: clf.score(df_test_att, df_test_label)
clf sebagai testing yang sudah di training tadi
Out[41]: 0.45248152059134106
```

Gambar 3.302 Hasil 4 Bagian 18

3.9.2.5 Nomor 5

```
1 # In [23]
2 import matplotlib.pyplot as plt #Mengimport library matplotlib
    sebagai plt
3 import itertools #Mengimport library itertools
```

```
In [42]: from sklearn.metrics import confusion_matrix
#Mengimport Confusion Matrix
...: pred_labels = clf.predict(df_test_att) #Membuat
variable pred_labels dari data testing
...: cm = confusion_matrix(df_test_label, pred_labels)
sebagai variabel data label
```

Gambar 3.303 Hasil 5 Bagian 1

```
1 title='Confusion matrix',
```

```
In [43]: cm #Memunculkan data label berbentuk array
Out[43]:
array([[ 2,  3,  3, ...,  0,  0,  0],
       [ 0, 11,  0, ...,  0,  1,  0],
       [ 0,  0, 12, ...,  0,  0,  0],
       ...,
       [ 0,  0,  0, ...,  2,  0,  0],
       [ 0,  0,  0, ...,  0, 10,  0],
       [ 0,  0,  0, ...,  0,  0, 12]], dtype=int64)
```

Gambar 3.304 Hasil 5 Bagian 2

```
1 if normalize:
2     cm = cm.astype('float') / cm.sum(axis=1)[:, np.newaxis]
3     print("Normalized confusion matrix") #Jika normalisasi
sebagai ketentuan yang ada maka print normalized confusion
matrix
4 else:
5     print('Confusion matrix, without normalization') #Jika
tidak memenuhi kondisi if maka pring else
6
7 print(cm) #Print data cm
8
9 plt.imshow(cm, interpolation='nearest', cmap=cmap) #plt
sebagai fungsi untuk membuat plot
10 plt.title(title) #Membuat title pada plot
11 #plt.colorbar()
```

```

12 tick_marks = np.arange(len(classes)) #Membuat marks pada plot
13 plt.xticks(tick_marks, classes, rotation=90) #Membuat ticks
14 pada x
15 plt.yticks(tick_marks, classes) #Membuat ticks pada y
16
17 fmt = '.2f' if normalize else 'd' #fmt sebagai normalisasi
18 thresh = cm.max() / 2. #Variable thresh menambil data max
19 pada cm kemudian dibagi 2
20
21 plt.tight_layout() #Mengatur layout pada plot
22 plt.ylabel('True label') #Menambahkan nama label pada sumbu y
23 plt.xlabel('Predicted label') #Menambahkan nama label pada
24 sumbu x
25
26 # In [24]
27 birds = pd.read_csv("D:/ Pictures/classes.txt",
28                      sep='\s+', header=None, usecols=[1], names=[],
29                      birdname']) #membaca csv dengan ketentuan nama birdname
30 birds = birds['birdname'] #nama birds dengan ketentuan birdname
31 birds #Menampilkan data birds

```

```

In [44]: import matplotlib.pyplot as plt #Mengimport library
matplotlib sebagai plt
.... import iterools #Mengimport library iterools
.... def plot_confusion_matrix(cm, classes,
....                            normalize=False,
....                            title='Confusion matrix',
....                            cmap=plt.cm.Blues):
#Membuat fungsi dengan ketentuan data yang ada pada cm
.... if normalize:
....     cm = cm.astype('float') / cm.sum(axis=1)[:, np.newaxis]
....     print("Normalized confusion matrix") #Jika
normalize sebagai ketentuan yang ada maka print normalized
confusion matrix
.... else:
....     print('Confusion matrix, without
normalization') #Jika tidak memenuhi kondisi if maka print
else
.... print(cm) #Print data cm
.... plt.imshow(cm, interpolation='nearest',
cmap=cmap) #plt sebagai fungsi untuk membuat plot
.... plt.title(title) #Membuat title pada plot
.... plt.colorbar()
.... tick_marks = np.arange(len(classes)) #Membuat
marks pada plot
.... plt.xticks(tick_marks, classes, rotation=90)
#Membuat ticks pada x

```

Gambar 3.305 Hasil 5 Bagian 3

```

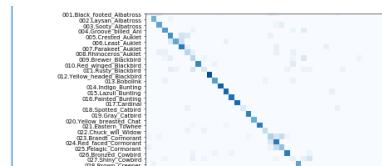
1 import numpy as np #Mengimport library numpy sebagai np
2 np.set_printoptions(precision=2) #np sebagai variabel yang
    membuat set precision=2
3 plt.figure(figsize=(30,30), dpi=50) #Plot sebagai figure dengan
    ketentuan size 60,60 dan dpi 300
4 plot_confusion_matrix(cm, classes=birds, normalize=True) #Data cm
    dan clas birds dibuat sebagai plot

```

```
Out[45]:
0      001.Black_footed_Albatross
1      002.Laysan_Albatross
2      003.Sooty_Albatross
3      004.Groove_billed_Ani
4      005.Crested_Auklet
...
195     196.House_Wren
196     197.Marsh_Wren
197     198.Rock_Wren
198     199.Winter_Wren
199    200.Common_Yellowthroat
Name: birdname, Length: 200, dtype: object
```

Gambar 3.306 Hasil 5 Bagian 4

```
1
2
3 # In [26]: Mencoba dengan metode Decission Tree dan SVM
4 from sklearn import tree #Mengimport library tree
5 clftree = tree.DecisionTreeClassifier() #clftree sebagai variabel
   untuk decision tree
6 clftree.fit(df_train_att, df_train_label) #Mengatur data training
```



Gambar 3.307 Hasil 5 Bagian 5

3.9.2.6 Nomor 6

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

| **Out[49]:** 0.2732312565997888

Gambar 3.308 Hasil 6 Bagian 1

```
1 scores = cross_val_score(clf, df_train_att, df_train_label, cv=5)
   #Membuat variable scores sebagai variabel prediksi dari data
   training
2 print("Accuracy: %.2f (+/- %.2f)" % (scores.mean(), scores.std()
   () * 2)) #Print data scores dengan ketentuan akurasi
3
4 # In [29]
```

```
Out[50]: 0.28880675818373813
```

Gambar 3.309 Hasil 6 Bagian 2

3.9.2.7 Nomor 7

```
1 # In [30]
2 scoresSVM = cross_val_score(clfsvm, df_train_att, df_train_label,
3                             cv=5) #Membuat variable data training
```

```
In [51]: from sklearn.model_selection import cross_val_score
#Impor cross_val_score
...: scores = cross_val_score(clf, df_train_att,
df_train_label, cv=5) #Membuat variable scores sebagai
variabel prediksi dari data training
...: print("Accuracy: %.2f (+/- %.2f)" % (scores.mean(),
scores.std() * 2)) #print data scores dengan ketentuan akurasi
Accuracy: 0.44 (+/- 0.01)
```

Gambar 3.310 Hasil 7 Bagian 1

```
1 # In [31]: Pengamatan komponen informasi
2 max_features_opts = range(5, 50, 5) #Variable max_features_opts
sebagai variabel untuk membuat range 5,50,5
```

```
In [52]: scorestree = cross_val_score(clftree, df_train_att,
df_train_label, cv=5) #Membuat variable predict menggunakan
score dan metoda tree
...: print("Accuracy: %.2f (+/- %.2f)" %
(scorestree.mean(), scorestree.std() * 2)) #Menampilkan dengan
ketentuan yang ada
Accuracy: 0.26 (+/- 0.02)
```

Gambar 3.311 Hasil 7 Bagian 2

```
1 i = 0
2 for max_features in max_features_opts: #Perulangan
```

| Accuracy: 0.27 (+/- 0.01)

Gambar 3.312 Hasil 7 Bagian 3

3.9.2.8 Nomor 8

```
1 scores = cross_val_score(clf, df_train_att,
df_train_label, cv=5) #Variable scores sebagai variabel
training
2 rf_params[i,0] = max_features #index 0
3 rf_params[i,1] = n_estimators #index 1
4 rf_params[i,2] = scores.mean() #index 2
5 rf_params[i,3] = scores.std() * 2 #index 3
6 i += 1 #Dengan ketentuan i += 1
```

```

7     print("Max features: %d, num estimators: %d, accuracy:
8         %0.2f (+/- %0.2f)" % (max_features, n_estimators,
9             scores.mean(), scores.std() * 2))
    #Print hasil pengulangan yang sudah ditentukan
10
11 # In[32]
12 import matplotlib.pyplot as plt #Mengimport library matplotlib
13     sebagai plt
14 from mpl_toolkits.mplot3d import Axes3D #Mengimport axes3D untuk
15     menampilkan plot 3 dimensi
16 from matplotlib import cm #Memanggil data cm yang sudah tersedia
17 fig = plt.figure() #Menghasilkan plot sebagai figure
18 fig.clf() #Figure di ambil dari clf
19 ax = fig.gca(projection='3d') #ax sebagai projection 3d

```

```

Max features: 5, num estimators: 10, accuracy: 0.27 (+/- 0.01)
Max features: 5, num estimators: 30, accuracy: 0.36 (+/- 0.03)
Max features: 5, num estimators: 50, accuracy: 0.39 (+/- 0.02)
Max features: 5, num estimators: 70, accuracy: 0.40 (+/- 0.01)
Max features: 5, num estimators: 90, accuracy: 0.41 (+/- 0.01)

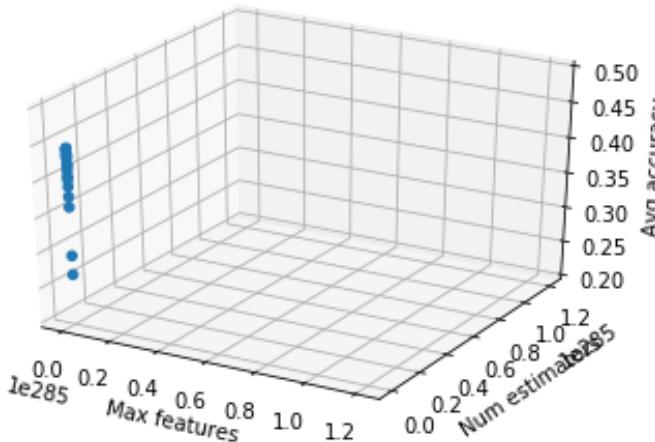
```

Gambar 3.313 Hasil 8 Bagian 1

```

1 y = rf_params[:,1] #y sebagai index 1
2 z = rf_params[:,2] #z sebagai index 2
3 ax.scatter(x, y, z) #Membuat plot scatter x y z
4 ax.set_zlim(0.2, 0.5) #Set zlim dengan ketentuan yang ada
5 ax.set_xlabel('Max features') #Memberikan nama label x
6 ax.set_ylabel('Num estimators') #Memberikan nama label y
7 ax.set_zlabel('Avg accuracy') #Memberikan nama label z
8 plt.show() #Print hasil plot yang sudah dibuat.

```



Gambar 3.314 Hasil 8 Bagian 2

3.9.3 Penanganan Error

NameError: name 'dataset' is not defined : Biasanya karena typo, cek lagi penulisan kode

3.9.3.1 Tuliskan Kode Error dan Jenis Error NameError: name 'dataset' is not defined

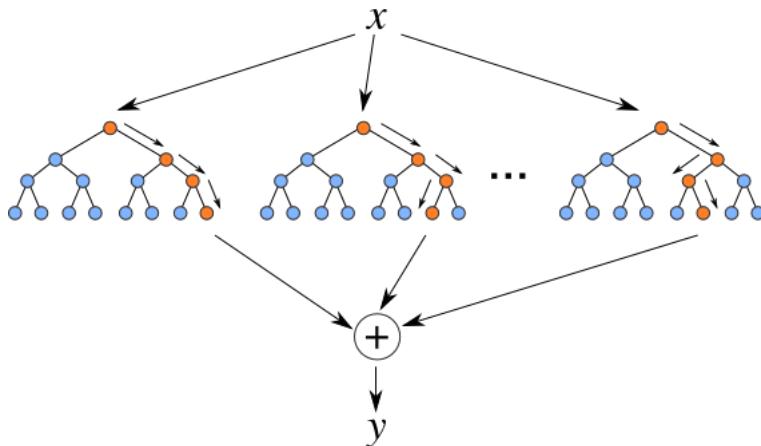
3.9.3.2 Cara Penanganan Error Pastikan untuk tidak menulis function secara typo dan ceklagi

3.10 Muhammad Reza Syachrani (1174084)

3.10.1 Teori

3.10.1.1 Apa Itu Random Forest

Random Forest merupakan sebuah algoritma yang digunakan pada klasifikasi data dalam jumlah yang besar. Klasifikasi pada random forest dilakukan dengan penggabungan dicision tree dengan melakuakn training terhadap sempel data yang dimiliki. Dengan menggunakan dicision tree yang banyak akan mempengaruhi akurasi yang akan didapatkan menjadi lebih baik. Penentuan klasifikasi dengan random forest sesuai dengan hasil voting dari tree yang terbentuk.



Gambar 3.315 Random Forest

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

Dataset adalah kumpulan data. Paling umum satu data set sesuai dengan isi tabel database tunggal, atau matriks data statistik tunggal, dimana setiap

kolom tabel mewakili variabel tertentu, dan setiap baris sesuai dengan anggota tertentu dari dataset yang dipertanyakan. cara membaca dataset :

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

3.10.1.3 Cross Validation

Cross Validation adalah teknik validasi model untuk menilai suatu hasil statistik analisis yang akan menggeneralisasi kumpulan data independen. Teknik ini digunakan untuk melakukan prediksi model dan memperkirakan seberapa akurat sebuah model prediktif ketika dijalankan dalam praktiknya. Tujuan dari cross validation adalah untuk mendefinisikan dataset untuk menguji model dalam tahap pelatihan (yaitu, validasi data), dalam rangka untuk membatasi masalah seperti terjadinya overfitting, memberikan wawasan tentang bagaimana model dapat menggeneralisasi independen dataset.

3.10.1.4 Arti score 44 % pada random forest, 27% pada decision tree dan 29 % dari SVM.

Dimana Score 44 % didapatkan dari hasil pengelahan dataset jenis burung. Dimana akan dilakukan proses pembagian data testing dan data training lalu diproses dan menghasilkan score sebanyak 44 % dimana menjelaskan bahwa score tersebut digunakan sebagai pembanding dalam tingkat keakuratannya. Pada decision tree akan memperoleh data lebih kecil yaitu sebanyak 27 % hal ini dikarenakan data yang diolah menggunakan decision tree dibagi menjadi beberapa tree dan lalu disimpulkan untuk mendapatkan data yang akurat. Pada SVM akan memperoleh score sebanyak 29 % hal ini dikarenakan data yang dimiliki masih bernilai netral sehingga tingkat keakuratannya masih belum jelas.

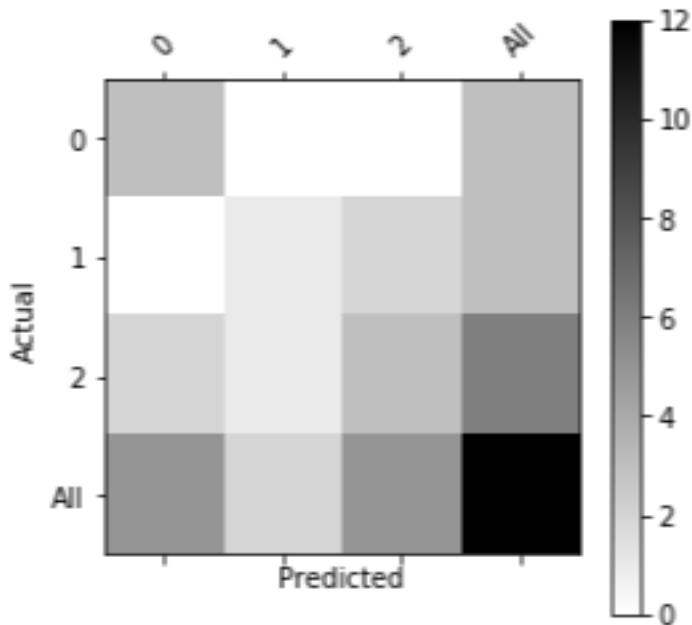
3.10.1.5 Cara membaca confusion matriks dan contohnya memakai gambar atau ilustrasi sendiri.

Perhitungan Confusion Matriks dapat dilakukan dengan cara dibawah ini.

- Import librari Pandas, Matplotlib, dan Numpy.

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

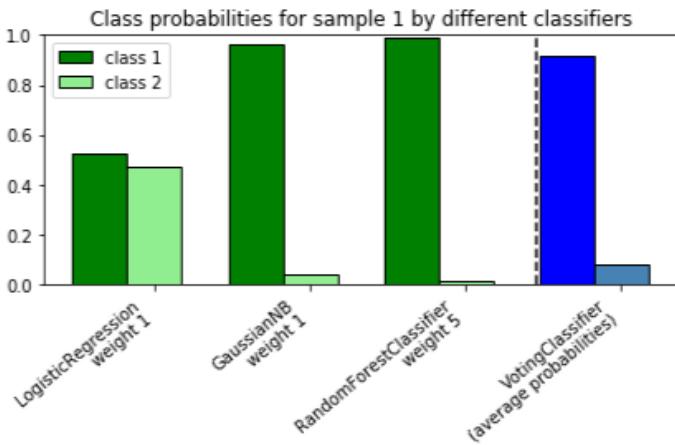
```
1 # -*- coding: utf-8 -*-
2 """
3 Created on Sat Mar 21 09:26:08 2020
4
5 @author: rezas
6 """
7 import numpy as np
8 import matplotlib.pyplot as plt
9 import pandas as pd
10 y_actu = pd.Series([2, 0, 2, 2, 0, 1, 1, 2, 2, 0, 1, 2], name
11                   ='Actual')
12 y_pred = pd.Series([0, 0, 2, 1, 0, 2, 1, 0, 2, 0, 2, 2], name
13                   ='Predicted')
14 df_confusion = pd.crosstab(y_actu, y_pred)
15 df_confusion = pd.crosstab(y_actu, y_pred, rownames=[ 'Actual',
16                             ], colnames=[ 'Predicted'], margins=True)
17 def plot_confusion_matrix(df_confusion , title='Confusion
18                           matrix' , cmap=plt.cm.gray_r):
19     plt.matshow(df_confusion , cmap=cmap) # imshow
20     #plt.title(title)
21     plt.colorbar()
22     tick_marks = np.arange(len(df_confusion.columns))
23     plt.xticks(tick_marks , df_confusion.columns , rotation=45)
24     plt.yticks(tick_marks , df_confusion.index)
25     #plt.tight_layout()
26     plt.ylabel(df.confusion.index.name)
27     plt.xlabel(df.confusion.columns.name)
28 plot_confusion_matrix(df_confusion)
29 plt.show()
```



Gambar 3.316 Confusion Matriks

3.10.1.6 Apa itu voting pada random forest

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



Gambar 3.317 Voting Random Matriks

3.10.2 Praktek

3.10.2.1 Nomor 1

```

1 import pandas as pd
2 nilai = {"mat": [70,80,75,90,90] , "ipa": [80,80,85,70,85] , "ips":
3   : [90,85,90,80,80]}
4 df = pd.DataFrame(data=nilai)
5 print("Original DataFrame")
6 print(df)
7 print("Baris Untuk nilai matematika dengan value 90")
7 print(df.loc[df["mat"] == 90])

```

Original DataFrame

	mat	ipa	ips
0	70	80	90
1	80	80	85
2	75	85	90
3	90	70	80
4	90	85	80

Baris Untuk nilai matematika dengan value 90

	mat	ipa	ips
3	90	70	80
4	90	85	80

Gambar 3.318 Membuat Aplikasi pakai pandas

3.10.2.2 Nomor 2

```

1 import numpy as np
2 mat = np.array([70, 80, 75, 90, 90])
3 print ("Matematika : " , mat )
4 ipa = np.array ([80, 80, 85, 70, 85])
5 print ("IPA: " , ipa )
6 ips = np.array ([90, 85, 90, 80, 80])
7 print ("IPS: " , ips )
8 print ("Data Yang Sama Dari Nilai Matematika dan IPA Adalah :")
9 print (np.intersect1d (mat , ipa))
10 print ("Data Yang Sama Dari Nilai Matematika dan IPS Adalah :")
11 print (np.intersect1d (mat , ips))

```

```

Matematika : [70 80 75 90 90]
IPA: [80 80 85 70 85]
IPS: [90 85 90 80 80]
Data Yang Sama Dari Nilai Matematika dan IPA Adalah :
[70 80]
Data Yang Sama Dari Nilai Matematika dan IPS Adalah :
[80 90]

```

Gambar 3.319 Membuat Aplikasi pakai numpy

3.10.2.3 Nomor 3

```

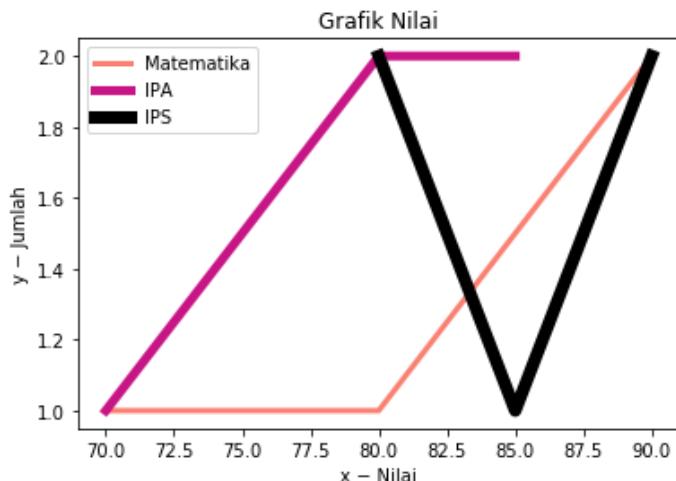
1 import matplotlib.pyplot as plt
2 # line 1 points
3 xmat = [70, 75, 80, 90]
4 ymat = [1 ,1 ,1 ,2]
5 # line 2 points
6 xipa = [70, 80, 85]
7 yipa = [1 ,2 ,2]
8
9 xips = [80, 85, 90]
10 yips = [2 ,1 ,2]
11
12 # Set the x axis label of the current axis .
13 plt.xlabel("x      Nilai")
14 # Set the y axis label of the current axis .
15 plt.ylabel("y      Jumlah")
16 # Set a title
17 plt.title("Grafik Nilai")
18 # Display the figure .
19 plt.plot(xmat ,ymat , color="salmon" , linewidth = 3, label = "
    Matematika")
20 plt.plot (xipa ,yipa , color="mediumvioletred" , linewidth = 5,
    label = "IPA")
21 plt.plot (xips ,yips , color="black" , linewidth = 7, label = "
    IPS")

```

```

22 # show a legend on the plot
23 plt.legend()
24 plt.show()

```



Gambar 3.320 Membuat Aplikasi pakai matplotlib

3.10.2.4 Nomor 4

```

1 import pandas as pd #Import library numpy menjadi pd
2
3 imgatt = pd.read_csv("C:/Users/rezas/Downloads/CUB_200_2011/
4   attributes/image_attribute_labels.txt",
5   sep='|', header=None, error_bad_lines=
6   False, warn_bad_lines=False,
7   usecols=[0,1,2], names=['imgid', 'attid', 'present']) #Membuat variable imgatt untuk membaca file csv
8 dari dataset
9
10 imgatt.head() #Menampilkan data paling atas yang sudah dibaca
11 imgatt.shape #Menampilkan jumlah seluruh data kolom
12 imgatt2 = imgatt.pivot(index='imgid', columns='attid', values='
13   present') #Membuat sebuah variabel baru dari fungsi imgatt,
14   dengan mengganti index menjadi kolom dan kolom menjadi index
15
16 imglabels = pd.read_csv("C:/Users/rezas/Downloads/CUB_200_2011/
17   image_class_labels.txt",
18   sep=' ', header=None, names=['imgid', 'label']) #baca data csv dan dimasukkan ke variable imglabels

```

```
18
19 imglabels = imglabels.set_index('imgid') #Variable imglabels dan
    set index (imgid)
20
21 imglabels.head() #Menampilkan data yang sudah dibaca tadi tapi
    cuman data paling atas
22
23 imglabels.shape #Menampilkan jumlah seluruh data, kolom-nya
24
25 df = imgatt2.join(imglabels) #Varibel df dimasukkan fungsi join
    dari data imgatt2 ke variabel imglabels
26 df = df.sample(frac=1) #Variabel df sebagai sample dengan
    ketentuan frac=1
27
28 df_att = df.iloc[:, :312] #Membuat kolom dengan ketentuan 312
29 df_label = df.iloc[:, 312:] #Membuat kolom dengan ketentuan 312
30
31 df_att.head() #Menampilkan data yang sudah dibaca hanya data
    paling atas
32
33 df_label.head() #Menampilkan data yang sudah dibaca hanya data
    paling atas
34
35 df_train_att = df_att[:8000] #Data akan dibagi dari 8000 row
    pertama menjadi data training dan sisanya adalah data testing
36 df_train_label = df_label[:8000] #Data akan dibagi dari 8000 row
    pertama menjadi data training dan sisanya adalah data testing
37 df_test_att = df_att[8000:] #data akan dibagi mulai dari 8000 row
    terakhir menjadi data training dan sisanya adalah data
    testing
38 df_test_label = df_label[8000:] # data akan dibagi mulai dari
    8000 row terakhir menjadi data training dan sisanya adalah
    data testing
39
40 df_train_label = df_train_label['label'] #Menambahkan label
41 df_test_label = df_test_label['label'] #Menambahkan label
42
43 from sklearn.ensemble import RandomForestClassifier #Import
    fungsi randomforestclassifier
44 clf = RandomForestClassifier(max_features=50, random_state=0,
    n_estimators=100) #clf sebagai variabel untuk klafisikasi
    random forest
45
46 clf.fit(df_train_att, df_train_label) #variable clf untuk fit
    yaitu menjadi data training
47 print(clf.predict(df_train_att.head())) #menampiulkan clf yang di
    sudah prediksi dari training yang data paling atas
48
49 clf.score(df_test_att, df_test_label) #Memunculkan clf sebagai
    testing yang sudah di training
```

```
[ 83 141 14 109 194]
Out[6]: 0.44693769799366423
```

Gambar 3.321 Hasil Random Forest

3.10.2.5 Nomor 5

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

Out[7]:

```
array([[ 7,  1,  4, ...,  0,  0,  0],
       [ 0, 13,  0, ...,  0,  0,  0],
       [ 1,  0,  5, ...,  0,  0,  0],
       ...,
       [ 0,  0,  0, ...,  3,  0,  0],
       [ 0,  0,  0, ...,  0,  7,  0],
       [ 0,  0,  0, ...,  0,  0, 17]], dtype=int64)
```

Gambar 3.322 Hasil Confusion Matrix

3.10.2.6 Nomor 6

```
1 df = pd.DataFrame(data=nilai)
2 print("Original DataFrame")
3 print(df)
4 print("Baris Untuk nilai matematika dengan value 90")
5 print(df.loc[df["mat"] == 90])
6
7 # In [2]:
8
9 import numpy as np
10 mat = np.array([70, 80, 75, 90, 90])
11 print ("Matematika : " , mat )
12 ipa = np.array ([80, 80, 85, 70, 85])
13 print ("IPA: " , ipa )
14 ips = np.array ([90, 85, 90, 80, 80])
15 print ("IPS: " , ips )
16 print ("Data Yang Sama Dari Nilai Matematika dan IPA Adalah :")
17 print (np.intersect1d (mat , ipa))
```

```
18 print ("Data Yang Sama Dari Nilai Matematika dan IPS Adalah :")
19 print (np.intersect1d (mat , ips))
20
21 # In [3]:
22
23 import matplotlib.pyplot as plt
24 # line 1 points
25 xmat = [70, 75, 80, 90]
26 ymat = [1 ,1 ,1 ,2]
27 # line 2 points
28 xipa = [70, 80, 85]
29 yipa = [1 ,2 ,2]
30
31 xips = [80, 85, 90]
32 yips = [2 ,1 ,2]
33
34 # Set the x axis label of the current axis .
35 plt.xlabel("x      Nilai")
36 # Set the y axis label of the current axis .
37 plt.ylabel("y      Jumlah")
38 # Set a title
39 plt.title("Grafik Nilai")
40 # Display the figure .
41 plt.plot(xmat ,ymat , color="salmon" , linewidth = 3, label = "Matematika")
42 plt.plot (xipa ,yipa , color="mediumvioletred" , linewidth = 5,
           label = "IPA")
43 plt.plot (xips ,yips , color="black" , linewidth = 7, label = "IPS")
44 # show a legend on the plot
45 plt.legend()
46 plt.show()
47
48 # In [4]:
49
50 import pandas as pd #Import library numpy menjadi pd
51
52 imgatt = pd.read_csv("C:/Users/rezas/Downloads/CUB_200_2011/
53                         attributes/image_attribute_labels.txt",
54                         sep='\s+' , header=None, error_bad_lines=
55                         False , warn_bad_lines=False ,
56                         usecols=[0,1,2] , names=['imgid' , 'attid' , 'present'])
57 #Membuat variable imgatt untuk membaca file csv
58 dari dataset
59
60 imgatt.head() #Menampilkan data paling atas yang sudah dibaca
61
62 imgatt.shape #Menampilkan jumlah seluruh data kolom
63 imgatt2 = imgatt.pivot(index='imgid' , columns='attid' , values='
64 present') #Membuat sebuah variabel baru dari fungsi imgatt ,
dengan mengganti index menjadi kolom dan kolom menjadi index
65
66 imgatt2.head() #Menampilkan data paling atas yang sudah dibaca
67 imgatt2.shape #Menampilkan jumlah seluruh data kolom
```

```
65 imglabels = pd.read_csv("C:/ Users/rezas/Downloads/CUB_200_2011/
66     image_class_labels.txt",
67     sep=' ', header=None, names=['imgid', 'label'])
68 #baca data csv dan dimasukkan ke variable imglabels
69
70 imglabels.set_index('imgid') #Variable imglabels dan
71     set index (imgid)
72
73 imglabels.head() #Menampilkan data yang sudah dibaca tadi tapi
74     cuman data paling atas
75
76 imglabels.shape #Menampilkan jumlah seluruh data, kolom-nya
77
78 df = imgatt2.join(imglabels) #Varibel df dimasukkan fungsi join
79     dari data imgatt2 ke variabel imglabels
80 df = df.sample(frac=1) #Variabel df sebagai sample dengan
81     ketentuan frac=1
82
83 df_att = df.iloc[:, :312] #Membuat kolom dengan ketentuan 312
84 df_label = df.iloc[:, 312:] #Membuat kolom dengan ketentuan 312
85
86 df_att.head() #Menampilkan data yang sudah dibaca hanya data
87     paling atas
88
89 df_label.head() #Menampilkan data yang sudah dibaca hanya data
90     paling atas
91
92 df_train_att = df_att[:8000] #Data akan dibagi dari 8000 row
93     pertama menjadi data training dan sisanya adalah data testing
94 df_train_label = df_label[:8000] #Data akan dibagi dari 8000 row
95     pertama menjadi data training dan sisanya adalah data testing
96 df_test_att = df_att[8000:] #data akan dibagi mulai dari 8000 row
97     terakhir menjadi data training dan sisanya adalah data
98     testing
99
100 df_test_label = df_label[8000:] # data akan dibagi mulai dari
      8000 row terakhir menjadi data training dan sisanya adalah
      data testing
101
102 df_train_label = df_train_label['label'] #Menambahkan label
103 df_test_label = df_test_label['label'] #Menambahkan label
104
105 from sklearn.ensemble import RandomForestClassifier #Import
106     fungsi randomforestclassifier
107 clf = RandomForestClassifier(max_features=50, random_state=0,
108     n_estimators=100) #clf sebagai variabel untuk klafisikasi
109     random forest
110
111 clf.fit(df_train_att, df_train_label) #variable clf untuk fit
112     yaitu menjadi data training
113 print(clf.predict(df_train_att.head())) #menampiulkan clf yang di
114     sudah prediksi dari training yang data paling atas
115
116 clf.score(df_test_att, df_test_label) #Memunculkan clf sebagai
117     testing yang sudah di training
118
119 # In [5]:
```

```

101
102 from sklearn.metrics import confusion_matrix #Mengimport
103     Confusion Matrix
103 pred_labels = clf.predict(df_test_att) #Membuat variable
104     pred_labels dari data testing
104 cm = confusion_matrix(df_test_label, pred_labels) #cm sebagai
105     variabel data label
105 cm
106 # In [6]:
107 from sklearn import tree #Mengimport tree
108 clftree = tree.DecisionTreeClassifier() #clftree sebagai variabel
109     untuk decision tree
109 clftree.fit(df_train_att, df_train_label) #Mengatur data training
110 clftree.score(df_test_att, df_test_label) #Mengatur data testing

```

```

In [9]: from sklearn import tree #Mengimport tree
...: clftree = tree.DecisionTreeClassifier() #clftree sebagai variabel untuk
decision tree
...: clftree.fit(df_train_att, df_train_label) #Mengatur data training
...: clftree.score(df_test_att, df_test_label) #Mengatur data testing
Out[9]: 0.26346356916578667

```

Gambar 3.323 Hasil Decision Tree**3.10.2.7 Nomor 7**

```

1 clfsvm = svm.SVC() #clfsvm sebagai variabel untuk mengatur fungsi
SVC
2 clfsvm.fit(df_train_att, df_train_label) #Mengatur data training
3 clfsvm.score(df_test_att, df_test_label) #Mengatur data testing

```

Accuracy: 0.44 (+/- 0.02)

Gambar 3.324 Hasil Cross Validation**3.10.2.8 Nomor 8**

```

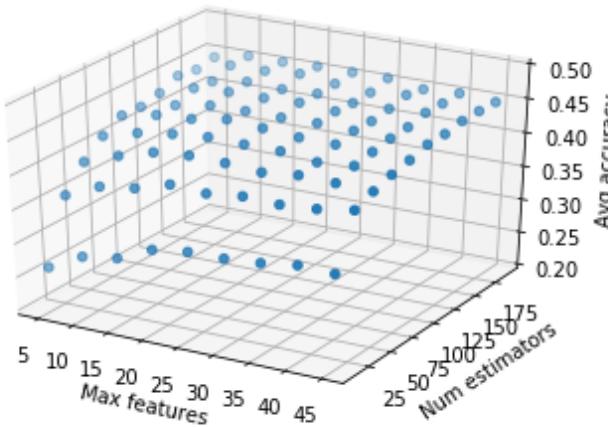
1 scores = cross_val_score(clf, df_train_att, df_train_label, cv=5)
#Membuat variable scores sebagai variabel prediksi dari data
training
2 print("Accuracy: %0.2f (+/- %0.2f)" % (scores.mean(), scores.std()
() * 2)) #menampilkan data scores dengan ketentuan akurasi
3
4 # In [8]:
5
6 import matplotlib.pyplot as plt #Mengimport library matplotlib
sebagai plt

```

```

7 from mpl_toolkits.mplot3d import Axes3D #Mengimport axes3D untuk
   menampilkan plot 3 dimensi
8 from matplotlib import cm #Memanggil data cm yang sudah tersedia
9 fig = plt.figure() #Menghasilkan plot sebagai figure
10 fig.clf() #Figure di ambil dari clf
11 ax = fig.gca(projection='3d') #ax sebagai projection 3d
12 x = rf_params[:,0] #x sebagai index 0
13 y = rf_params[:,1] #y sebagai index 1
14 z = rf_params[:,2] #z sebagai index 2
15 ax.scatter(x, y, z) #Membuat plot scatter x y z

```



Gambar 3.325 Hasil Pengamatan

3.10.3 Penanganan Error

1. ScreenShoot Error

```

FileNotFoundException: [Errno 2] File 'h:/User/reza/Downloads/00_200_201/
attributes/image_attribute_labels.txt' does not exist: b'C:/User/reza/Downloads/
00_200_201/attributes/image_attribute_labels.txt'

```

Gambar 3.326 FileNotFoundError

2. Tuliskan Kode Error dan Jenis Error

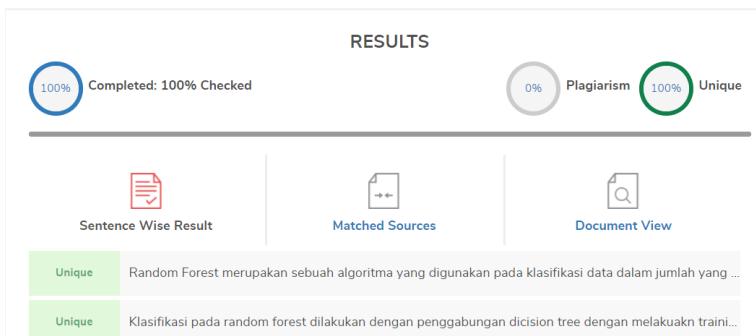
- FileNotFoundError

3. Cara Penanganan Error

- FileNotFoundError

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

3.10.4 Bukti Tidak Plagiat



Gambar 3.327 Bukti Tidak Plagiat

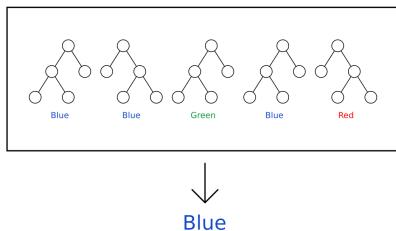
3.10.5 Link Youtube:

<https://youtu.be/qtmtKsziZ1I>

3.11 1174086 - Tia Nur Candida

3.11.1 Soal Teori

1. Jelaskan apa itu random forest, sertakan gambar ilustrasi buatan sendiri. Algoritma yang digunakan pada klasifikasi data dalam jumlah yang besar. Klasifikasi random forest dilakukan melalui penggabungan pohon (tree) dengan melakukan training pada sampel data yang dimiliki. Penggunaan pohon (tree) yang semakin banyak akan mempengaruhi akurasi yang akan didapatkan menjadi lebih baik. Penentuan klasifikasi dengan random forest diambil berdasarkan hasil voting dari tree yang terbentuk. Pemenang dari tree yang terbentuk ditentukan dengan vote terbanyak. Untuk gambar Random Forest dapat dilihat dibawah ini :



Gambar 3.328 Random Forest.

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

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

3. Jelaskan apa itu cross validation.

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

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

Dimana Score 44 % diperoleh dari hasil pengelahan dataset jenis burung. Dimana akan dilakukan proses pembagian data testing dan data training lalu diproses dan menghasilkan score sebanyak 44 % dimana menjelaskan bahwa score tersebut digunakan sebagai pembanding dalam

tingkat keakuratannya. Pada dicision tree akan memperoleh data lebih kecil yaitu sebanyak 27 % hal ini dikarenakan data yang diolah menggunakan dicision tree dibagi menjadi beberapa tree dan lalu disimpulkan untuk mendapatkan data yang akurat. Pada SVM akan memperoleh score sebanyak 29 % hal ini dikarenakan data yang dimiliki masih bernilai netral sehingga tingkat keakuratannya masih belum jelas.

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

Untuk membaca confusion matriks dapat menggunakan source code sebagai berikut :

```

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

```

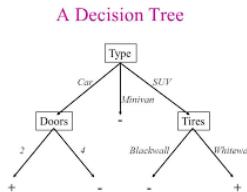
numpy akan mengurus semua data yang berhubungan dengan matrix. Pada source code tersebut digunakan dalam melakukan read pada dataset burung dengan menggunakan metode confusion matrix. Dalam confusion matrix memiliki 4 istilah yaitu True Positive yang merupakan data positif yang terditeksi benar, True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



Gambar 3.329 Confusion Matriks.

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

Sebuah proses pemilihan dari tree yang dimana akan dimunculkan hasilnya dan disimpulkan menjadi informasi yang pasti.

**Gambar 3.330** Decision Tree.**3.11.2 Praktek Program**

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

```

1 # In [44]: Soal1
2
3 import pandas as tia #melakukan import pada library pandas
4           sebagai tia
5
6 makanan = {"Makanan" : [ 'Pizza' , 'Batagor' , 'Cimol' , 'Lumpia' ]} #membuat variabel yang bernama boyband, dan mengisi
7           dataframe nama2 makanan
8 x = tia.DataFrame(makanan) #variabel x membuat DataFrame dari
9           library pandas dan akan memanggil variabel laptop.
10 print (' Makanan kesukaan tia ' + x) #print hasil dari x
  
```

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

	Makanan
0	Makanan kesukaan tiaPizza
1	Makanan kesukaan tiaBatagor
2	Makanan kesukaan tiaCimol
3	Makanan kesukaan tiaLumpia

Gambar 3.331 Hasil Soal1.

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

```

1 # In [44]: Soal2
2
3 import numpy as tia #melakukan import numpy sebagai tia
4
5 matrix_x = tia.eye(10) #membuat matrix dengan numpy dengan
6           menggunakan fungsi eye
7 matrix_x #deklrasikan matrix_x yang telah dibuat
  
```

```
8 print ( matrix_x ) #print matrix_x yang telah dibuat dengan 10
    x10
```

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

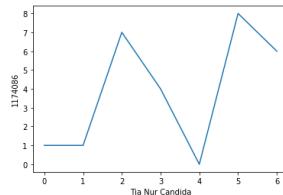
```
[[1. 0. 0. 0. 0. 0. 0. 0. 0.]
 [0. 1. 0. 0. 0. 0. 0. 0. 0.]
 [0. 0. 1. 0. 0. 0. 0. 0. 0.]
 [0. 0. 0. 1. 0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 1. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0. 1. 0. 0. 0.]
 [0. 0. 0. 0. 0. 0. 1. 0. 0.]
 [0. 0. 0. 0. 0. 0. 0. 1. 0.]
 [0. 0. 0. 0. 0. 0. 0. 0. 1.]
 [0. 0. 0. 0. 0. 0. 0. 0. 0.]]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

Gambar 3.332 Hasil Soal 2.

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

```
1 import matplotlib.pyplot as tia #import matplotlib sebagai
    tia
2
3 tia.plot([1,1,7,4,0,8,6]) #memberikan nilai plot atau grafik
    pada tia
4 tia.xlabel('Tia Nur Candida') #memberikan label pada x
5 tia.ylabel('1174086') #memberikan label pada y
6 tia.show() #print hasil plot berbentuk grafik
```

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



Gambar 3.333 Hasil Soal 3.

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

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

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

Hasilnya adalah seperti ini :

Gambar 3.334 Hasil Soal 4 - 1

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

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

Hasilnya adalah seperti ini :

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

Gambar 3.335 Hasil Soal 4 - 2

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

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

Hasilnya adalah seperti ini :

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

Gambar 3.336 Hasil Soal 4 - 3

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

```

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

```

Hasilnya adalah seperti ini :

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

Gambar 3.337 Hasil Soal 4 - 4

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

```

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

```

Hasilnya adalah seperti ini :

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

[5 rows x 312 columns]

Gambar 3.338 Hasil Soal 4 - 5

- Menghasilkan jumlah kolom dan baris pada DataFrame imgatt2.

```

1
2 imgatt2.shape #menampilkan jumlah data , kolom

```

Hasilnya adalah seperti ini :

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

Gambar 3.339 Hasil Soal 4 - 6

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

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

Hasilnya adalah seperti ini :

```
[In 8]: imglabels = pd.read_csv('mnist_250k/mnist_250k_train.csv', header=None, names=['label', 'text'])
[In 9]: imglabels['label'].value_counts()
[In 10]: imglabels.set_index('label', inplace=True)
[In 11]: imglabels
[Out 11]:
          label
0           0
1           1
2           2
3           3
4           4
5           5
6           6
7           7
8           8
9           9
```

Gambar 3.340 Hasil Soal 4 - 7

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

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

Hasilnya adalah seperti ini :

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

Gambar 3.341 Hasil Soal 4 - 8

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

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

Hasilnya adalah seperti ini :

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

Gambar 3.342 Hasil Soal 4 - 9

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

```

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

```

Hasilnya adalah seperti ini :

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

Gambar 3.343 Hasil Soal 4 - 10

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

```

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

```

Hasilnya adalah seperti ini :

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

Gambar 3.344 Hasil Soal 4 - 11

- Mengecek isi 5 data teratas pada df att.

```

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

```

Hasilnya adalah seperti ini :

In [13]:	df_att.head()																																																																																																														
Out[13]:	<table border="1"> <thead> <tr> <th></th> <th>2</th> <th>3</th> <th>4</th> <th>5</th> <th>6</th> <th>7</th> <th>...</th> <th>386</th> <th>387</th> <th>388</th> <th>389</th> <th>318</th> <th>311</th> <th>312</th> </tr> </thead> <tbody> <tr> <td>imgid</td> <td>0</td> <td>0</td> <td>0</td> <td>0</td> <td>0</td> <td>0</td> <td>...</td> <td>0</td> <td>0</td> <td>0</td> <td>0</td> <td>0</td> <td>0</td> <td>1</td> </tr> <tr> <td>2449</td> <td>0</td> <td>0</td> <td>0</td> <td>0</td> <td>0</td> <td>0</td> <td>1</td> <td>...</td> <td>0</td> <td>0</td> <td>0</td> <td>0</td> <td>0</td> <td>0</td> <td>0</td> </tr> <tr> <td>2516</td> <td>0</td> <td>0</td> <td>0</td> <td>0</td> <td>0</td> <td>0</td> <td>0</td> <td>...</td> <td>0</td> <td>0</td> <td>0</td> <td>0</td> <td>0</td> <td>0</td> <td>0</td> </tr> <tr> <td>7472</td> <td>0</td> <td>0</td> <td>0</td> <td>0</td> <td>0</td> <td>0</td> <td>0</td> <td>...</td> <td>0</td> <td>0</td> <td>0</td> <td>0</td> <td>0</td> <td>0</td> <td>1</td> </tr> <tr> <td>7584</td> <td>0</td> <td>0</td> <td>0</td> <td>0</td> <td>0</td> <td>0</td> <td>0</td> <td>...</td> <td>0</td> <td>0</td> <td>0</td> <td>0</td> <td>0</td> <td>0</td> <td>0</td> </tr> <tr> <td>9534</td> <td>0</td> <td>0</td> <td>0</td> <td>0</td> <td>0</td> <td>0</td> <td>1</td> <td>...</td> <td>0</td> <td>0</td> <td>1</td> <td>0</td> <td>0</td> <td>0</td> <td>1</td> </tr> </tbody> </table>		2	3	4	5	6	7	...	386	387	388	389	318	311	312	imgid	0	0	0	0	0	0	...	0	0	0	0	0	0	1	2449	0	0	0	0	0	0	1	...	0	0	0	0	0	0	0	2516	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	7472	0	0	0	0	0	0	0	...	0	0	0	0	0	0	1	7584	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	9534	0	0	0	0	0	0	1	...	0	0	1	0	0	0	1
	2	3	4	5	6	7	...	386	387	388	389	318	311	312																																																																																																	
imgid	0	0	0	0	0	0	...	0	0	0	0	0	0	1																																																																																																	
2449	0	0	0	0	0	0	1	...	0	0	0	0	0	0	0																																																																																																
2516	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0																																																																																																
7472	0	0	0	0	0	0	0	...	0	0	0	0	0	0	1																																																																																																
7584	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0																																																																																																
9534	0	0	0	0	0	0	1	...	0	0	1	0	0	0	1																																																																																																
	[5 rows x 312 columns]																																																																																																														

Gambar 3.345 Hasil Soal 4 - 12

- Mengecek isi data teratas dari df label.

```

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

```

Hasilnya adalah seperti ini :

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

Gambar 3.346 Hasil Soal 4 - 13

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

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

Hasilnya adalah seperti ini :

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

Gambar 3.347 Hasil Soal 4 - 14

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

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

Hasilnya adalah seperti ini :

```
In [18]: from sklearn.ensemble import RandomForestClassifier
...: clf = RandomForestClassifier(max_features=50, random_state=0, n_estimators=100)
```

Gambar 3.348 Hasil Soal 4 - 15

- Menunjukkan hasil prediksi dari Random Forest.

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

Hasilnya adalah seperti ini :

```
In [19]: clf.fit(df_train_att, df_train_label)
...: clf = RandomForestClassifier(bootstrap=True, class_weight=None,
...:                           criterion='gini', max_depth=None,
...:                           max_features=50, max_leaf_nodes=None,
...:                           max_samples=None, min_impurity_decrease=0.0,
...:                           min_impurity_split=None, min_samples_leaf=1,
...:                           min_samples_split=2, min_weight_fraction_leaf=0.0,
...:                           n_estimators=100, oob_score=False, random_state=0,
...:                           verbose=0,
...:                           warm_start=False)
```

Gambar 3.349 Hasil Soal 4 - 16

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

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

Hasilnya adalah seperti ini :

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

Gambar 3.350 Hasil Soal 4 - 17

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

```
1
2 clf.score(df_test_att, df_test_label) #print clf sebagai
   testing yang sudah di training tadi
```

Hasilnya adalah seperti ini :

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

Gambar 3.351 Hasil Soal 4 - 18

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

- Melakukan import confusion matrix pada library sklearn matriks.

```

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

```

Hasilnya adalah seperti ini :

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

Gambar 3.352 Hasil Soal 5 - 1

- Menampilkan isi cm dalam bentuk matrix yang berupa array.

```

1
2 cm #menampilkan data label berbentuk array

```

Hasilnya adalah seperti ini :

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

Gambar 3.353 Hasil Soal 5 - 2

- Membuat dan menampilkan hasil plot.

```

1
2 import matplotlib.pyplot as plt #import library matplotlib
    sebagai plt
3 import itertools #import library itertools
4 def plot_confusion_matrix(cm, classes,
                           normalize=False,
                           title='Confusion matrix',
                           cmap=plt.cm.Blues): #membuat
    fungsi dengan ketentuan data yang ada pada cm
8
9     if normalize:
10         cm = cm.astype('float') / cm.sum(axis=1)[:, np.
    newaxis]
11         print("Normalized confusion matrix") #jika
    normalisasi sebagai ketentuan yang ada maka print
    normalized confusion matrix
12     else:
13         print('Confusion matrix, without normalization') #
    jika tidak memenuhi kondisi if maka print else
14
15     print(cm) #print data cm
16

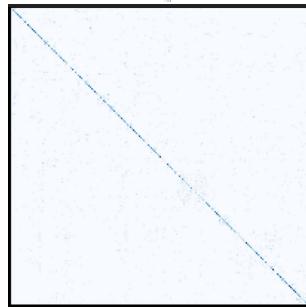
```

```

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

```

Hasilnya adalah seperti dibawah ini :



```
    'a/CUB_200_2011/classes.txt',
    'r', header=None, usecols=[1], names=['birdname']
]
```

Gambar 3.354 Menampilkan hasil plot

6. Jalankan program klasifikasi SVM dan Decision Tree pada bagian teori bab ini. Tunjukkan keluarannya dari komputer sendiri dan artikan mak-sud setiap luaran yang didapatkan.

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

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

Hasilnya adalah seperti ini :

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

Gambar 3.355 Hasil Soal 6 - 1

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

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

Hasilnya adalah seperti ini :

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

Gambar 3.356 Hasil Soal 6 - 2

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

- Menunjukkan hasil cross validation pada Random Forest.

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

Hasilnya adalah seperti ini :

```
In [29]: from sklearn.model_selection import cross_val_score
...: scores = cross_val_score(clf, df_train_att, df_train_label, cv=5)
...: # it shows average score and +/- two standard deviations away (covering 95% of scores)
...: print("Accuracy: %0.2f (+/- %0.2f)" % (scores.mean(), scores.std() * 2))
Accuracy: 0.45 (+/- 0.05)
```

Gambar 3.357 Hasil Soal 7 - 1

- Menunjukkan hasil cross validation pada Desiccion Tree.

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

Hasilnya adalah seperti ini :

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

Gambar 3.358 Hasil Soal 7 - 2

- Menunjukkan hasil cross validation pada SVM.

```

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

```

Hasilnya adalah seperti ini :

```
[In [31]: scoressvm = cross_val_score(clfsvm, df_train_att, df_train_label, cv=5)
...: print("Accuracy: %0.2f (+/- %0.2f)" % (scoressvm.mean(), scoressvm.std() * 2))
Accuracy: 8.47 (+/- 8.81)
```

Gambar 3.359 Hasil Soal 7 - 3

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

- Menunjukkan informasi-informasi tree yang dibuat.

```

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

```

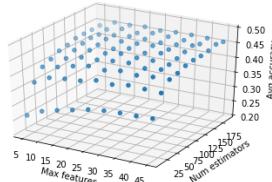
Hasilnya adalah seperti ini :

Gambar 3.360 Hasil Soal 8 - 1

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

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

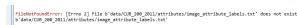
Hasilnya adalah seperti ini :



Gambar 3.361 Hasil Soal 8 - 2

3.11.3 Penanganan Error

1. ScreenShoot Error



```
*[1] Fehler: [Errno 2] File '3\data\108_300_2011\atributes\image_attributes\label1.txt' does not exist:
v\data\108_300\2011\atributes\image_attributes\label1.txt"
```

Gambar 3.362 FileNotFoundError

2. Tuliskan Kode Error dan Jenis Error

- FileNotFoundError

3. Cara Penanganan Error

- FileNotFoundError

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

3.11.4 Bukti Tidak Plagiat



Gambar 3.363 Bukti Tidak Melakukan Plagiat Chapter 3

3.11.5 Link Youtube:

<https://youtu.be/0DsMRyczk80>

3.12 1174070 - Arrizal Furqona Gifary

Chapter 3 - Prediksi dengan Random Forest

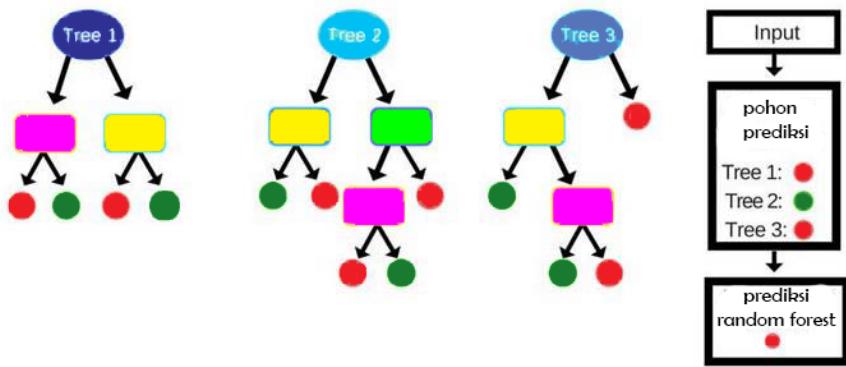
3.12.1 Teori

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

Random Forest adalah konstruk data yang diterapkan pada machine learning yang mengembangkan sejumlah besar pohon keputusan acak yang menganalisis sekumpulan variabel. Jenis algoritma ini membantu meningkatkan cara teknologi menganalisis data yang kompleks. Juga merupakan algoritma machine learning yang fleksibel, mudah digunakan, bahkan tanpa penyetelan

hyperparameter, dengan hasil yang baik. Ini juga merupakan salah satu algoritma yang paling banyak digunakan, karena kesederhanaan dan faktanya dapat digunakan untuk tugas klasifikasi dan regresi.

algoritma random forest



Gambar 3.364 contoh random forest

3.12.1.2 Jelaskan cara membaca dataset kasus dan artikan makna setiap file dan isi field masing masing file.

Dataset adalah kumpulan data. Paling umum satu data set sesuai dengan isi tabel database tunggal, atau matriks data statistik tunggal, dimana setiap kolom tabel mewakili variabel tertentu, dan setiap baris sesuai dengan anggota tertentu dari dataset yang dipertanyakan.

1. cara membaca dataset

- Gunakan librari Pandas pada python untuk dapat membaca dataset dengan format text file.
- Setelah itu,buat variabel baru "dataset" yang berisikan perintah untuk membaca file csv. seperti berikut

```

1 import pandas as pd
2
3 dataset = pd.read_csv('E:/backup/sem 6/Kecerdasan Buatan/
  KB3C - Copy/src/1174083/src3/case.csv', sep=',')
4 dataset.head()
  
```

Pada kodingan diatas dapat dijelaskan bahwa : Memanggil Librari Panda untuk membaca dataset Membuat variabel "Dataset" yang berisikan pd.read_csv untuk membaca dataset.

- setelah di run maka hasilnya seperti berikut

Index	case_id	province	city	group	infection_case	confirmed	latitude	longitude
0	1	Seoul	Guro-gu	True	Guro-gu Call Center	70	37.508163	126.884387
1	2	Seoul	Eunpyeong-gu	True	Eunpyeong St. Mary's Hospital	14	37.63369	126.9165
2	3	Seoul	Seongdong-gu	True	Seongdong-gu APT	13	37.55713	127.0403
3	4	Seoul	Jongno-gu	True	Jongno Community Center	10	37.57681	127.0065
4	5	Seoul	Dongdaemun-gu	True	Donghan Church	9	37.592888	127.056766
5	6	Seoul	Jung-gu	True	Jung-gu Fashion Company	7	37.562405	126.984377
6	7	Seoul	from other city	True	Shincheonji Church	6	-	-
7	8	Seoul	-	False	etc	46	-	-
8	9	Busan	Dongnae-gu	True	Onchun Church	34	35.21628	129.0771
9	10	Busan	from other city	True	Shincheonji Church	18	-	-
10	11	Busan	Suyeong-gu	True	Suyeong-gu Kindergarten	6	35.16708	129.1124
11	12	Busan	Haeundae-gu	True	Haeundae-gu Catholic Church	6	35.20599	129.1256
12	13	Busan	Jin-gu	True	Jin-gu Academy	4	35.17371	129.0633
13	14	Busan	from other city	True	Cheongdo Daemam Hospital	1	-	-
14	15	Busan	-	False	etc	29	-	-
15	16	Daegu	Nam-gu	True	Shincheonji Church	4126	35.84008	128.5667
16	17	Daegu	-	False	contact with patient	688	-	-
17	18	Daegu	from other city	True	Cheongdo Daemam Hospital	2	-	-
18	19	Daegu	-	False	etc	1059	-	-
19	20	Incheon	from other city	True	Guro-gu Call Center	15	-	-
20	21	Incheon	-	False	contact with patient	4	-	-

Gambar 3.365 contoh dataset

Pertama tama gambar diatas merupakan dataset yang digunakan untuk mengetahui penyebaran virus corona di negara korea selatan. Datasetnya dapat didapatkan dari laman https://www.kaggle.com/kimjihoo/covid19_korea. Penjelasan dari isi field diatas adalah sebagai berikut :

- Atribut Index merupakan atribut otomatis untuk penomoran data yang ada.
- Atribut case_id merupakan nomor id dari case
- Atribut province merupakan nama provinsi
- Atribut city merupakan nama kota
- Atribut group terdiri dari dua variable, yaitu TRUE (untuk terindikasi infeksi dari group) dan FALSE(sebaliknya)
- Atribut infection_case merupakan nama kasus yang menginfeksi
- Atribut confirmed merupakan kasus yang sudah terkonfirmasi
- Atribut latitude merupakan titik koordinat
- Atribut longitude merupakan titik koordinat

3.12.1.3 Jelaskan apa itu cross validation

Cross-validation (CV) adalah metode statistik yang dapat digunakan untuk mengevaluasi kinerja model atau algoritma dimana data dipisahkan menjadi dua subset yaitu data proses pembelajaran dan data validasi / evaluasi. Model

atau algoritma dilatih oleh subset pembelajaran dan divalidasi oleh subset validasi. Selanjutnya pemilihan jenis CV dapat didasarkan pada ukuran dataset. Biasanya CV K-fold digunakan karena dapat mengurangi waktu komputasi dengan tetap menjaga keakuratan estimasi. Teknik ini utamanya digunakan untuk melakukan prediksi model dan memperkirakan seberapa akurat sebuah model prediktif ketika dijalankan dalam praktiknya. Dalam sebuah masalah prediksi, sebuah model biasanya diberikan kumpulan data (dataset) yang diketahui untuk digunakan dalam menjalankan pelatihan (dataset pelatihan), serta kumpulan data yang tidak diketahui (atau data yang pertama kali dilihat) terhadap model yang diuji (pengujian dataset) Metode 3 - fold cross validation membagi sebuah himpunan contoh secara acak menjadi 3 subset yang saling bebas. Dilakukan pengulangan sebanyak 3kali untuk pelatihan dan pengujian. Pada setiap ulangan, disisakan satu subset untuk pengujian dan subset lainnya untuk pelatihan. Tingkat akurasi dihitung dengan membagi jumlah keseluruhan klasifikasi yang benar dengan jumlah semua instance pada data awal.

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

Itu merupakan persentase keakurasi prediksi yang dilakukan pada saat testing menggunakan label pada dataset yang digunakan. Score merupakan mendefinisikan aturan evaluasi model. Maka pada saat dijalankan akan muncul persentase tersebut yang menunjukkan keakurasi atau keberhasilan dari prediksi yang dilakukan. Jika menggunakan Random Forest maka hasilnya 40% , jika menggunakan Decision Tree hasil prediksinya yaitu 27% dan pada SVM 29%

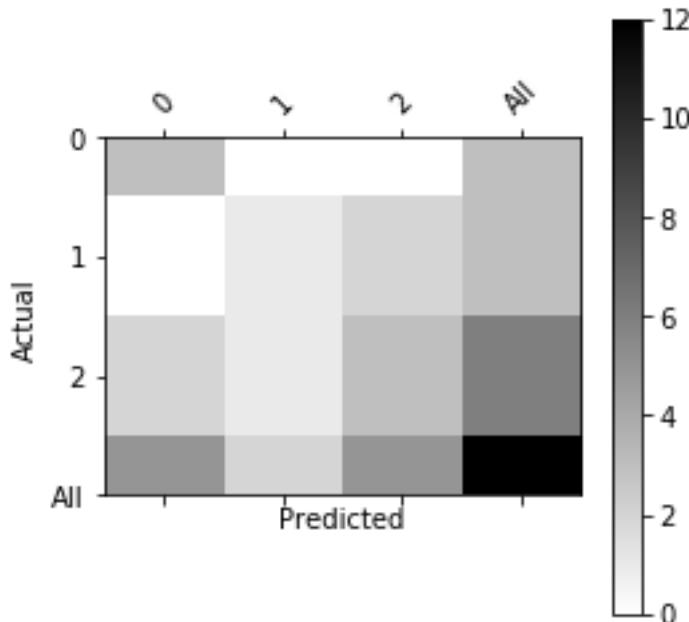
3.12.1.5 Jelaskan bagaimana cara membaca confusion matriks dan contohnya memakai gambar atau ilustrasi sendiri.

Perhitungan Confusion Matriks dapat dilakukan sebagai berikut. Disini saya menggunakan data yang dibuat sendiri untuk menampilkan data aktual dan prediksi.

- Import librari Pandas, Matplotlib, dan Numpy.
- Buat variabel y_actu yang berisikan data aktual.
- Buat variabel y_pred berisikan data yang akan dijadikan sebagai prediksi.
- Buat variabel df_confusion yang berisikan crosstab untuk membangun tabel tabulasi silang yang dapat menunjukkan frekuensi kemunculan kelompok data tertentu.
- Pada variabel df_confusion definisikan lagi nama baris yaitu Actual dan kolomnya Predicted
- Kemudian definisikan suatu fungsi yang diberi nama plot_confusion_matrix yang berisikan pendefinisian confusion matrix dan juga akan di plotting. untuk code lengkapnya sebagai berikut:

```
1 import pandas as pd
2 import numpy as np
3 import matplotlib.pyplot as plt
4
5 y_actu = pd.Series([2, 0, 2, 2, 0, 1, 1, 2, 2, 0, 1, 2], name
6                   ='Actual')
7 y_pred = pd.Series([0, 0, 2, 1, 0, 2, 1, 0, 2, 0, 2, 2], name
8                   ='Predicted')
9 df_confusion = pd.crosstab(y_actu, y_pred, rownames=['Actual',
10                           ], colnames=['Predicted'], margins=True)
11
12 def plot_confusion_matrix(df_confusion, title='Confusion
13                           matrix', cmap=plt.cm.gray_r):
14     plt.matshow(df_confusion, cmap=cmap) # imshow
15     #plt.title(title)
16     plt.colorbar()
17     tick_marks = np.arange(len(df_confusion.columns))
18     plt.xticks(tick_marks, df_confusion.columns, rotation=45)
19     plt.yticks(tick_marks, df_confusion.index)
20     plt.ylabel(df_confusion.index.name)
21     plt.xlabel(df_confusion.columns.name)
22
23 plot_confusion_matrix(df_confusion)
```

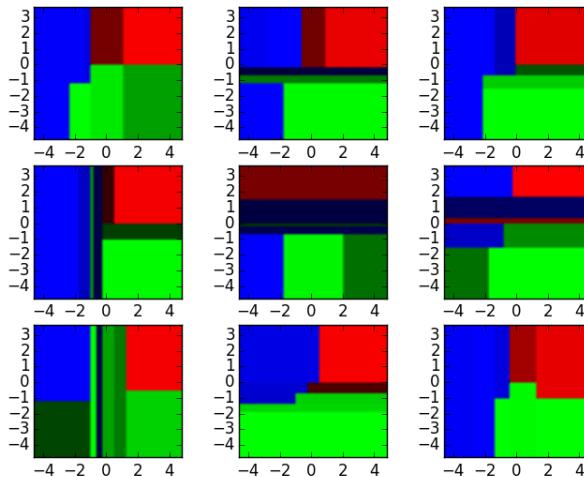
Hasil dari kode diatas akan menghasilkan confusion matrix seperti berikut:



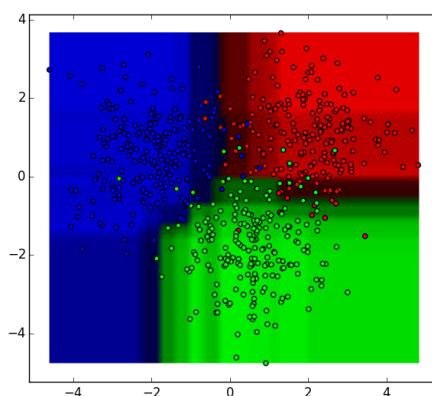
Gambar 3.366 hasil confusion matrix

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

Voting yaitu suara untuk setiap target yang diprediksi pada saat melakukan Random Forest. Pertimbangkan target prediksi dengan voting tertinggi sebagai prediksi akhir dari algoritma random forest. contohnya pada gambar berikut, dimana ada 9 hasil yang berbeda dari decision tree, dan menghasilkan suatu random forest



Gambar 3.367 contoh 9 hasil decision tree yang berbeda



Gambar 3.368 contoh random forest

3.12.2 Praktek

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

Disini saya akan membuat program sederhana menggunakan Pandas yaitu untuk memilih baris dari DataFrame yang diberikan berdasarkan value disalah satu kolom.

```

1 import pandas as pd
2
3 d = { 'col1':[1 ,2 ,3 ,4 ,5] , 'col2' :[3 ,5 ,5 ,4 ,2] , 'col3' :[2 ,3 ,1 ,1 ,4] }
4 df = pd.DataFrame(data=d)
5 print("nomor 1")
6 print("Hasil: ")
7 print(df)

```

Dari code diatas dapat dijelaskan perbarisnya sebagai berikut :

- Baris pertama, yaitu import pandas yang artinya kita akan mengimport librari Pandas dari python dengan inisiasi pd.
- Variabel d didefinisikan sebagai data. data untuk col1, col2, dan col3.
- Variabel df akan mengubah data pada variabel d disejajarkan menjadi baris dan kolom dengan menggunakan fungsi pd.DataFrame(data=d).
- Baris selanjutnya yaitu akan mencetak atau menampilkan tulisan "nomor 1" dan juga "Hasil: " pada jendela console
- print(df) artinya akan mencetak atau menampilkan DataFrame dari data yang telah dibuat tadi.

Hasilnya seperti berikut:

nomor 1			
Hasil:			
	col1	col2	col3
0	1	3	2
1	2	5	3
2	3	5	1
3	4	4	1
4	5	2	4

Gambar 3.369 contoh aplikasi sederhana menggunakan pandas

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

Disini saya akan membuat program sederhana menggunakan numpy, dimana saya membuat array dari angka 1 sebanyak 26 data, artinya dari angka 1-25. dan mengubah array tersebut menjadi matrix 5x5.

```

1 import numpy as np
2
3 slur = np.arange(1,26).reshape(5,5)
4 print("nomor 2")
5 print(slur)

```

Dari code diatas dapat dijelaskan perbarisnya sebagai berikut :

- baris pertama kita meng-import numpy dan menamainya dengan np
- lalu kita membuat variabel slur dengan diisi arraya yang dimulai dari angka 1 sebanyak 26 data(sampai angka 25) lalu mengubahnya kedalam bentuk matrix 5x5
- Baris selanjutnya yaitu akan mencetak atau menampilkan tulisan "nomor 2" pada jendela console
- print(slur) artinya akan mencetak atau menampilkan data yang ada pada variable slur

Hasilnya seperti berikut:

```

nomor 2
[[ 1  2  3  4  5]
 [ 6  7  8  9 10]
 [11 12 13 14 15]
 [16 17 18 19 20]
 [21 22 23 24 25]]

```

Gambar 3.370 contoh aplikasi sederhana menggunakan numpy

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

Disini saya akan membuat program sederhana menggunakan matplotlib.

```

1 import matplotlib.pyplot as plt
2
3 t = np.arange(0., 5., 0.2)
4 print("nomor 3")
5 plt.plot(t, t, 'r—', t, t**2, 'bo', t, t**3, 'g^')
6 plt.show()

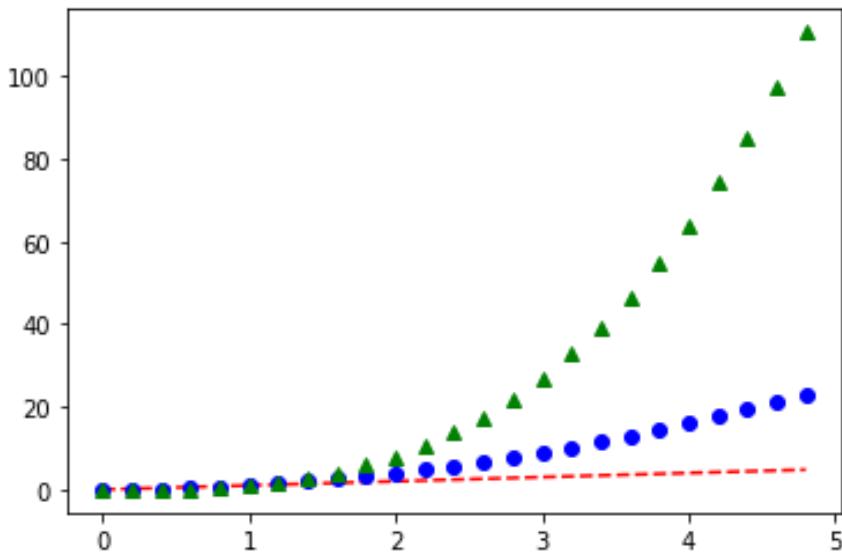
```

Dari code diatas dapat dijelaskan perbarisnya sebagai berikut :

- baris pertama kita meng-import matplotlib.pyplot dan menamainya dengan plt
- membuat variable t dan mengisi datanya dengan array(menggunakan library numpy) yang dimulai dari angka 0 sebanyak 5 array dengan interval 0.2 untuk setiap arraynya.
- Baris selanjutnya yaitu akan mencetak atau menampilkan tulisan "nomor 3" pada jendela console
- plt.plot akan membuat grafik dengan data yang ada pada variable t. 'r-' itu untuk membuat garis putus-putus berwarna merah, t^{**2} itu sama dengan t pangkat 2, 'bo' sama dengan black oval, 'g^' sama dengan segitiga berwarna hijau.
- plt.show() digunakan untuk menampilkan grafik pada saat skrip dijalankan.

Hasilnya seperti berikut:

nomor 3



Gambar 3.371 contoh aplikasi sederhana menggunakan matplotlib

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

yang didapatkan.

Pertama dataset kita baca terlebih dahulu.

```

1 # In [4]: Random Forest
2 import pandas as pd #import library pandas sebagai pd
3
4 imgatt = pd.read_csv("E:/backup/sem 6/Kecerdasan Buatan/
5   CUB_200_2011/attributes/image_attribute_labels.txt",
6   sep='\s+', header=None, error_bad_lines=
7   False, warn_bad_lines=False,
8   usecols=[0,1,2], names=['imgid', 'attid', 'present'])#untuk membaca file txt

```

Listing 3.25 Membaca data file txt

Hasilnya seperti berikut:

```

Name Type Size Value
imgatt DataFrame (3677856, 3) Column names: imgid, attid, present
Variable explorer File explorer Help
IPython console
Console 1/A ×
In [1]: import pandas as pd #import library pandas sebagai pd
...:
...: imgatt = pd.read_csv("E:/backup/sem 6/Kecerdasan Buatan/CUB_200_2011/attributes/
...: image_attribute_labels.txt",
...: sep='\s+', header=None, error_bad_lines=False,
...: warn_bad_lines=False,
...: usecols=[0,1,2], names=['imgid', 'attid', 'present'])#untuk membaca file txt
In [2]:

```

Gambar 3.372 Hasil dari listing 3.1

Melihat sebagian data awal, dengan menggunakan listing 3.2

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

Listing 3.26 Melihat sebagian data(bagian awal)

Hasilnya seperti berikut:

```

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

```

Gambar 3.373 Hasil dari listing 3.2

Melihat jumlah data menggunakan listing 3.3

```
1 imgatt.shape
```

Listing 3.27 Mengetahui jumlah data

Hasilnya seperti berikut:

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

Gambar 3.374 Hasil dari listing 3.3

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

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

Listing 3.28 Pivot dataset

Hasilnya seperti berikut:

imgatt2	DataFrame	(11788, 312)	Column names: 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14,
			15, 16, 1 ...

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

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

Gambar 3.375 Hasil dari listing 3.4

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

```
1 imglabels = pd.read_csv("E:/backup/sem 6/Kecerdasan Buatan/
CUB_200_2011/image_class_labels.txt",
2 sep=' ', header=None, names=['imgid', 'label'])
3
4 imglabels = imglabels.set_index('imgid')
5 imglabels.head()
6 imglabels.shape
```

Listing 3.29 Membaca dataset label file txt

Hasilnya seperti berikut:

<code>imglabels</code>	<code>DataFrame</code>	(11788, 1)	Column names: label
<code>In [8]: imglabels.head()</code>			
<code>Out[8]:</code>			
	label		
<code>imgid</code>			
1	1		
2	1		
3	1		
4	1		<code>In [9]: imglabels.shape</code>
5	1		<code>Out[9]: (11788, 1)</code>

Gambar 3.376 Hasil dari listing 3.5

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

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

Listing 3.30 Menggabungkan field dari dua file terpisah

Hasilnya seperti berikut:

Name	Type	Size	Value
df	DataFrame	(11788, 313)	Column names: 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 1 ...

Gambar 3.377 Hasil dari listing 3.6

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

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

Listing 3.31 Memisahkan dan memilih label

Hasilnya seperti berikut:

Name	Type	Size	Value
df_att	DataFrame	(11788, 312)	Column names: 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 1 ...
df_label	DataFrame	(11788, 1)	Column names: label

Gambar 3.378 Hasil dari listing 3.7

Kita bisa mengecek isinya dengan perintah listing 3.8

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

Listing 3.32 Melihat isi masing-masing dataframe

Hasilnya seperti berikut:

```
In [12]: df_att.head()
Out[12]:
   1    2    3    4    5    6    7    ...   306   307   308   309   310   311   312
imgid
3652    0    0    0    0    0    0    0    ...    1    0    0    1    0    0    0
647     0    0    0    0    0    0    1    ...    0    0    0    1    0    0    0
11192    0    0    0    0    0    0    1    ...    0    1    0    0    0    0    0
8033     0    0    0    0    0    0    1    ...    0    0    0    1    0    0    0
7286     0    0    0    0    0    0    0    ...    0    0    0    0    0    1    0

[5 rows x 312 columns]

In [13]: df_label.head()
Out[13]:
      label
imgid
3652      63
647       12
11192     191
8033      137
7286      125
```

Gambar 3.379 Hasil dari listing 3.8

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

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

Listing 3.33 Pembagian data training dan test

Hasilnya seperti berikut:

Name	Type	Size	Value
df_test_att	DataFrame	(3788, 312)	Column names: 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 1 ...
df_test_label	Series	(3788,)	Series object of pandas.core.series module
df_train_att	DataFrame	(8000, 312)	Column names: 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 1 ...
df_train_label	Series	(8000,)	Series object of pandas.core.series module

Gambar 3.380 Hasil dari listing 3.9

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

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

Listing 3.34 Pembagian data training dan testInstansiasi kelas Random Forest

Hasilnya seperti berikut:

```
In [15]: from sklearn.ensemble import RandomForestClassifier
...: clf = RandomForestClassifier(max_features=50, random_state=0, n_estimators=100)
```

Gambar 3.381 Hasil dari listing 3.10

Kemudian lakukan fit untuk membangun random forest yang sudah ditentukan dengan maksimum fitur sebanya 50 untuk perpohnnya dengan perintah listing 3.11

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

Listing 3.35 Fitting random forest dengan dataset training

Hasilnya seperti berikut:

```
In [16]: clf.fit(df_train_att, df_train_label)
Out[16]:
RandomForestClassifier(bootstrap=True, class_weight=None, criterion='gini',
   max_depth=None, max_features=50, max_leaf_nodes=None,
   min_impurity_decrease=0.0, min_impurity_split=None,
   min_samples_leaf=1, min_samples_split=2,
   min_weight_fraction_leaf=0.0, n_estimators=100,
   n_jobs=None, oob_score=False, random_state=0, verbose=0,
   warm_start=False)
```

Gambar 3.382 Hasil dari listing 3.11

Hasilnya bisa kita dapatkan dengan perintah predict dengan perintah listing 3.12

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

Listing 3.36 Melihat Hasil prediksi

Hasilnya seperti berikut:

```
In [17]: print(clf.predict(df_train_att.head()))
[ 63  12 191 137 125]
```

Gambar 3.383 Hasil dari listing 3.12

Untuk besaran akurasinya dengan perintah listing 3.13

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

Listing 3.37 Score perolehan dari klasifikasi

Hasilnya seperti berikut:

```
In [18]: clf.score(df_test_att, df_test_label)
Out[18]: 0.4263463569165787
```

Gambar 3.384 Hasil dari listing 3.13

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

Dari Random Forest kita coba petakan ke dalam Confusion Matrix dan lihat hasilnya dengan perintah listing 3.14

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

Listing 3.38 Membuat Confusion Matrix

Hasilnya seperti berikut:

Name	Type	Size	Value
cm	int64	(200, 200)	[[5 0 0 ... 0 0 0] [0 11 0 ... 0 0 0]
pred_labels	int64	(3788,)	[44 19 11 ... 45 47 70]

```
In [20]: cm
Out[20]:
array([[ 5,  0,  0, ...,  0,  0,  0],
       [ 0, 11,  0, ...,  0,  0,  0],
       [ 2,  0,  6, ...,  0,  0,  0],
       ...,
       [ 0,  0,  0, ...,  4,  0,  0],
       [ 0,  0,  0, ...,  0,  8,  0],
       [ 0,  0,  0, ...,  0,  0, 13]], dtype=int64)
```

Gambar 3.385 Hasil dari listing 3.14

Kemudian kita plot dengan perintah

```
1 import matplotlib.pyplot as plt
2 import itertools
3 def plot_confusion_matrix(cm, classes,
4                           normalize=False,
5                           title='Confusion matrix',
6                           cmap=plt.cm.Blues):
7     """
8     This function prints and plots the confusion matrix.
9     Normalization can be applied by setting `normalize=True`.
10    """
11    if normalize:
12        cm = cm.astype('float') / cm.sum(axis=1)[:, np.newaxis]
13        print("Normalized confusion matrix")
14    else:
15        print('Confusion matrix, without normalization')
16
17    print(cm)
18
19    plt.imshow(cm, interpolation='nearest', cmap=cmap)
20    plt.title(title)
21    #plt.colorbar()
22    tick_marks = np.arange(len(classes))
23    plt.xticks(tick_marks, classes, rotation=90)
24    plt.yticks(tick_marks, classes)
25
26    fmt = '.2f' if normalize else 'd'
27    thresh = cm.max() / 2.
28
29    plt.tight_layout()
30    plt.ylabel('True label')
31    plt.xlabel('Predicted label')
```

Listing 3.39 Plotting Confusion Matrix

Hasilnya seperti berikut:

```
In [21]: import matplotlib.pyplot as plt
...: import itertools
...: def plot_confusion_matrix(cm, classes,
...:                         normalize=False,
...:                         title='Confusion matrix',
...:                         cmap=plt.cm.Blues):
...:     """
...:     This function prints and plots the confusion matrix.
...:     Normalization can be applied by setting `normalize=True`.
...:     """
...:     if normalize:
...:         cm = cm.astype('float') / cm.sum(axis=1)[:, np.newaxis]
...:         print("Normalized confusion matrix")
...:     else:
...:         print('Confusion matrix, without normalization')
...:
...:     print(cm)
...:
...:     plt.imshow(cm, interpolation='nearest', cmap=cmap)
...:     plt.title(title)
...:     #plt.colorbar()
...:     tick_marks = np.arange(len(classes))
...:     plt.xticks(tick_marks, classes, rotation=90)
...:     plt.yticks(tick_marks, classes)
...:
...:     fmt = '.2f' if normalize else 'd'
...:     thresh = cm.max() / 2.
...:
...:     plt.tight_layout()
...:     plt.ylabel('True label')
...:     plt.xlabel('Predicted label')
```

Gambar 3.386 Hasil dari listing 3.15

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

```
1 birds = pd.read_csv("E:/backup/sem 6/Kecerdasan Buatan/  
2 CUB_200_2011/classes.txt",  
3 sep='\s+', header=None, usecols=[1], names=[  
4 'birdname'])  
birds = birds['birdname']  
birds
```

Listing 3.40 Membaca file classes.txt

Hasilnya seperti berikut:

```
In [22]: birds = pd.read_csv("E:/backup/sem 6/Kecerdasan Buatan/CUB_200_2011/classes.txt",
    ...:                         sep='\s+', header=None, usecols=[1], names=['birdname'])
    ...: birds = birds['birdname']
    ...: birds
Out[22]:
0      001.Black_footed_Albatross
1      002.Laysan_Albatross
2      003.Sooty_Albatross
3      004.Groove_billed_Ani
4      005.Crested_Auklet
...
195      196.House_Wren
196      197.Marsh_Wren
197      198.Rock_Wren
198      199.Winter_Wren
199      200.Common_Yellowthroat
Name: birdname, Length: 200, dtype: object
```

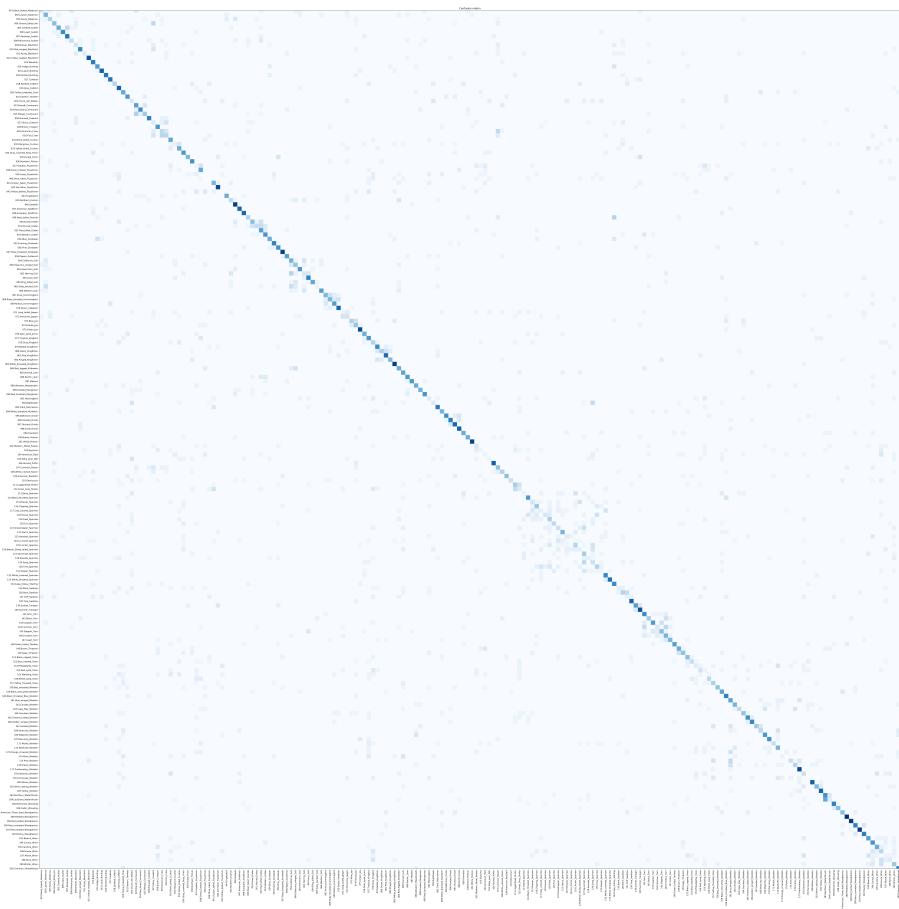
Gambar 3.387 Hasil dari listing 3.16

Lalu kita plot

```
1 import numpy as np
2 np.set_printoptions(precision=2)
3 plt.figure(figsize=(60,60), dpi=300)
4 plot_confusion_matrix(cm, classes=birds, normalize=True)
5 plt.savefig("E:/backup/sem 6/Kecerdasan Buatan/KB3C - Copy/
    figures/1174083/figures3/ganti.png")
```

Listing 3.41 Plot hasil perubahan label

Hasilnya seperti berikut:



Gambar 3.388 Hasil dari listing 3.17

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

Kita coba menggunakan Decission tree

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

Listing 3.42 Mencoba klasifikasi dengan decission tree dengan dataset yang sama

Hasilnya seperti berikut:

```
In [23]: from sklearn import tree
.... clftree = tree.DecisionTreeClassifier()
.... clftree.fit(df_train_att, df_train_label)
.... clftree.score(df_test_att, df_test_label)
Out[23]: 0.2732312565997888
```

Gambar 3.389 Hasil dari listing 3.18

Kita coba menggunakan SVM

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

Listing 3.43 Mencoba klasifikasi dengan SVM dengan dataset yang sama

Hasilnya seperti berikut:

```
In [24]: from sklearn import svm
.... clfsvm = svm.SVC()
.... clfsvm.fit(df_train_att, df_train_label)
.... clfsvm.score(df_test_att, df_test_label)
C:\ProgramData\Anaconda\lib\site-packages\sklearn\svm\base.py:193: FutureWarning: The
default value of gamma will change from 'auto' to 'scale' in version 0.22 to account better
for unscaled features. Set gamma explicitly to 'auto' or 'scale' to avoid this warning.
  "avoid this warning.", FutureWarning)
Out[24]: 0.27666314677930304
```

Gambar 3.390 Hasil dari listing 3.19

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

Pengecekan Cross Validation untuk Random Forest

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

Listing 3.44 Hasil Cross Validation random forest

Hasilnya seperti berikut:

```
In [25]: from sklearn.model_selection import cross_val_score
.... scores = cross_val_score(clf, df_train_att, df_train_label, cv=5)
.... # show average score and +/- two standard deviations away (covering 95% of scores)
.... print("Accuracy: %0.2f (+/- %0.2f)" % (scores.mean(), scores.std() * 2))
Accuracy: 0.44 (+/- 0.02)
```

Gambar 3.391 Hasil dari listing 3.20

untuk decision tree

```

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

```

Listing 3.45 Hasil Cross Validation Random Forest

Hasilnya seperti berikut:

```

In [26]: scorestree = cross_val_score(clftree, df_train_att, df_train_label, cv=5)
...: print("Accuracy: %0.2f (+/- %0.2f)" % (scorestree.mean(), scorestree.std() * 2))
Accuracy: 0.25 (+/- 0.02)

```

Gambar 3.392 Hasil dari listing 3.21

untuk SVM

```

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

```

Listing 3.46 Hasil Cross Validation SVM

Hasilnya seperti berikut:

```

In [27]: scoressvm = cross_val_score(clfsvm, df_train_att, df_train_label, cv=5)
...: print("Accuracy: %0.2f (+/- %0.2f)" % (scoressvm.mean(), scoressvm.std() * 2))
Accuracy: 0.26 (+/- 0.01)

```

Gambar 3.393 Hasil dari listing 3.22

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

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

```

1 max_features_opts = range(5, 50, 5)
2 n_estimators_opts = range(10, 200, 20)
3 rf_params = np.empty((len(max_features_opts)*len(
    n_estimators_opts),4), float)
4 i = 0
5 for max_features in max_features_opts:
6     for n_estimators in n_estimators_opts:
7         clf = RandomForestClassifier(max_features=max_features ,
    n_estimators=n_estimators)
8         scores = cross_val_score(clf , df_train_att ,
    df_train_label , cv=5)
9         rf_params[i,0] = max_features
10        rf_params[i,1] = n_estimators
11        rf_params[i,2] = scores.mean()

```

```

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

```

Listing 3.47 Melakukan Pengamatan komponen informasi

Hasilnya seperti berikut:

```

(max_features, n_estimators, scores.mean(), scores.std() * 2))
Max features: 5, num estimators: 10, accuracy: 0.26 (+/- 0.02)
Max features: 5, num estimators: 30, accuracy: 0.36 (+/- 0.04)
Max features: 5, num estimators: 50, accuracy: 0.39 (+/- 0.02)
Max features: 5, num estimators: 70, accuracy: 0.40 (+/- 0.02)
Max features: 5, num estimators: 90, accuracy: 0.42 (+/- 0.02)
Max features: 5, num estimators: 110, accuracy: 0.42 (+/- 0.02)
Max features: 5, num estimators: 130, accuracy: 0.43 (+/- 0.03)
Max features: 5, num estimators: 150, accuracy: 0.43 (+/- 0.02)
Max features: 5, num estimators: 170, accuracy: 0.44 (+/- 0.02)
Max features: 5, num estimators: 190, accuracy: 0.44 (+/- 0.03)
Max features: 10, num estimators: 10, accuracy: 0.28 (+/- 0.03)
Max features: 10, num estimators: 30, accuracy: 0.38 (+/- 0.03)
Max features: 10, num estimators: 50, accuracy: 0.40 (+/- 0.03)
Max features: 10, num estimators: 70, accuracy: 0.42 (+/- 0.03)
Max features: 10, num estimators: 90, accuracy: 0.43 (+/- 0.03)
Max features: 10, num estimators: 110, accuracy: 0.43 (+/- 0.02)
Max features: 10, num estimators: 130, accuracy: 0.44 (+/- 0.03)
Max features: 10, num estimators: 150, accuracy: 0.44 (+/- 0.04)
Max features: 10, num estimators: 170, accuracy: 0.45 (+/- 0.02)
Max features: 10, num estimators: 190, accuracy: 0.45 (+/- 0.03)
Max features: 15, num estimators: 10, accuracy: 0.30 (+/- 0.03)
Max features: 15, num estimators: 30, accuracy: 0.38 (+/- 0.03)
Max features: 15, num estimators: 50, accuracy: 0.42 (+/- 0.03)
Max features: 15, num estimators: 70, accuracy: 0.43 (+/- 0.03)
Max features: 15, num estimators: 90, accuracy: 0.44 (+/- 0.04)
Max features: 15, num estimators: 110, accuracy: 0.44 (+/- 0.02)
Max features: 15, num estimators: 130, accuracy: 0.44 (+/- 0.02)
Max features: 15, num estimators: 150, accuracy: 0.45 (+/- 0.02)
Max features: 15, num estimators: 170, accuracy: 0.45 (+/- 0.03)
Max features: 15, num estimators: 190, accuracy: 0.45 (+/- 0.02)
Max features: 20, num estimators: 10, accuracy: 0.32 (+/- 0.01)
Max features: 20, num estimators: 30, accuracy: 0.39 (+/- 0.03)
Max features: 20, num estimators: 50, accuracy: 0.42 (+/- 0.02)
Max features: 20, num estimators: 70, accuracy: 0.43 (+/- 0.03)
Max features: 20, num estimators: 90, accuracy: 0.43 (+/- 0.02)
Max features: 20, num estimators: 110, accuracy: 0.44 (+/- 0.03)
Max features: 20, num estimators: 130, accuracy: 0.45 (+/- 0.03)

```

Gambar 3.394 Hasil dari listing 3.23

```

Max features: 30, num estimators: 10, accuracy: 0.32 (+/- 0.02)
Max features: 30, num estimators: 30, accuracy: 0.40 (+/- 0.02)
Max features: 30, num estimators: 50, accuracy: 0.42 (+/- 0.02)
Max features: 30, num estimators: 70, accuracy: 0.43 (+/- 0.01)
Max features: 30, num estimators: 90, accuracy: 0.44 (+/- 0.03)
Max features: 30, num estimators: 110, accuracy: 0.44 (+/- 0.02)
Max features: 30, num estimators: 130, accuracy: 0.45 (+/- 0.03)
Max features: 30, num estimators: 150, accuracy: 0.45 (+/- 0.03)
Max features: 30, num estimators: 170, accuracy: 0.45 (+/- 0.02)
Max features: 30, num estimators: 190, accuracy: 0.45 (+/- 0.03)
Max features: 35, num estimators: 10, accuracy: 0.33 (+/- 0.02)
Max features: 35, num estimators: 30, accuracy: 0.39 (+/- 0.02)
Max features: 35, num estimators: 50, accuracy: 0.42 (+/- 0.03)
Max features: 35, num estimators: 70, accuracy: 0.43 (+/- 0.02)
Max features: 35, num estimators: 90, accuracy: 0.44 (+/- 0.03)
Max features: 35, num estimators: 110, accuracy: 0.44 (+/- 0.02)
Max features: 35, num estimators: 130, accuracy: 0.44 (+/- 0.02)
Max features: 35, num estimators: 150, accuracy: 0.45 (+/- 0.03)
Max features: 35, num estimators: 170, accuracy: 0.45 (+/- 0.02)
Max features: 35, num estimators: 190, accuracy: 0.45 (+/- 0.02)
Max features: 40, num estimators: 10, accuracy: 0.33 (+/- 0.02)
Max features: 40, num estimators: 30, accuracy: 0.40 (+/- 0.01)
Max features: 40, num estimators: 50, accuracy: 0.42 (+/- 0.02)
Max features: 40, num estimators: 70, accuracy: 0.43 (+/- 0.02)
Max features: 40, num estimators: 90, accuracy: 0.44 (+/- 0.03)
Max features: 40, num estimators: 110, accuracy: 0.44 (+/- 0.02)
Max features: 40, num estimators: 130, accuracy: 0.45 (+/- 0.03)
Max features: 40, num estimators: 150, accuracy: 0.45 (+/- 0.03)
Max features: 40, num estimators: 170, accuracy: 0.45 (+/- 0.02)
Max features: 40, num estimators: 190, accuracy: 0.45 (+/- 0.03)
Max features: 45, num estimators: 10, accuracy: 0.33 (+/- 0.02)
Max features: 45, num estimators: 30, accuracy: 0.40 (+/- 0.03)
Max features: 45, num estimators: 50, accuracy: 0.42 (+/- 0.02)
Max features: 45, num estimators: 70, accuracy: 0.43 (+/- 0.03)
Max features: 45, num estimators: 90, accuracy: 0.44 (+/- 0.03)
Max features: 45, num estimators: 110, accuracy: 0.44 (+/- 0.02)
Max features: 45, num estimators: 130, accuracy: 0.44 (+/- 0.02)
Max features: 45, num estimators: 150, accuracy: 0.45 (+/- 0.02)
Max features: 45, num estimators: 170, accuracy: 0.45 (+/- 0.02)
Max features: 45, num estimators: 190, accuracy: 0.45 (+/- 0.02)
Max features: 45, num estimators: 190, accuracy: 0.45 (+/- 0.01)

```

Gambar 3.395 Hasil dari listing 3.23(2)

Dan kita bisa melakukan plot informasi ini dengan kode

```

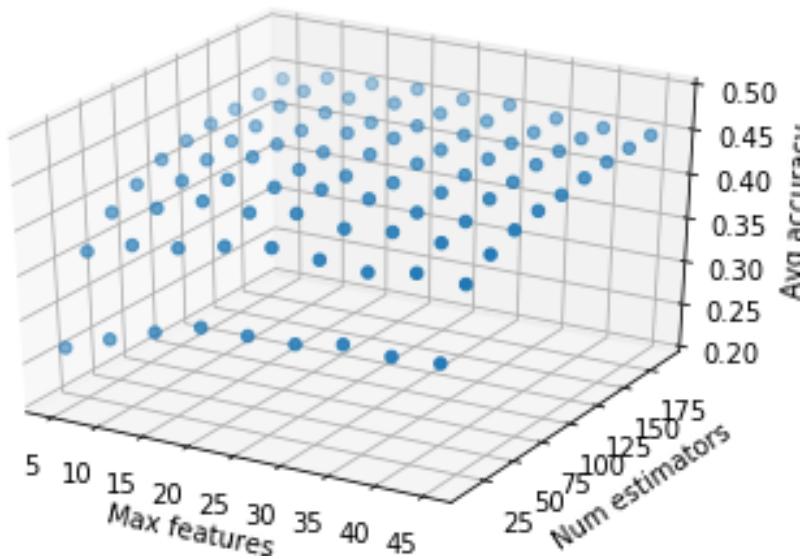
1 import matplotlib.pyplot as plt
2 from mpl_toolkits.mplot3d import Axes3D
3 from matplotlib import cm
4 fig = plt.figure()
5 fig.clf()
6 ax = fig.gca(projection='3d')
7 x = rf_params[:,0]
8 y = rf_params[:,1]
9 z = rf_params[:,2]
10 ax.scatter(x, y, z)
11 ax.set_zlim(0.2, 0.5)
12 ax.set_xlabel('Max features')

```

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

Listing 3.48 Plot Komponen informasi agar bisa dibaca

Hasilnya seperti berikut:



Gambar 3.396 Hasil dari listing 3.24

3.12.3 Penanganan Error

3.12.3.1 *skrinsut Error*

Untuk Error, saya tidak menemukannya. tetapi saya mengalami not responding beberapa kali. seperti pada gambar berikut, dimana not responding terjadi karena laptop dengan spek yang minim melakukan kinerja yang sangat berat.

The screenshot shows the Spyder Python IDE interface. The top menu bar includes File, Edit, Search, Source, Run, Debug, IPython, Projects, Tools, View, Help, and a Help icon. The toolbar contains icons for file operations, run, stop, and refresh. The left sidebar has sections for Editor, IPython console, and Task Manager.

The main area displays Python code for a bird classifier:

```
# Spyder (Python 3.7)
# File Edit Search Source Run Debug IPython Projects Tools View Help
# E:\backup\item\Kecerdasan Buatan\K3C - Copy\1174083\3\bird-identifier.py
# brd-identifier.py  Student performance.py  1174083.py
# Help
# x Help
# C:\Users\bala Qlan
# Editor: E:\backup\item\Kecerdasan Buatan\K3C - Copy\1174083\3\bird-identifier.py
# prakhar10y Student performance.py  1174083.py
# x Help
# 131     fet = '1.2f' if normalize else 'd'
# 132     thresh = cm.max() / 2.
# 133
# 134     plt.tight_layout()
# 135     plt.ylabel('True label')
# 136     plt.xlabel('Predicted label')
# 137
# 138 # In[5,]: Confusion Matrix
# 139
# 140 birds = pd.read_csv("E:/backup/semeval6/Kecerdasan Buatan/CUB_200_2011/classes.txt",
# 141 sep='\t', header=None, usecols=[1], names=['birdname'])
# 142 birds = birds['birdname']
# 143 birds
# 144
# 145 # In[5,]: Confusion Matrix
# 146
# 147 import numpy as np
# 148 np.set_printoptions(precision=2)
# 149 plt.figure(figsize=(60,60), dpi=300)
# 150 cm = confusion_matrix(cm, classes=birds, normalize=True)
# 151 plt.imshow(cm)
# 152 plt.show()
# 153
# 154 # In[6]: Decision Tree dan SVM
# 155
# 156 from sklearn import tree
# 157 from sklearn.svm import SVC
# 158
# 159 # Decision Tree
# 160 clftree = tree.DecisionTreeClassifier()
# 161 clftree.fit(df_train_att, df_train_label)
# 162 clftree.score(df_test_att, df_test_label)
# 163 plt.savefig("E:/Backup/item/Kecerdasan Buatan/K3C - Copy/images/1174083/figures3/green_birds.png")
# 164
# 165 # SVM
# 166 from sklearn import svm
# 167 cflsvm = svm.SVC()
# 168 cflsvm.fit(df_train_att, df_train_label)
# 169 cflsvm.score(df_test_att, df_test_label)
# 170
# 171 # In[7]: Cross Validation
```

The IPython console shows the command `In [3]:`. The Task Manager window is open, showing the following processes:

Name	Status	23%	64%	1%	0%
Google Chrome (13)	0%	621.8 MB	0 MB/s	0	0
Python (9)	3.9%	551.9 MB	0 MB/s	0	0
Microsoft Edge (14)	0.7%	268.8 MB	0 MB/s	0	0
Bidfender Security Service	0%	187.0 MB	0 MB/s	0	0
OBS 24.0.3 (64-bit, windows) - Profile U...	5.7%	124.7 MB	0 MB/s	0	0

The bottom status bar shows permissions: RW, end-of-lines: LF, encoding: UTF-8, line: 148, column: 33, memory: 64%, and the date/time: 18/05/2020 6:12.

Gambar 3.397 Before Disaster

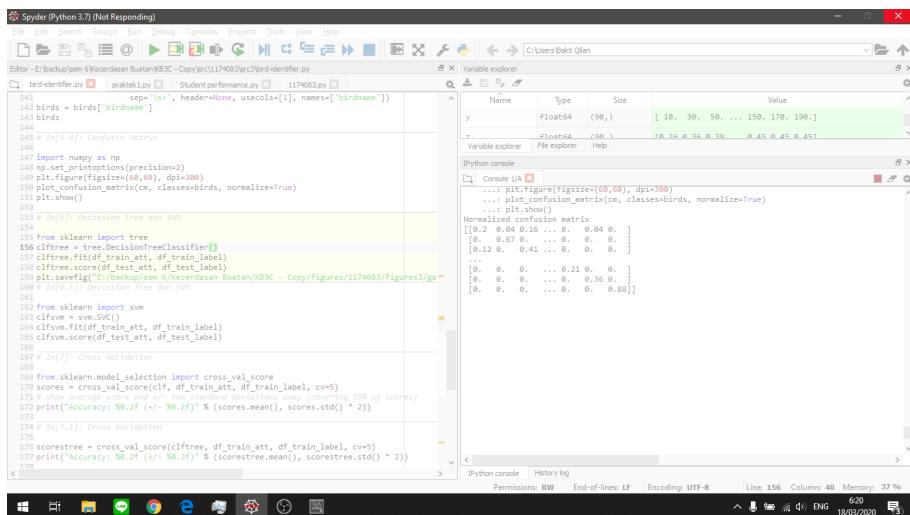
The screenshot shows the Spyder Python IDE interface. A large red arrow points from the bottom left towards the Task Manager window, which is overlaid on the main code editor area. The Task Manager window displays a list of running processes:

Name	Status
Python (9)	Running
Google Chrome (12)	Running
Microsoft Edge (14)	Running
SideFinder Security Service	Running
OSB 24.0.3 (64-bit, windows) - Profile U...	Running

Below the Task Manager, there is a "Fewer details" button and an "End task" button. The main code editor area contains Python code for a bird classifier, and the IPython console shows the execution of the code and its output.

Gambar 3.398 After Disaster

3.12.3.2 Tuliskan Kode Error dan Jenis Error



Gambar 3.399 Not Responding pun terjadi

3.12.3.3 Cara Penanganan Error

Untuk mengatasinya, saya endtask terlebih dahulu spydernya. dan memodifikasi codingannya. yang asalnya

```
# In[5.4]: Confusin Matrix
```

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

Gambar 3.400 Sebelum diubah

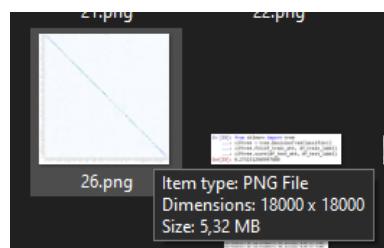
```
# In[5.4]: Confusin Matrix
```

```
import numpy as np
np.set_printoptions(precision=2)
plt.figure(figsize=(60,60), dpi=300)
plot_confusion_matrix(cm, classes=birds, normalize=True)
plt.savefig("E:/backup/sem 6/Kecerdasan Buatan/KB3C - Copy/figures/1174083/figures3/ganti.png")
```

Gambar 3.401 Setelah diubah

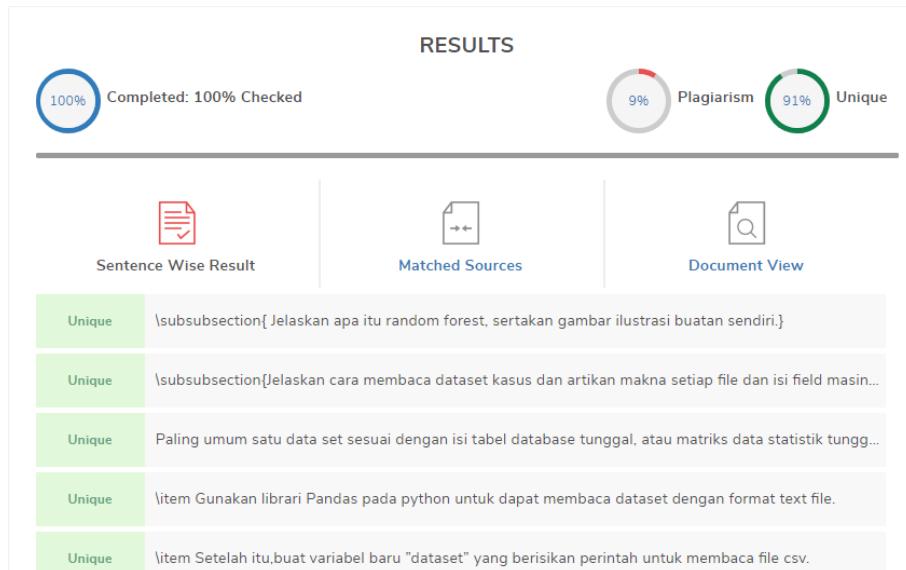
yang asalnya pada baris terakhir itu plt.show() yaitu untuk menampilkan-nya di console, saya ganti menjadi plt.savefig yang berfungsi menyimpan langsung gambar dari grafik tersebut. dan not responding ini terjadi karena laptop berusaha menampilkan gambar dengan ukuran yang besar, yaitu 18000x18000px

tetapi speknya hanya tidak mampu maka terjadilah not responding. tetapi jika kita hanya langsung men save figurenya maka proses akan berjalan lancar meskipun agak lama.

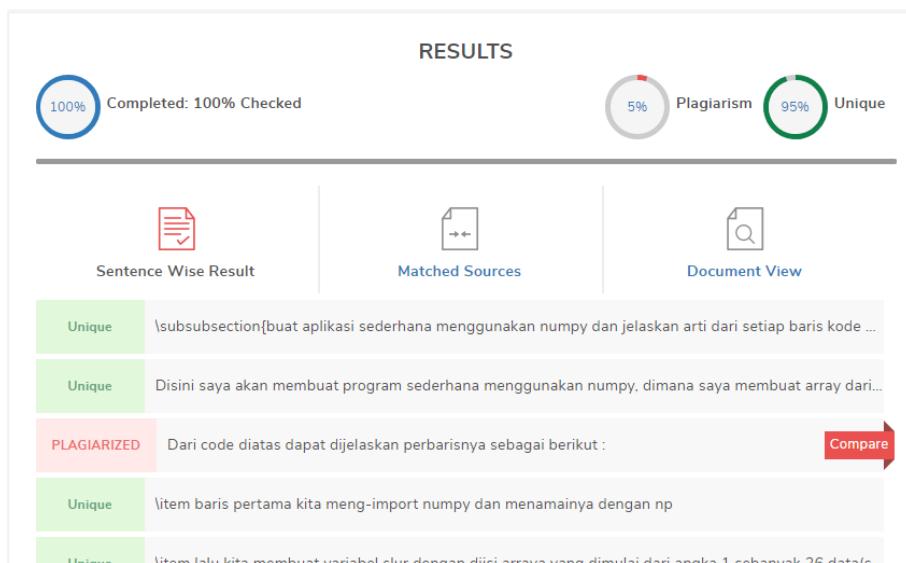


Gambar 3.402 Detail Gambar Confusion Matrix

3.12.3.4 *Bukti Tidak Plagiat*



Gambar 3.403 Cek Plagiarism



Gambar 3.404 Cek Plagiarism(2)

3.12.4 Link Video Youtube

<https://youtu.be/DyJRzPrFxiQ>

BAB 4

CHAPTER 4

4.1 1174006 - Kadek Diva Krishna Murti

Lorem ipsum dolor sit amet, consectetur adipiscing elit.

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



Gambar 4.1 Kecerdasan Buatan.

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

4.1.1 Teori

4.1.2 Praktek

4.1.3 Penanganan Error

4.1.4 Bukti Tidak Plagiat



Gambar 4.2 Kecerdasan Buatan.

4.2 1174069 - Fanny Shafira Damayanti

4.2.1 Teori

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

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



Gambar 4.3 Klasifikasi teks.

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

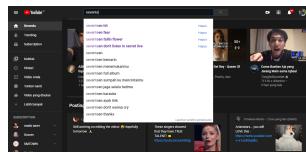
Karena machine learning tidak dapat menampilkan inputan sesuai dengan apa yang kita inputkan. Karena inputan tersebut serupa namun mesin memberikan output yang berbeda, biasanya output atau error ini disebut dengan istilah noise. Untuk contoh sederhananya misalkan kita inputkan salah satu label yang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



Gambar 4.4 Klasifikasi Bunga.

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

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



Gambar 4.5 Klasifikasi teks Youtube.

4. Jelaskan apa yang dimaksud vektorisasi data.

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

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

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



Gambar 4.6 Bag of words.

6. Jelaskan apa yang dimaksud dengan TF-IDF.

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

$$\text{tfidf}(t, d, D) = \text{tf}(t, d) \times \text{idf}(t, D)$$

	Doc 1	Doc 2	...	Doc n
Term(s) 1	12	2	...	1
Term(s) 2	0	1	...	0
...
Term(s) n	0	6	...	3

Gambar 4.7 TF-IDF.

4.2.2 Praktek Program

1. Soal 1

```

1 #%% Soal 1
2 import pandas as pd #digunakan untuk mengimport library
# pandas dengan alias pd
3 pd = pd.read_csv("F://Semester 6/Artificial Intelligence/
# Tugas 4/src/csv_fanny.csv") #membaca file csv

```

Kode di atas digunakan untuk buat aplikasi sederhana menggunakan pandas dengan format csv sebanyak 500 baris, hasilnya ialah sebagai berikut :

Gambar 4.8 Hasil Soal 1.

2. Soal 2

```

1 #%% Soal 2
2 d_train=pd[:450] #membagi data training menjadi 450
3 d_test=pd[450:] #membagi data menjadi 50 atau sisa dari data
      yang tersedia

```

Kode di atas digunakan untuk memecah dataframe tersebut menjadi dua bagian yaitu 450 row pertama dan 50 row kedua. Hasilnya adalah sebagai berikut :

Gambar 4.9 Hasil Soal 2.

3. Soal 3

```

1 #%% Soal 3
2 import pandas as fanny #untuk import library pandas berguna
      untuk mengelola dataframe
3 fanny = fanny.read_csv("F://Semester 6/Artificial
      Intelligence/Tugas 4/src/Youtube03-LMFAO.csv") #membaca
      file dengan format csv
4
5 spam=fanny.query('CLASS == 1') #membagi tabel spam
6 nospam=fanny.query('CLASS == 0')#membagi tabel no spam
7
8 from sklearn.feature_extraction.text import CountVectorizer #
      untuk import countvectorizer berfungsi untuk memecah data
      tersebut menjadi sebuah kata yang lebih sederhana
9 vectorizer = CountVectorizer () #ntuk menjalankan fungsi
      tersebut , pada code ini tidak ada hasilnya dikarenakan
      spyder tidak mendukung hasil dari instasiasi.
10
11 dvec = vectorizer.fit_transform(fanny['CONTENT']) #untuk
      melakukan pemecahan data pada dataframe yang terdapat pada
      kolom konten

```

```

12 dvec #Untuk menampilkan hasil dari code sebelumnya
13
14 Daptarkata= vectorizer.get_feature_names()
15
16 dshuf = fanny.sample(frac=1)
17
18 d_train=dshuf[:300]
19 d_test=dshuf[300:]
20
21 d_train_att = vectorizer.fit_transform(d_train['CONTENT'])
22 d_train_att
23
24 d_train_label=d_train['CLASS']
25 d_test_label=d_test['CLASS']

```

Dengan menggunakan $1174069 \bmod 4$ adalah 1, yang artinya menggunakan dataset LMFAO. Hasilnya adalah sebagai berikut :

fanny - DataFrame					
Index	COMMENT_ID	AUTHOR	DATE	CONTENT	CLASS
0	r13au1nhepm...	Cory Wilson	2015-09-28T2...	ca...nt...http://	0
1	r124yvrcxsl...	Epic Gaming	2015-09-28T2...	view but /	0
2	r13ttc3jy3R...	Lsd Music	2015-09-28T2...	Hey guys... i ha...	1
3	r13t3m0npma...	Oney3 Fox	2015-09-28T2...	Party	0
4	r13evcixek...	PATICK_TV	2015-09-28T2...	Party rock	0
5	r13nxlymew...	Brian Brasl	2015-09-28T0...	ShutUp!	0
6	r13betkll0ov...	Brian Brasl	2015-09-28T0...	Omg	0
7	r133i0n0odqu...	Alex Defeo	2015-09-28T0...	This song is...	0
8	r13geneket...	Glowand...	2015-09-28T0...	Just like you...	0
9	r13lmmcr0n...	Silvia Basso	2015-09-28T0...	wow!!!!!!	0
10	r13ctvFw4n...	Michael	2015-09-28T0...	I love this	0
11	r13qgk4f4n...	whitehorse	2015-09-28T0...	song so much	0
12	r13qgk4f4n...	rodrigued	2015-09-28T0...	I kiss when	0
13	r13tr3vztr...	Alex Martin	2015-09-28T0...	people dress,	0
14	r13lZm2tr7rv...	Alex Jansen	2015-09-28T0...	last year or,	0
15	r13mkfz3k0y...	Joe Miller	2015-09-27T2...	ever!!!!	0
16	r13tr7gjx0n...	lejanc	2015-09-27T2...	love music	0
17	r13qrgjx0n...	vlad kerec	2015-09-27T2...	monstros rock	0
18	r13qrgjx0n...	SoulInDust...	2015-09-27T2...	(S)	0
19	r13dyqgn0n...	Nguchi Ngaci	2015-09-27T2...	you watched ..	0
20	r13wq3q0n...	Ferrare	2015-09-27T2...	incredible	0
LMFAO!!!!!!					

Gambar 4.10 Hasil Soal 3.

4. Soal 4

```

1 #%% Soal 4
2
3 from sklearn import svm
4 clfsvm = svm.SVR(gamma = 'auto')
5 clfsvm.fit(d_train_att , d_train_label)

```

Klasifikasi dari data vektorisasi menggunakan klasifikasi Decision Tree. Hasilnya adalah sebagai berikut :

```

In [7]: from sklearn import svm
...: clfsvm = svm.SVR(gamma = 'auto')
...: clfsvm.fit(d_train_att , d_train_label)
Out[7]:
SVR(C=1.0, cache_size=200, cache_size=200, cache_size=200, cache_size=200,
     class_weight=None, degree=3, epsilon=0.1,
     gamma='auto', kernel='rbf', max_iter=-1, shrinking=True, tol=0.001,
     verbose=False)

```

Gambar 4.11 Hasil Soal 4.

5. Soal 5

```

1 #%%soal 5
2
3 from sklearn import tree
4 clftree = tree.DecisionTreeClassifier()
5 clftree.fit(d_train_att, d_train_label)

```

Plotlah confusion matrix dari praktik modul ini menggunakan matplotlib.
Hasilnya adalah sebagai berikut :

```

In [8]: from sklearn import tree
...: clftree = tree.DecisionTreeClassifier()
Out[8]: DecisionTreeClassifier(ccp_alpha=0.0, class_weight=None,
criterion='gini',
max_depth=None, max_features=None,
max_leaf_nodes=None,
min_impurity_decrease=0.0,
min_impurity_split=None,
min_samples_leaf=1, min_samples_split=2,
min_weight_fraction_leaf=0.0,
presort='deprecated',
random_state=None, splitter='best')

```

Gambar 4.12 Hasil Soal 5.

6. Soal 6

```

1 #%%soal 6/
2
3 from sklearn.metrics import confusion_matrix
4 pred_labels=clftree.predict(d_test)
5 cm=confusion_matrix(d_test_label, pred_labels)
6
7 #%%
8
9 import matplotlib.pyplot as plt
10
11 def plot_confusion_matrix(cm, classes,
12                           normalize=False,
13                           title='Confusion matrix',
14                           cmap=plt.cm.Blues):
15
16     if normalize:
17         cm = cm.astype('float') / cm.sum(axis=1)[:, np.
18         newaxis]
19         print("Normalized confusion matrix")
20     else:
21         print('Confusion matrix, without normalization')
22
23     print(cm)

```

Menjalankan program cross validation. Hasilnya adalah sebagai berikut :

```
In [11]: import matplotlib.pyplot as plt
...
...: def plot_confusion_matrix(cm, classes,
...:                         normalize=False,
...:                         title='Confusion matrix',
...:                         cmap=plt.cm.Blues):
...:     """
...:     If normalize is True, cm = cm.astype('float') / cm.sum(axis=1)[:, np.newaxis]
...:     """
...:     if normalize:
...:         cm = cm.astype('float') / cm.sum(axis=1)[:, np.newaxis]
...:         print("Normalized confusion matrix")
...:     else:
...:         print('Confusion matrix, without normalization')
...:     print(cm)
...:     cm
```

Gambar 4.13 Hasil Soal 6.**7. Soal 7**

```
1 %%soal 7
2
3 from sklearn.model_selection import cross_val_score
4
5 scores=cross_val_score(clf, d_train_att, d_train_label, cv=5)
6
7 skor_rata2=scores.mean()
8 skoresd=scores.std()
```

Tree.export graphviz merupakan fungsi yang menghasilkan representasi Graphviz dari decision tree, yang kemudian ditulis ke outfile. Disini akan menyimpan classifiernya, akan meng ekspor file student performance jika salah akan mengembalikan nilai fail. Hasilnya adalah sebagai berikut :

```
In [12]: from sklearn.model_selection import cross_val_score
...
...: scores=cross_val_score(clf, d_train_att, d_train_label, cv=5)
...: skor_rata2=scores.mean()
...: skoresd=scores.std()
```

Gambar 4.14 Hasil Soal 7.**8. Soal 8**

```
1 %%soal 8 /
2
3 max_features_opts = range(5, 50, 5) #max_features_opts
4     sebagai variabel untuk membuat range 5,50,5
5 n_estimators_opts = range(10, 200, 20) #n_estimators_opts
6     sebagai variabel untuk membuat range 10,200,20
7 rf_params = fanny.empty((len(max_features_opts)*len(
8         n_estimators_opts),4), float) #rf_params sebagai variabel
9     untuk menjumlahkan yang sudah di tentukan sebelumnya
10 i = 0
11 for max_features in max_features_opts: #pengulangan
12     for n_estimators in n_estimators_opts: #pengulangan
13         clf = RandomForestClassifier(max_features=
14             max_features, n_estimators=n_estimators) #menampilkan
15             variabel csf
16             scores = cross_val_score(clf, df_train_att,
17             df_train_label, cv=5) #scores sebagai variabel training
```

```

11     rf_params[1,0] = max_features #index 0
12     rf_params[1,1] = n_estimators #index 1
13     rf_params[1,2] = scores.mean() #index 2
14     rf_params[1,3] = scores.std() * 2 #index 3
15     i += 1 #dengan ketentuan i += 1
16     print("Max features: %d, num estimators: %d, accuracy
17 : %0.2f (+/- %0.2f)" %(max_features, n_estimators, scores.
mean(), scores.std() * 2))
    #print hasil pengulangan yang sudah ditentukan

```

Buatlah program pengamatan komponen informasi. Jadi disini kita akan memprediksi nilai dari variabel test att dan test pass Hasilnya adalah sebagai berikut :

Gambar 4.15 Hasil Soal 8.

4.2.3 Penanganan Error

1. ScreenShoot Error

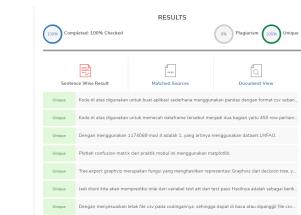
```
FileNotFoundException: [Errno 2] File b'F:/Semester 6/Artificial Intelligence/Tugas 4/src/funny.csv' does not exist: b'F:/Semester 6/Artificial Intelligence/Tugas 4/src/funny.csv'
```

2. Cara Penangan Error

- SyntaxError

Dengan menyesuaikan letak file csv pada codingannya, sehingga dapat di baca atau dipanggil file csvnya.

4.2.4 Bukti Tidak Plagiat



Gambar 4.17 Bukti Tidak Melakukan Plagiat Chapter 4

4.2.5 Link Youtube

<https://youtu.be/X-xd9Nb78Gs>

4.3 1174070 - Arrizal Furqona Gifary

4.3.1 Teori

1. Jelaskan apa itu klasifikasi teks, sertakan gambar ilustrasi buatan sendiri.

Klasifikasi teks merupakan sebuah model yang biasa digunakan untuk untuk mengkategorikan sebuah teks ke dalam kelompok-kelompok yang lebih terorganisir. Jadi untuk setiap kalimat yang di masukan ke dalam mesin, mesin tersebut akan menjadikan setiap kata dari kalimat tersebut menjadi sebuah kolom. Untuk ilustrasinya bisa dilihat pada gambar berikut :



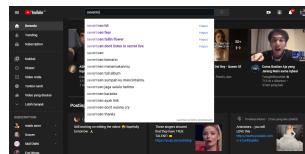
Gambar 4.18 Klasifikasi teks.

2. Jelaskan mengapa hal ini bisa terjadi, klasifikasi bunga tidak bisa digunakan untuk machine learning, sertakan ilustrasi gambar sendiri. Karena machine learning tidak dapat menampilkan inputan sesuai dengan apa yang kita inputkan. Karena inputan tersebut serupa namun mesin memberikan output yang berbeda, biasanya output atau error ini disebut dengan istilah noise. Untuk contoh sederhananya misalkan kita inputkan salah satu label yang terdapat pada bunga, output yang dihasilkan oleh mesin tersebut ialah label yang lain. Itu dikarenakan bunga banyak jenis yang serupa namun tidak sama. Untuk ilustrasinya bisa dilihat pada gambar berikut :



Gambar 4.19 Klasifikasi Bunga.

- Jelaskan bagaimana yang dimaksud dengan teknik pembelajaran mesin pada teks yang digunakan dan sertakan ilustrasi buatan sendiri.
Teknik yang digunakan pada youtube salah satunya ialah keywords. Dengan keywords tersebut mesin dapat memberikan video sesuai dengan keyword yang kita inputkan pada kolom pencarian. Teknik pembelajarannya tergantung user memberikan input teks seperti apa, karena pada youtube itu sendiri akan menyesuaikan dengan apa yang biasa kita inputkan dan akan memfilter video secara otomatis sesuai dengan keyword yang biasa kita inputkan. Contoh ilustrasi sederhananya seperti berikut :



Gambar 4.20 Klasifikasi teks Youtube.

- Jelaskan apa yang dimaksud vektorisasi data.
Vektorisasi data ialah suatu pemecahan atau pembagian data berupa teks, sebagai contoh terdapat 5 paragraf, data teks tersebut di pecah menjadi kalimat-kalimat yang lebih sederhana, lalu di pecah lagi menjadi kata untuk setiap kalimatnya.
- Jelaskan apa yang dimaksud dengan bag of words dengan ilustrasi sendiri.

Representasi penyederhanaan sebuah kalimat atau perhitungan setiap kata pada suatu kalimat dengan presentase berapa kali muncul kata tersebut untuk setiap kalimatnya. Contoh ilustrasi sederhananya seperti berikut :



Gambar 4.21 Bag of words.

- Jelaskan apa yang dimaksud dengan TF-IDF.
TF-IDF merupakan metode untuk menghitung bobot setiap kata pada

suatu kalimat yang paling sering digunakan. TF-IDF ini akan menghitung nilai Term Frequency dan Inverse Document Frequency pada setiap kata dalam setiap kalimat yang muncul dengan diimbangi dengan jumlah dokumen dalam korpus yang mengandung kata. Contoh ilustrasi sederhananya seperti gambar berikut :

$$\text{tfidf}(t, d, D) = \text{tf}(t, d) \times \text{idf}(t, D)$$

	Doc 1	Doc 2	...	Doc n
Term(s) 1	12	2	...	1
Term(s) 2	0	1	...	0
...
Term(s) n	0	6	...	3

Gambar 4.22 TF-IDF.

4.3.2 Praktek Program

1. Soal 1

```

1 %% Soal 1
2 import pandas as pd #digunakan untuk mengimport library
# pandas dengan alias pd
3 pd = pd.read_csv("F:/Semester 6/Artificial Intelligence/
#Tugas 4/src/csv_izal.csv") #membaca file csv

```

Kode di atas digunakan untuk buat aplikasi sederhana menggunakan pandas dengan format csv sebanyak 500 baris, hasilnya ialah sebagai berikut :



Gambar 4.23 Hasil Soal 1.

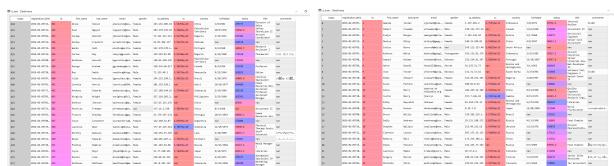
2. Soal 2

```

1 %% Soal 2
2 d_train=pd[:450] #membagi data training menjadi 450
3 d_test=pd[450:] #membagi data menjadi 50 atau sisa dari data
# yang tersedia

```

Kode di atas digunakan untuk memecah dataframe tersebut menjadi dua bagian yaitu 450 row pertama dan 50 row kedua. Hasilnya adalah sebagai berikut :



Gambar 4.24 Hasil Soal 2.

3. Soal 3

```

1 %% Soal 3
2 import pandas as izal #untuk import library pandas berguna
# untuk mengelola dataframe
3 izal = izal.read_csv("F:// Semester 6/Artificial Intelligence/
# Tugas 4/src/Youtube03-LMFAO.csv") #membaca file dengan
# format csv
4
5 spam=izal.query('CLASS == 1') #membagi tabel spam
6 nospam=izal.query('CLASS == 0')#membagi tabel no spam
7
8 from sklearn.feature_extraction.text import CountVectorizer #
# untuk import countvectorizer berfungsi untuk memecah data
# tersebut menjadi sebuah kata yang lebih sederhana
9 vectorizer = CountVectorizer () #ntuk menjalankan fungsi
# tersebut , pada code ini tidak ada hasilnya dikarenakan
# spyder tidak mendukung hasil dari instasiasi.
10
11 dvec = vectorizer.fit_transform(izal[ 'CONTENT' ]) #untuk
# melakukan pemecahan data pada dataframe yang terdapat pada
# kolom konten
12 dvec #Untuk menampilkan hasil dari code sebelumnya
13
14 Daptarkata= vectorizer.get_feature_names()
15
16 dshuf = izal.sample(frac=1)
17
18 d_train=dshuf[:300]
19 d_test=dshuf[300:]
20
21 d_train_att = vectorizer.fit_transform(d_train[ 'CONTENT' ])
22 d_train_att
23
24 d_train_label=d_train[ 'CLASS' ]
25 d_test_label=d_test[ 'CLASS' ]

```

Dengan menggunakan $1174070 \bmod 4$ adalah 1, yang artinya menggunakan dataset LMFAO. Hasilnya adalah sebagai berikut :

Genre - DataName						
Index	Comment_ID	AUTHOR	DATE	CONTENT	CLASS	
0	113wunfghd	Cory Wilson	2010-05-28T12:00:00	here's a href="http://www.	4	
1	1134zcrxh3d	Talia Goming	2010-05-28T12:00:00	funny	4	
2	113t3yjy3n	Luis Munic	2010-05-28T12:00:00	that's it! I'm a noob	4	
3	113t10hnpq	Ferny Fox	2010-05-28T12:00:00	Rock...lol :)	4	
4	113p0c14ed	PATRICKU	2010-05-28T12:00:00	Parry Rock	4	
5	113t3yjy3n	Brian Brat	2010-05-28T12:00:00	Shuttle	4	
6	113t3yjy3n	Brian Brat	2010-05-28T12:00:00	Ong	4	
7	113t3yjy3n	Alex Defer	2010-05-28T12:00:00	This song is	4	
8	113t3yjy3n	Giovanni	2010-05-28T12:00:00	so real	4	
9	113t3yjy3n	Silvia Bresser	2010-05-28T12:00:00	Awesome :)	4	
10	113t3yjy3n	Himawari	2010-05-28T12:00:00	I love this	4	
11	113t3yjy3n	Yuri mafra	2010-05-28T12:00:00	song	4	
12	113t3yjy3n	Alvaro	2010-05-28T12:00:00	2011 LIEEEF	4	
13	113t3yjy3n	Alex Martin	2010-05-28T12:00:00	i add this	4	
14	113t3yjy3n	people dress	2010-05-28T12:00:00	up like this	4	
15	113t3yjy3n	Alex Jansen	2010-05-28T12:00:00	last year of	4	
16	113t3yjy3n	Alexander will	2010-05-27T12:00:00	school ever!!	4	
17	113t3yjy3n	Joe peplin	2010-05-27T12:00:00	so funny!!!	4	
18	113t3yjy3n	lebianc	2010-05-27T12:00:00	give mice	4	
19	113t3yjy3n	luha kerino	2010-05-27T12:00:00	sooooooo	4	
20	113t3yjy3n	SoulInCloud	2010-05-27T12:00:00	(8)	4	
21	113t3yjy3n	Ricardo	2010-05-27T12:00:00	you're so in	4	
22	113t3yjy3n	Nguyen Ngoc	2010-05-27T12:00:00	what you watched	4	
23	113t3yjy3n	lizlalopam	2010-05-27T12:00:00	so weird!	4	
24	113t3yjy3n	Perseverance	2010-05-27T12:00:00	incredible!	4	
25	113t3yjy3n	utara_07	2010-05-27T12:00:00	LOL!!!!!!	4	

Gambar 4.25 Hasil Soal 3.

4. Soal 4

```
1 %% Soal 4
2
3 from sklearn import svm
4 clfsvm = svm.SVR(gamma = 'auto')
5 clfsvm.fit(d_train_att, d_train_label)
```

Klasifikasi dari data vektorisasi menggunakan klasifikasi Decision Tree. Hasilnya adalah sebagai berikut :

```
In [7]: from sklearn import svm
      ...: clfsvm = svm.SVR(gamma= 'auto')
      ...: clfsvm.fit(d_train_attr, d_train_label)
Out[7]:
SVR(C=1.0, cache_size=200, coef0=0.0, degree=3, epsilon=0.1,
gamma='auto',
      kernel='rbf', max_iter=-1, shrinking=True, tol=0.001,
verbose=False)
```

Gambar 4.26 Hasil Soal 4.

5. Soal 5

```
1 #%%soal 5
2
3 from sklearn import tree
4 clftree = tree.DecisionTreeClassifier()
5 clftree.fit(d_train_att, d_train_label)
```

Plotlah confusion matrix dari praktik modul ini menggunakan matplotlib. Hasilnya adalah sebagai berikut :

```
In [8]: from sklearn import tree
       clftree = tree.DecisionTreeClassifier()
       ...; clftree.fit(d_train_att, d_train_label)
Out[8]: DecisionTreeClassifier(ccp_alpha=0.0, class_weight=None,
                           criterion='gini',
                           max_depth=None, max_features=None,
                           max_leaf_nodes=None, min_impurity_decrease=0.0,
                           min_impurity_split=None,
                           min_samples_leaf=1, min_samples_split=2,
                           min_weight_fraction_leaf=0.0,
                           presort='deprecated',
                           random_state=None, splitter='best')
```

Gambar 4.27 Hasil Soal 5.

6. Soal 6

```
1 #%%soal 6/
2
3 from sklearn.metrics import confusion_matrix
4 pred_labels=clftree.predict(d_test)
5 cm=confusion_matrix(d_test_label ,pred_labels)
6
7 #%%
8
9 import matplotlib.pyplot as plt
10
11 def plot_confusion_matrix(cm, classes,
12                           normalize=False,
13                           title='Confusion matrix',
14                           cmap=plt.cm.Blues):
15
16     if normalize:
17         cm = cm.astype('float') / cm.sum(axis=1)[:, np.
newaxis]
18         print("Normalized confusion matrix")
19     else:
20         print('Confusion matrix, without normalization')
21
22     print(cm)
```

Menjalankan program cross validation. Hasilnya adalah sebagai berikut :

```
In [11]: import matplotlib.pyplot as plt
...
...: def plot_confusion_matrix(cm, classes,
...:                          normalize=False,
...:                          title='Confusion matrix',
...:                          cmap=plt.cm.Blues):
...:
...:     if normalize:
...:         cm = cm.astype('float') / cm.sum(axis=1)[:, np.
newaxis]
...:         print("Normalized confusion matrix")
...:     else:
...:         print('Confusion matrix, without normalization')
...:
...:     print(cm)
...:     cm
```

Gambar 4.28 Hasil Soal 6.

7. Soal 7

```
1 #%%soal 7
2
```

```

3 from sklearn.model_selection import cross_val_score
4
5 scores=cross_val_score(clftree,d_train_att,d_train_label, cv
6 =5)
7
8 skor_rata2=scores.mean()
9 skoresd=scores.std()

```

Tree.export graphviz merupakan fungsi yang menghasilkan representasi Graphviz dari decision tree, yang kemudian ditulis ke outfile. Disini akan menyimpan classifiernya, akan meng ekspor file student performance jika salah akan mengembalikan nilai fail. Hasilnya adalah sebagai berikut :

```

In [12]: from sklearn.model_selection import cross_val_score
...:
scores=cross_val_score(clftree,d_train_att,d_train_label, cv=5)
...:
skor_rata2=scores.mean()
...:
skoresd=scores.std()

```

Gambar 4.29 Hasil Soal 7.

8. Soal 8

```

1 %%soal 8 /
2
3 max_features_opts = range(5, 50, 5) #max_features_opts
4 sebagai variabel untuk membuat range 5,50,5
5 n_estimators_opts = range(10, 200, 20) #n_estimators_opts
6 sebagai variabel untuk membuat range 10,200,20
7 rf_params = izal.empty((len(max_features_opts)*len(
8     n_estimators_opts),4), float) #rf_params sebagai variabel
9 untuk menjumlahkan yang sudah di tentukan sebelumnya
10 i = 0
11 for max_features in max_features_opts: #pengulangan
12     for n_estimators in n_estimators_opts: #pengulangan
13         clftree = RandomForestClassifier(max_features=
14             max_features, n_estimators=n_estimators) #menampilkan
15         variabel csf
16         scores = cross_val_score(clf, df_train_att,
17             df_train_label, cv=5) #scores sebagai variabel training
18         rf_params[i,0] = max_features #index 0
19         rf_params[i,1] = n_estimators #index 1
20         rf_params[i,2] = scores.mean() #index 2
21         rf_params[i,3] = scores.std() * 2 #index 3
22         i += 1 #dengan ketentuan i += 1
23         print("Max features: %d, num estimators: %d, accuracy
24 : %0.2f (+/- %0.2f)" %(max_features, n_estimators, scores.
25 mean(), scores.std() * 2))
26         #print hasil pengulangan yang sudah ditentukan

```

Buatlah program pengamatan komponen informasi. Jadi disini kita akan memprediksi nilai dari variabel test att dan test pass Hasilnya adalah sebagai berikut :

```

Max features: 25, num estimators: 100, accuracy: 0.46 (+/- 0.03)
Max features: 30, num estimators: 100, accuracy: 0.33 (+/- 0.02)
Max features: 35, num estimators: 100, accuracy: 0.43 (+/- 0.03)
Max features: 36, num estimators: 70, accuracy: 0.44 (+/- 0.03)
Max features: 36, num estimators: 100, accuracy: 0.44 (+/- 0.03)
Max features: 36, num estimators: 130, accuracy: 0.45 (+/- 0.03)
Max features: 38, num estimators: 130, accuracy: 0.46 (+/- 0.03)
Max features: 38, num estimators: 150, accuracy: 0.46 (+/- 0.03)
Max features: 38, num estimators: 170, accuracy: 0.46 (+/- 0.03)
Max features: 38, num estimators: 180, accuracy: 0.34 (+/- 0.03)
Max features: 35, num estimators: 100, accuracy: 0.34 (+/- 0.03)
Max features: 35, num estimators: 130, accuracy: 0.43 (+/- 0.03)
Max features: 35, num estimators: 150, accuracy: 0.44 (+/- 0.03)
Max features: 35, num estimators: 170, accuracy: 0.44 (+/- 0.03)
Max features: 35, num estimators: 180, accuracy: 0.44 (+/- 0.03)
Max features: 35, num estimators: 190, accuracy: 0.46 (+/- 0.04)
Max features: 35, num estimators: 200, accuracy: 0.45 (+/- 0.03)
Max features: 35, num estimators: 210, accuracy: 0.46 (+/- 0.03)
Max features: 35, num estimators: 220, accuracy: 0.46 (+/- 0.04)
Max features: 35, num estimators: 230, accuracy: 0.46 (+/- 0.04)
Max features: 35, num estimators: 240, accuracy: 0.46 (+/- 0.04)
Max features: 35, num estimators: 250, accuracy: 0.46 (+/- 0.04)
Max features: 35, num estimators: 260, accuracy: 0.46 (+/- 0.04)
Max features: 35, num estimators: 270, accuracy: 0.46 (+/- 0.04)
Max features: 35, num estimators: 280, accuracy: 0.46 (+/- 0.04)
Max features: 35, num estimators: 290, accuracy: 0.45 (+/- 0.04)
Max features: 35, num estimators: 300, accuracy: 0.46 (+/- 0.04)
Max features: 35, num estimators: 310, accuracy: 0.46 (+/- 0.04)
Max features: 35, num estimators: 320, accuracy: 0.46 (+/- 0.04)
Max features: 35, num estimators: 330, accuracy: 0.46 (+/- 0.04)
Max features: 35, num estimators: 340, accuracy: 0.46 (+/- 0.04)
Max features: 35, num estimators: 350, accuracy: 0.46 (+/- 0.04)
Max features: 35, num estimators: 360, accuracy: 0.46 (+/- 0.04)
Max features: 35, num estimators: 370, accuracy: 0.46 (+/- 0.04)
Max features: 35, num estimators: 380, accuracy: 0.46 (+/- 0.04)
Max features: 35, num estimators: 390, accuracy: 0.46 (+/- 0.04)
Max features: 35, num estimators: 400, accuracy: 0.46 (+/- 0.04)
Max features: 35, num estimators: 410, accuracy: 0.46 (+/- 0.04)
Max features: 35, num estimators: 420, accuracy: 0.46 (+/- 0.04)
Max features: 35, num estimators: 430, accuracy: 0.46 (+/- 0.04)
Max features: 35, num estimators: 440, accuracy: 0.46 (+/- 0.04)
Max features: 35, num estimators: 450, accuracy: 0.46 (+/- 0.04)
Max features: 35, num estimators: 460, accuracy: 0.46 (+/- 0.04)
Max features: 35, num estimators: 470, accuracy: 0.46 (+/- 0.04)
Max features: 35, num estimators: 480, accuracy: 0.46 (+/- 0.04)
Max features: 35, num estimators: 490, accuracy: 0.46 (+/- 0.04)
Max features: 35, num estimators: 500, accuracy: 0.43 (+/- 0.03)
Max features: 45, num estimators: 70, accuracy: 0.44 (+/- 0.03)
Max features: 45, num estimators: 100, accuracy: 0.44 (+/- 0.03)
Max features: 45, num estimators: 130, accuracy: 0.45 (+/- 0.04)
Max features: 45, num estimators: 150, accuracy: 0.45 (+/- 0.03)
Max features: 45, num estimators: 170, accuracy: 0.45 (+/- 0.04)
Max features: 45, num estimators: 180, accuracy: 0.34 (+/- 0.03)
Max features: 45, num estimators: 190, accuracy: 0.43 (+/- 0.03)
Max features: 45, num estimators: 200, accuracy: 0.44 (+/- 0.03)
Max features: 45, num estimators: 210, accuracy: 0.44 (+/- 0.03)
Max features: 45, num estimators: 220, accuracy: 0.44 (+/- 0.03)
Max features: 45, num estimators: 230, accuracy: 0.44 (+/- 0.03)
Max features: 45, num estimators: 240, accuracy: 0.44 (+/- 0.03)
Max features: 45, num estimators: 250, accuracy: 0.44 (+/- 0.03)
Max features: 45, num estimators: 260, accuracy: 0.44 (+/- 0.03)
Max features: 45, num estimators: 270, accuracy: 0.44 (+/- 0.03)
Max features: 45, num estimators: 280, accuracy: 0.44 (+/- 0.03)
Max features: 45, num estimators: 290, accuracy: 0.45 (+/- 0.03)
Max features: 45, num estimators: 300, accuracy: 0.45 (+/- 0.03)
Max features: 45, num estimators: 310, accuracy: 0.45 (+/- 0.03)
Max features: 45, num estimators: 320, accuracy: 0.45 (+/- 0.03)
Max features: 45, num estimators: 330, accuracy: 0.45 (+/- 0.03)
Max features: 45, num estimators: 340, accuracy: 0.45 (+/- 0.03)
Max features: 45, num estimators: 350, accuracy: 0.45 (+/- 0.03)
Max features: 45, num estimators: 360, accuracy: 0.45 (+/- 0.03)
Max features: 45, num estimators: 370, accuracy: 0.45 (+/- 0.03)
Max features: 45, num estimators: 380, accuracy: 0.45 (+/- 0.03)
Max features: 45, num estimators: 390, accuracy: 0.46 (+/- 0.04)
Max features: 45, num estimators: 400, accuracy: 0.46 (+/- 0.04)
Max features: 45, num estimators: 410, accuracy: 0.46 (+/- 0.04)
Max features: 45, num estimators: 420, accuracy: 0.46 (+/- 0.04)
Max features: 45, num estimators: 430, accuracy: 0.46 (+/- 0.04)
Max features: 45, num estimators: 440, accuracy: 0.46 (+/- 0.04)
Max features: 45, num estimators: 450, accuracy: 0.46 (+/- 0.04)
Max features: 45, num estimators: 460, accuracy: 0.46 (+/- 0.04)
Max features: 45, num estimators: 470, accuracy: 0.46 (+/- 0.04)
Max features: 45, num estimators: 480, accuracy: 0.46 (+/- 0.04)
Max features: 45, num estimators: 490, accuracy: 0.46 (+/- 0.04)
Max features: 45, num estimators: 500, accuracy: 0.43 (+/- 0.03)

```

Gambar 4.30 Hasil Soal 8.

4.3.3 Penanganan Error

1. ScreenShoot Error

FileNotFoundError: [Errno 2] File b'F:/Semester 6/Artificial Intelligence/Tugas 4/src/fancy.csv' does not exist: b'F:/Semester 6/Artificial Intelligence/Tugas 4/src/fancy.csv'

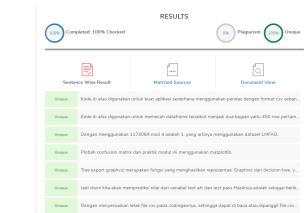
Gambar 4.31 SyntaxError

2. Cara Penanganan Error

- SyntaxError

Dengan menyesuaikan letak file csv pada codingannya, sehingga dapat di baca atau dipanggil file csvnya.

4.3.4 Bukti Tidak Plagiat



Gambar 4.32 Bukti Tidak Melakukan Plagiat Chapter 4

4.3.5 Link Youtube

BAB 5

CHAPTER 5

5.1 1174006 - Kadek Diva Krishna Murti

Lorem ipsum dolor sit amet, consectetur adipiscing elit.

```
1 @inproceedings{awangga2017colenak,
2   title={Colenak: GPS tracking model for post-stroke
3   rehabilitation program using AES-CBC URL encryption and QR-
4   Code},
5   author={Awangga, Rolly Maulana and Fathonah, Nuraini Siti and
6   Hasanudin, Trisna Irmayadi},
7   booktitle={Information Technology, Information Systems and
8   Electrical Engineering (ICITISEE), 2017 2nd International
   conferences on},
9   pages={255--260},
10  year={2017},
11  organization={IEEE}
12 }
```



Gambar 5.1 Kecerdasan Buatan.

1. Lorem ipsum dolor sit amet, consectetur adipiscing elit.
2. Lorem ipsum dolor sit amet, consectetur adipiscing elit.
3. Lorem ipsum dolor sit amet, consectetur adipiscing elit.

5.1.1 Teori

5.1.2 Praktek

5.1.3 Penanganan Error

5.1.4 Bukti Tidak Plagiat



Gambar 5.2 Kecerdasan Buatan.

BAB 6

CHAPTER 6

6.1 1174006 - Kadek Diva Krishna Murti

Lorem ipsum dolor sit amet, consectetur adipiscing elit.

```
1 @inproceedings{awangga2017colenak,
2   title={Colenak: GPS tracking model for post-stroke
3   rehabilitation program using AES-CBC URL encryption and QR-
4   Code},
5   author={Awangga, Rolly Maulana and Fathonah, Nuraini Siti and
6   Hasanudin, Trisna Irmayadi},
7   booktitle={Information Technology, Information Systems and
8   Electrical Engineering (ICITISEE), 2017 2nd International
   conferences on},
9   pages={255--260},
10  year={2017},
11  organization={IEEE}
12 }
```



Gambar 6.1 Kecerdasan Buatan.

1. Lorem ipsum dolor sit amet, consectetur adipiscing elit.
2. Lorem ipsum dolor sit amet, consectetur adipiscing elit.
3. Lorem ipsum dolor sit amet, consectetur adipiscing elit.

6.1.1 Teori

6.1.2 Praktek

6.1.3 Penanganan Error

6.1.4 Bukti Tidak Plagiat



Gambar 6.2 Kecerdasan Buatan.

BAB 7

CHAPTER 7

7.1 1174006 - Kadek Diva Krishna Murti

Lorem ipsum dolor sit amet, consectetur adipiscing elit.

```
1 @inproceedings{awangga2017colenak,
2   title={Colenak: GPS tracking model for post-stroke
3   rehabilitation program using AES-CBC URL encryption and QR-
4   Code},
5   author={Awangga, Rolly Maulana and Fathonah, Nuraini Siti and
6   Hasanudin, Trisna Irmayadi},
7   booktitle={Information Technology, Information Systems and
8   Electrical Engineering (ICITISEE), 2017 2nd International
   conferences on},
9   pages={255--260},
10  year={2017},
11  organization={IEEE}
12 }
```



Gambar 7.1 Kecerdasan Buatan.

1. Lorem ipsum dolor sit amet, consectetur adipiscing elit.
2. Lorem ipsum dolor sit amet, consectetur adipiscing elit.
3. Lorem ipsum dolor sit amet, consectetur adipiscing elit.

7.1.1 Teori

7.1.2 Praktek

7.1.3 Penanganan Error

7.1.4 Bukti Tidak Plagiat



Gambar 7.2 Kecerdasan Buatan.