

Modul Praktikum Kecerdasan Buatan



Rolly Maulana Awangga
0410118609

Applied Bachelor of Informatics Engineering
Program Studi D4 Teknik Informatika

Applied Bachelor Program of Informatics Engineering
Politeknik Pos Indonesia

Bandung 2019

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

Acknowledgements

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

Abstract

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

Contents

1	Mengenal Kecerdasan Buatan dan Scikit-Learn	1
1.1	Teori	1
1.2	Instalasi	2
1.3	Penanganan Error	2
1.4	Ahmad Syafrizal Huda/1164062	2
1.4.1	Teori	2
1.4.2	Instalasi	4
1.4.2.1	Instalasi Library Scikit dari Anaconda	4
1.4.2.2	Mencoba Loading an example Dataset	4
1.4.2.3	Learning and Predicting	5
1.4.2.4	Model Presistence	5
1.4.2.5	Conventions	7
1.4.3	Penanganan eror	10
1.4.3.1	ScreenShoot Error	10
1.4.3.2	Tuliskan Kode Error dan Jenis Erornya	11
1.4.3.3	Solusi Pemecahan Masalah Error	11
1.5	Cokro Edi Prawiro/1164069	12
1.5.1	Praktek teori penunjang	12
1.5.2	Instalisasi	18
1.6	Fathi Rabbani / 1164074	31
1.6.1	Teori	31
1.6.2	Praktikum	32
1.6.3	Penanganan Error	37
2	Related Works	49
2.1	Cokro Edi Prawiro/ 1164069	49
2.1.1	Teori	49
2.1.2	Sikic-Learn /Cokro Edi Prawiro/1164069	51

2.1.3	Penanganan Error /Cokro Edi Prawiro/1164069	58
2.2	Ahmad Syafrizal Huda/1164062	59
2.2.1	Teori	59
2.2.2	Scikit-learn	62
2.2.3	Penanganan Error	66
2.3	Fathi Rabbani / 1164074	68
2.3.1	Teori	68
2.3.2	Praktek	71
2.3.3	Penanganan Error	77
3	Methods	100
3.1	Fathi Rabbani / 1164074	100
3.1.1	Teori	100
4	Experiment and Result	106
4.1	Experiment	106
4.2	Result	106
5	Conclusion	107
5.1	Conclusion of Problems	107
5.2	Conclusion of Method	107
5.3	Conclusion of Experiment	107
5.4	Conclusion of Result	107
6	Discussion	108
7	Discussion	109
8	Discussion	110
9	Discussion	111
10	Discussion	112
11	Discussion	113
12	Discussion	114
13	Discussion	115

14 Discussion	116
A Form Penilaian Jurnal	117
B FAQ	120
Bibliography	122

List of Figures

1.1	Hasil Tampilan Error.	10
1.2	Hasil Tampilan Install joblib.	12
1.3	Hasil Tampilan Uji coba perintah joblib.	12
1.4	Download Anaconda.	13
1.5	Langkah pertama instalasi anaconda.	13
1.6	Langkah kedua instalasi anaconda.	14
1.7	Langkah ketiga instalasi anaconda.	15
1.8	Langkah terakhir instalasi anaconda.	16
1.9	Langkah pertama instalasi scikit pada CMD.	16
1.10	Langkah kedua instalasi scikit pada CMD.	17
1.11	Langkah ketiga instalasi scikit pada CMD.	17
1.12	Langkah compile code pada python anaconda.	18
1.13	Hasil Tampilan 1.	18
1.14	Hasil Tampilan 2.	19
1.15	Hasil Tampilan 3.	19
1.16	Hasil Tampilan 4.	20
1.17	Hasil Tampilan 5.	20
1.18	Hasil Tampilan 6.	21
1.19	Hasil Tampilan 7.	21
1.20	Hasil Tampilan 8.	22
1.21	Hasil Tampilan 9.	22
1.22	Hasil Tampilan 10.	23
1.23	Hasil Tampilan 11.	23
1.24	Hasil Tampilan 12.	24
1.25	Hasil Tampilan 13.	24
1.26	Hasil Tampilan 14.	25
1.27	Hasil Tampilan 15.	25
1.28	Hasil Tampilan 16.	26

1.29 Hasil Tampilan 17.	26
1.30 Hasil Tampilan 18.	27
1.31 Hasil Tampilan 19.	27
1.32 Hasil Tampilan 20.	28
1.33 Hasil Tampilan 21.	28
1.34 Hasil Tampilan 22.	29
1.35 Tampilan website Scikit 1.	29
1.36 Tampilan website Scikit 2.	30
1.37 setelah membuka data instalasi klik next	39
1.38 pilih i agree	39
1.39 pilih instalasi Just Me	40
1.40 langsung saja next	40
1.41 cek kedua pilihan tersebut	41
1.42 proses Instalasi	41
1.43 klik next	42
1.44 selesai instalasi anaconda	42
1.45 Instalasi SCIKIT dengan menggunakan anaconda	43
1.46 Konfirmasi Instalasi	43
1.47 hasil dari instalasi SCIKIT	44
1.48 data variable explorer	44
1.49 code example dataset yang digunakan	44
1.50 data hasil dari code example dataset yang digunakan	44
1.51 Tampilan Versi Python dan Anaconda	45
1.52 Instalasi Library Sikic.	45
1.53 Instalasi Library Sikic Melalui Conda	46
1.54 Console Python Include Anaconda	46
1.55 Contoh Codingan Dataset	46
1.56 Error Coding 1	47
1.57 Error Coding 2	47
1.58 Codingan Solusi Untuk Error digits	47
1.59 Codingan Solusi Untuk Error Joblib	47
1.60 Hasil Solusi Error Joblib	48
1.61 Error Type data, yang harus digunakan number sedangkan isinya 'SCALE' pada gamma	48
1.62 Error no Module found, modul yang dicari tidak ditemukan atau tidak ada 'JOBLIB'	48

2.1	Source Code Load Dataset	52
2.2	Hasil Load Dataset	52
2.3	memberikan nilai satau atau nol	53
2.4	hasil dari memberikan nilai nol dan satu	53
2.5	Penambahan nilai numerik	54
2.6	hasil Penambahan nilai numerik	54
2.7	penentuan data training dan data testing	54
2.8	Hasil 1 penentuan data training dan data testing	55
2.9	hasil 2 penentuan data training dan data testing	55
2.10	memberikan nilai pada pohon keputusan	56
2.11	hasil memberikan nilai pada pohon keputusan	56
2.12	pembuatan diagram puhon keputusan	57
2.13	hasil pembuatan puhon keputusan	57
2.14	coding save	58
2.15	hasil coding save	58
2.16	Contoh Binary Classification	59
2.17	hasil coding save	59
2.18	Akurasi perhitungan pohon keputusan	60
2.19	hasil Akurasi perhitungan pohon keputusan	60
2.20	Contoh Binary Classification	61
2.21	hasil Akurasi perhitungan pohon keputusan	61
2.22	Contoh Binary Classification	62
2.23	hasil Akurasi perhitungan pohon keputusan	63
2.24	codingan pembuatan grafik	64
2.25	hasil grafik	65
2.26	Screensot error	66
2.27	Proses instalisasi graphviz	67
2.28	Proses instalisasi graphviz di user	68
2.29	Path Komputer	69
2.30	Memasukan Direktori Ke Path	70
2.31	Hasil Codingan No 1.	71
2.32	Hasil Codingan No 2.	71
2.33	Hasil Codingan No 3.	72
2.34	Hasil Codingan No 4.	73
2.35	Hasil Codingan No 5.	73
2.36	Hasil Codingan No 6.	74

2.37 Hasil Codingan No 7.	74
2.38 Hasil Codingan No 8.	75
2.39 Hasil Codingan No 9.	75
2.40 Hasil Codingan No 10.	76
2.41 Hasil Codingan No 11.	77
2.42 Hasil Codingan No 12.	78
2.43 Hasil Gambar Error No 6.	78
2.44 Hasil Gambar Penanganan Error No 6.	79
2.45 Binary Classification.	80
2.46 Supervised Learning.	81
2.47 Unsupervised Learning.	81
2.48 Clustering.	81
2.49 Evaluasi dan Akurasi.	82
2.50 K-fold Cross Validation.	82
2.51 Decision Tree.	83
2.52 Gain.	83
2.53 Contoh Binary Classification	84
2.54 Ilustrasi Suvervised Learning	85
2.55 Ilustrasi Unsuvervised Learning	86
2.56 Ilustrasi Clustering	86
2.57 Ilustrasi Evaluasi	87
2.58 Ilustrasi Confusion Matrix	88
2.59 Ilustrasi K-Fold	89
2.60 Ilustrasi Decision Tree	89
2.61 Contoh Ilustrasi Information Gain.	90
2.62 Contoh Penggunaan Binary Classification	90
2.63 Contoh Penggunaan Supervised Learning	91
2.64 Contoh Penggunaan Unsupervised Learning	91
2.65 Contoh Penggunaan Clustering	91
2.66 Contoh Penggunaan Evaluasi dan Akurasi	92
2.67 Contoh Matrix Confusion	92
2.68 Contoh Matrix Confusion	92
2.69 Contoh Penggunaan K Fold Cross Validation	93
2.70 Contoh Penggunaan Decision Tree	93
2.71 Contoh Penggunaan Information Gain	94
2.72 code 1 hasil	94

2.73	code 2 hasil	94
2.74	code 3 hasil	94
2.75	code 4 hasil	95
2.76	code 5 hasil	95
2.77	code 6 hasil	95
2.78	code 7 hasil	95
2.79	code 8 hasil	95
2.80	code 9 hasil	95
2.81	code 10 hasil	96
2.82	code 11 hasil	97
2.83	code 12 hasil	98
2.84	Error Path	98
2.85	Fix Error	99
3.1	Random Forest	103
3.2	Hasil dari membaca data dengan Confusion Matriks	104
3.3	Voting Random Forest	105
A.1	Form nilai bagian 1.	118
A.2	form nilai bagian 2.	119

Chapter 1

Mengenai Kecerdasan Buatan dan Scikit-Learn

Buku umum yang digunakan adalah [5] dan untuk sebelum UTS menggunakan buku *Python Artificial Intelligence Projects for Beginners*[2]. Dengan praktek menggunakan python 3 dan editor anaconda dan library python scikit-learn. Tujuan pembelajaran pada pertemuan pertama antara lain:

1. Mengerti definisi kecerdasan buatan, sejarah kecerdasan buatan, perkembangan dan penggunaan di perusahaan
2. Memahami cara instalasi dan pemakaian sci-kit learn
3. Memahami cara penggunaan variabel explorer di spyder

Tugas dengan cara dikumpulkan dengan pull request ke github dengan menggunakan latex pada repo yang dibuat oleh asisten riset.

1.1 Teori

Praktek teori penunjang yang dikerjakan :

1. Buat Resume Definisi, Sejarah dan perkembangan Kecerdasan Buatan, dengan bahasa yang mudah dipahami dan dimengerti. Buatan sendiri bebas plagiat[hari ke 1](10)
2. Buat Resume mengenai definisi supervised learning, klasifikasi, regresi dan unsupervised learning. Data set, training set dan testing set.[hari ke 1](10)

1.2 Instalasi

Membuka <https://scikit-learn.org/stable/tutorial/basic/tutorial.html>. Dengan menggunakan bahasa yang mudah dimengerti dan bebas plagiat. Dan wajib skrinsut dari komputer sendiri.

1. Instalasi library scikit dari anaconda, mencoba kompilasi dan uji coba ambil contoh kode dan lihat variabel explorer[hari ke 1](10)
2. Mencoba Loading an example dataset, menjelaskan maksud dari tulisan tersebut dan mengartikan per baris[hari ke 1](10)
3. Mencoba Learning and predicting, menjelaskan maksud dari tulisan tersebut dan mengartikan per baris[hari ke 2](10)
4. mencoba Model persistence, menjelaskan maksud dari tulisan tersebut dan mengartikan per baris[hari ke 2](10)
5. Mencoba Conventions, menjelaskan maksud dari tulisan tersebut dan mengartikan per baris[hari ke 2](10)

1.3 Penanganan Error

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

1. skrinsut error[hari ke 2](10)
2. Tuliskan kode eror dan jenis errornya [hari ke 2](10)
3. Solusi pemecahan masalah error tersebut[hari ke 2](10)

iiiiiii HEAD

1.4 Ahmad Syafrizal Huda/1164062

1.4.1 Teori

1. Definisi, sejarah, dan perkembangan kecerdasan buatan.

Definisi kecerdasan buatan adalah suatu pengetahuan yang dapat membuat komputer untuk meniru kecerdasan manusia yang berhubungan dengan

penangkapan, pemodelan, dan penyimpanan kecerdasan manusia dalam sebuah sistem teknologi. Contohnya yaitu melakukan analisa penalaran untuk mengambil suatu kesimpulan atau penerjemahan atau keputusan dari satu bahasa satu ke bahasa lain.

Sejarah dan perkembangan kecerdasan buatan terjadi pada musim panas tahun 1956 tercatat adanya seminar mengenai AI di Darmouth College. Seminar pada waktu itu dihadiri oleh sejumlah pakar komputer dan membahas potensi komputer dalam meniru kepandaian manusia. Akan tetapi perkembangan yang sering terjadi semenjak diciptakannya LISP, yaitu bahasa kecerdasan buatan yang dibuat tahun 1960 oleh John McCarthy. Istilah pada kecerdasan buatan atau Artificial Intelligence diambil dari Marvin Minsky dari MIT. Dia menulis karya ilmiah berjudul Step towards Artificial Intelligence, The Institute of radio Engineers Proceedings 49, January 1961[1].

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

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

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

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

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

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

Testing set digunakan untuk mengukur sejauh mana classifier berhasil melakukan klasifikasi dengan benar[6].

1.4.2 Instalasi

1.4.2.1 Instalasi Library Scikit dari Anaconda

1. Download aplikasi Anaconda terlebih dahulu. Lihat pada gambar 1.4.
2. Install aplikasi Anaconda yang sudah di download tadi. Lihat pada gambar 1.5.
3. Simpan aplikasi sesuai folder yang kita pilih lalu next. Lihat pada gambar 1.6.
4. Centang Keduanya lalu tekan tombol install. Lihat pada gambar 1.7.
5. Setelah itu tunggu sampai proses instalasi selesai lalu jika sudah tekan tombol finish. Lihat pada gambar 1.8.
6. Lalu buka command prompt anda dan tuliskan perintah berikut ini untuk mengecek apakah aplikasinya sudah terinstall. Lihat pada gambar 1.9.
7. Kemudian ketikkan perintah `pip install -U scikit-learn` seperti gambar berikut. Lihat pada gambar 1.10.
8. Lalu jika sudah ketikkan juga perintah `conda install scikit-learn`. Lihat pada gambar 1.11.
9. Hasil compile dari beberapa code yang mempunyai variable explorer. Lihat pada gambar 1.12.

1.4.2.2 Mencoba Loading an example Dataset

- `from sklearn import datasets`

(pada baris ini merupakan sebuah perintah untuk mengimport class datasets dari packaged sklearn).

- `iris = datasets.load_iris()`

(pada baris kedua ini dimana iris merupakan suatu estimator/parameter yang berfungsi untuk mengambil data pada item datasets.load_iris).

- `digits = datasets.load_digits()`

(pada baris ketiga ini dimana digits merupakan suatu estimator/parameter yang berfungsi untuk mengambil data pada item datasets.load_digits).

- `print(digits.data)`

(pada baris keempat ini merupakan perintah yang berfungsi untuk menampilkan estimator/parameter yang dipanggil pada item `digits.data` dan menampilkan outputannya) Lihat gambar 1.13.

- `digits.target`

(barisan ini untuk mengambil target pada estimator/parameter `digits` dan menampilkan outputannya) Lihat gambar 1.14.

- `digits.images[0]`

(barisan ini untuk mengambil `images[0]` pada estimator/parameter `digits` dan menampilkan outputannya) Lihat gambar 1.15.

1.4.2.3 Learning and Predicting

- `from sklearn import svm`

(pada baris ini merupakan sebuah perintah untuk mengimport class `svm` dari packaged `sklearn`).

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

(pada baris kedua ini `clf` sebagai estimator/parameter, `svm.SVC` sebagai class, `gamma` sebagai parameter untuk menetapkan nilai secara manual).

- `clf.fit(digits.data[:-1], digits.target[:-1])`

(pada baris ketiga ini `clf` sebagai estimator/parameter, `fit` sebagai metode, `digits.data` sebagai item, `[:-1]` sebagai syntax pythonnya dan menampilkan outputannya) Lihat gambar 1.16.

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

(pada baris terakhir ini `clf` sebagai estimator/parameter, `predict` sebagai metode lainnya, `digits.data` sebagai item dan menampilkan outputannya) Lihat gambar 1.17.

1.4.2.4 Model Persistence

- `from sklearn import svm`

(pada baris ini merupakan sebuah perintah untuk mengimport class `svm` dari packaged `sklearn`).

- `from sklearn import datasets`

(pada baris ini merupakan sebuah perintah untuk mengimport class datasets dari packaged sklearn).

- `clf = svm.SVC(gamma='scale')`

(pada baris ketiga ini clf sebagai estimator/parameter, svm.SVC sebagai class, gamma sebagai parameter untuk menetapkan nilai secara manual dengan nilai scale).

- `iris = datasets.load_iris()`

(pada baris keempat ini iris sebagai estimator/parameter, datasets.load_iris() sebagai item dari suatu nilai).

- `X, y = iris.data, iris.target`

(pada baris kelima ini X, y sebagai estimator/parameter, iris.data, iris.target sebagai item dari 2 nilai yang ada).

- `clf.fit(X, y)`

(pada baris keenam ini clf sebagai estimator/parameter dengan menggunakan metode fit untuk memanggil estimator X, y dengan outputannya) Lihat gambar 1.18.

- `import pickle`

(pickle merupakan sebuah class yang di import).

- `s = pickle.dumps(clf)`

(pada baris ini s sebagai estimator/parameter dengan pickle.dumps merupakan suatu nilai/item dari estimator/parameter clf)

- `clf2 = pickle.loads(s)`

(pada baris ini clf2 sebagai estimator/parameter, pickle.loads sebagai suatu item, dan s sebagai estimator/parameter yang dipanggil)

- `clf2.predict(X[0:1])`

(pada baris ini clf2.predict sebagai suatu item dengan menggunakan metode predict untuk menentukan suatu nilai dari (X[0:1])) Lihat gambar 1.19.

- `y[0]`

(pada estimator/parameter `y` berapapun angka yang diganti nilainya akan selalu konstan yaitu 0) Lihat gambar 1.20.

- `from joblib import dump, load`

(pada baris berikut ini merupakan sebuah perintah untuk mengimport class `dump`, `load` dari packaged `joblib`).

- `dump(clf, 'filename.joblib')`

(pada baris berikutnya `dump` di sini sebagai class yang didalamnya terdapat nilai dari suatu item `clf` dan data `joblib`).

- `clf = load('filename.joblib')`

(pada baris terakhir `clf` sebagai estimator/parameter dengan suatu nilai `load` berfungsi untuk mengulang data sebelumnya)

- dari ketiga baris akhir tersebut jika di jalankan aau dituliskan perintah seperti itu maka akan menampilkan tampilan eror terlihat pada gambar 1.21.

1.4.2.5 Conventions

1. Type Casting

- `from sklearn import svm`

(pada baris ini merupakan sebuah perintah untuk mengimport class `svm` dari packaged `sklearn`).

- `from sklearn import random_projection`

(pada baris ini merupakan sebuah perintah untuk mengimport class `random_projection` dari packaged `sklearn`).

- `rng = np.random.RandomState(0)`

(`rng` sebagai estimator/parameter dengan nilai suatu itemnya yaitu `np.random.RandomS`)

- `X = rng.rand(10, 2000)`

(`X` sebagai estimator/parameter dengan nilai item `rng.rand`).

- `X = np.array(X, dtype='float32')`

(`X` sebagai estimator/parameter dengan nilai item `np.array`).

- `X.dtype`
(`X.dtype` sebagai item pemanggil) Lihat gambar 1.22.
- `transformer = random_projection.GaussianRandomProjection()`
(`transformer` sebagai estimator/parameter dengan memanggil class `random_projection`).
- `X_new = transformer.fit_transform(X)`
(`X_new` di sini sebagai estimator/parameter dan menggunakan metode `fit`)
- `X_new.dtype`
(`X_new.dtype` sebagai item) Lihat gambar 1.23.
- `from sklearn import datasets`
(pada baris ini merupakan sebuah perintah untuk mengimport class `datasets` dari packaged `sklearn`).
- `from sklearn.svm import SVC`
(pada baris ini merupakan sebuah perintah untuk mengimport class `SVC` dari packaged `sklearn.svm`).
- `iris = datasets.load_iris()`
(`iris` sebagai estimator/parameter dengan item `datasets.load_iris()`).
- `clf = SVC(gamma='scale')`
(`clf` sebagai estimator/parameter dengan nilai class `SVC` pada parameter `gamma` sebagai set penilaian).
- `clf.fit(iris.data, iris.target)`
(estimator/parameter `clf` menggunakan metode `fit` dengan itemnya) Lihat gambar 1.24.
- `list(clf.predict(iris.data[:3]))`
(menambahkan item `list` dengan metode `predict`) Lihat gambar 1.25.
- `clf.fit(iris.data, iris.target_names[iris.target])`
(estimator/parameter `clf` menggunakan metode `fit` dengan itemnya) Lihat gambar 1.26.
- `list(clf.predict(iris.data[:3]))` (menambahkan item `list` dengan metode `predict`)
Lihat gambar 1.27.

2. Refitting and Updating Parameters

- `import numpy as np`
(pada baris ini merupakan sebuah perintah untuk mengimport class svm dari np).
- `from sklearn.svm import SVC`
(pada baris ini merupakan sebuah perintah untuk mengimport class SVC dari packaged sklearn.svm).
- `rng = np.random.RandomState(0)`
(rng sebagai estimator/parameter dengan nilai suatu itemnya yaitu np.random.RandomS
- `X = rng.rand(100, 10)`
(X sebagai estimator/parameter dengan nilai item rng.rand).
- `y = rng.binomial(1, 0.5, 100)`
(y sebagai estimator/parameter dengan nilai item rng.binomial).
- `X_test = rng.rand(5, 10)`
(X_test sebagai estimator/parameter dengan nilai item rng.rand).
- `clf = SVC()`
(clf sebagai estimator/parameter dan class SVC)
- `clf.set_params(kernel='linear').fit(X, y)`
(set_params sebagai item) Lihat gambar 1.28.
- `clf.predict(X_test)`
(menggunakan metode predict) Lihat gambar 1.29.
- `clf.set_params(kernel='rbf', gamma='scale').fit(X, y)`
Lihat gambar 1.30.
- `clf.predict(X_test)`
Lihat gambar 1.31.

3. Multiclass vs. Multilabel Fitting

- `from sklearn.svm import SVC`
(pada baris ini merupakan sebuah perintah untuk mengimport class SVC dari packaged sklearn.svm).
- `from sklearn.multiclass import OneVsRestClassifier`
(pada baris ini merupakan sebuah perintah untuk mengimport class OneVs-RestClassifier dari packaged sklearn.multiclass).

- `from sklearn.preprocessing import LabelBinarizer`
(pada baris ini merupakan sebuah perintah untuk mengimport class LabelBinarizer dari packaged sklearn.preprocessing).
- `X = [[1, 2], [2, 4], [4, 5], [3, 2], [3, 1]]`
- `y = [0, 0, 1, 1, 2]`
- `classif = OneVsRestClassifier(estimator=SVC(gamma='scale', random_state=0))`
- `classif.fit(X, y).predict(X)`

Lihat gambar 1.32.

- `y = LabelBinarizer().fit_transform(y)`
- `classif.fit(X, y).predict(X)`

Lihat gambar 1.33.

- `from sklearn.preprocessing import MultiLabelBinarizer`
- `y = [[0, 1], [0, 2], [1, 3], [0, 2, 3], [2, 4]]`
- `y = MultiLabelBinarizer().fit_transform(y)`
- `classif.fit(X, y).predict(X)`

Lihat gambar 1.34.

1.4.3 Penanganan error

1.4.3.1 ScreenShoot Error

```
>>> from joblib import dump, load
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
ModuleNotFoundError: No module named 'joblib'
>>> dump(clf, 'filename.joblib')
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
NameError: name 'dump' is not defined
>>> clf = load('filename.joblib')
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
NameError: name 'load' is not defined
```

Figure 1.1: Hasil Tampilan Error.

1.4.3.2 Tuliskan Kode Error dan Jenis Erornya

- `from joblib import dump, load`

(Kode baris pertama)

```
Traceback(most recent call last):
  File "<stdin>", line 1, in<module>
ModuleNotFoundError: No module named 'joblib'
```

(Errornya)

- `dump(clf, 'filename.joblib')`

(Kode baris kedua)

```
Traceback(most recent call last):
  File "<stdin>", line 1, in<module>
NameError: name 'dump' is not defined
```

(Errornya)

- `clf = load('filename.joblib')`

(Kode baris ketiga)

```
Traceback(most recent call last):
  File "<stdin>", line 1, in<module>
NameError: name 'load' is not defined
```

(Errornya)

1.4.3.3 Solusi Pemecahan Masalah Error

1. Pada masalah error sebelumnya itu dikarenakan kita belum mempunyai packaged joblib. Jadi solusinya yaitu dengan cara menginstall terlebih dahulu packaged joblibnya setelah itu baru perintah tersebut dapat dijalankan seperti pada gambar 1.2 dan 1.3

===== iiiiii HEAD

```
Command Prompt
Microsoft Windows [Version 10.0.17134.590]
(c) 2018 Microsoft Corporation. All rights reserved.

C:\Users\HUDA>pip install joblib
Requirement already satisfied: joblib in f:\anaconda\lib\site-packages (0.13.2)
distributed 1.21.8 requires msgpack, which is not installed.
You are using pip version 10.0.1, however version 19.0.3 is available.
You should consider upgrading via the 'python -m pip install --upgrade pip' command.

C:\Users\HUDA>
```

Figure 1.2: Hasil Tampilan Install joblib.

```
>>> from joblib import dump, load
>>> dump(clf, 'filename.joblib')
['filename.joblib']
>>> clf = load('filename.joblib')
>>>
```

Figure 1.3: Hasil Tampilan Uji coba perintah joblib.

1.5 Cokro Edi Prawiro/1164069

1.5.1 Praktek teori penunjang

1. Kecerdasan Buatan *Artificial Intelligence* adalah suatu cabang dalam bidang sains komputer yang mengkaji bagaimana untuk melengkapi sebuah komputer dengan kemampuan atau kepintaran seperti manusia. Komputer tersebut diharapkan dapat belajar sendiri dengan cara mengumpulkan data-data yang diterimanya, yang berguna sebagai parameter untuk memecahkan masalah. Jadi kecerdasan buatan merupakan kecerdasan yang di program dalam komputer untuk memecahkan masalah secara tepat dan cepat atau untuk memberikan kemungkinan keberhasilan dan kegagalan pada solusi dari suatu masalah.

Adapun kecerdasan buatan menurut para ahli adalah sebagai berikut :

- Kecerdasan Buatan merupakan Kawasan penelitian, aplikasi dan intruksi yang terkait dengan pemrograman komputer untuk melakukan sesuatu hal yang dalam pandangan manusia adalah cerdas (H. A. Simon[1997]).
- Kecerdasan buatan adalah bidang studi yang berhubungan dengan penangkapan, pemodelan, dan penyimpanan kecerdasan manusia dalam sebuah sistem teknologi sehingga sistem tersebut dapat memfasilitasi proses pengambilan keputusan yang biasanya dilakukan oleh manusia (Haag dan Keen[1996]).

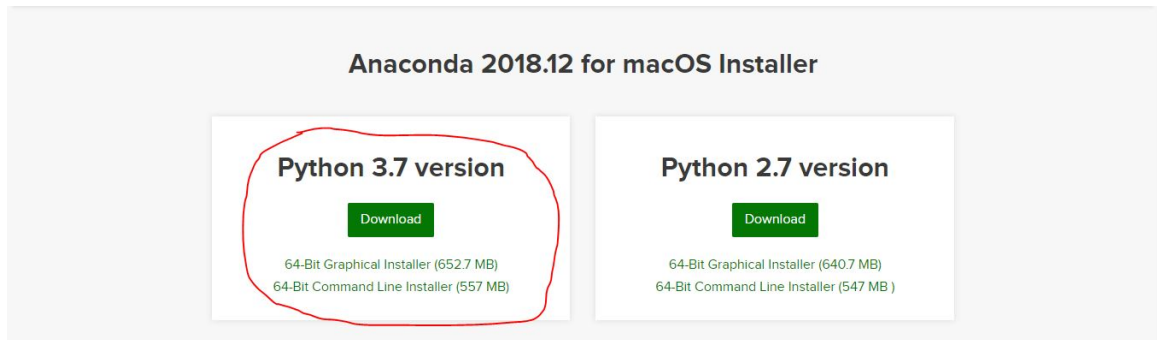


Figure 1.4: Download Anaconda.

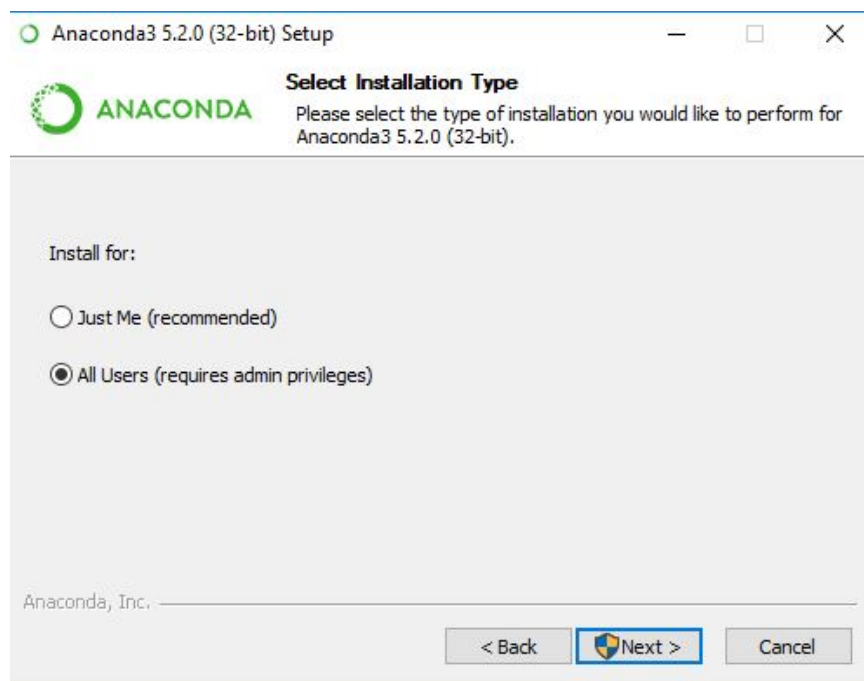


Figure 1.5: Langkah pertama instalasi anaconda.

Sejarah dan Perkembangan Kecerdasan Buatan.

ketika *Rene Descartes* mengemukakan gagasan yang menjadi cikal bakal kecerdasan buatan pada abad 17 mengemukakan bahwa hewan bukan apa-apa melainkan hanya mesin yang rumit yang dilanjutkan oleh *Belaise Pascal* yang telah menciptakan mesin penghitung digital mekanis pertama pada tahun 1642. Lalu pada abad 19 *Charles Babbage* dan *Ada lovelace* bekerja sama membuat mesin penghitung mekanis yang dapat di program.

Perkembangan kecerdasan buatan inipun terus berlanjut, *Bertrand Russell* dan *Alferd North Whithead* menerbitkan *mathematica*, yang merombak logika formal. Setelah itu dilanjutkan dengan penemuan oleh *Warren McCulloch* dan

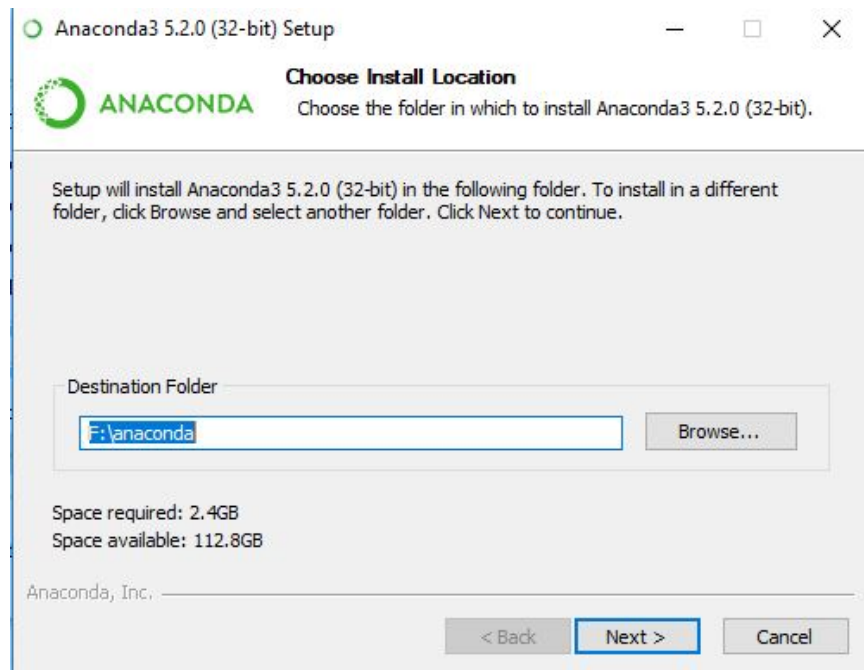


Figure 1.6: Langkah kedua instalasi anaconda.

Walter Pitts menerbitkan Kalkulus Logis Gagasan yang tetap ada dalam Aktivitas pada 1943 yang meletakkan pondasi awal berupa jaringan syaraf Kemudian pada tahun 1950-an adalah periode awal usaha aktif kecerdasan buatan. Program Kecerdasan buatan pertama yang bekerja di ciptakan pada tahun 1951 untuk menjalankan mesin Ferranti Mark I di University of Manchester (UK) yaitu sebuah program permainan naskah yang ditulis oleh *Christoper Strachey* dan program permainan catur yang ditulis oleh *Dietric Prinz*. Kemudian pada konferensi pertama tahun 1956 *John McCarthy* mengemukakan istilah kecerdasan buatan kemudian dia juga menemukan bahasa pemrograman lips. *Joseph Weizenbaum* menciptakan ELIZA, sebuah chatterbot yang menerapkan psikoterapi Rogerian.

Selama rentang waktu tahun 1960-an dan 1970-an, *Joel Moses* mendemonstrasikan kekuatan pertimbangan simbolis untuk mengintegrasikan masalah di dalam program Macsyma, yang merupakan program yang pertamakali sukses dalam bidang matematika. Kemudian pada tahun 1980-an industry kecerdasan buatan ini berkembang walu sudah di mulai pada tahun 1970-an Evolusi kecerdasan buatan berjalan dalam dua jalur yang berbeda yaitu meniru proses berpikir manusia untuk menyelesaikan masalah umum. Kedua mengkombinasikan pemikiran terbaik para ahli pada sepotong software yang dirancang

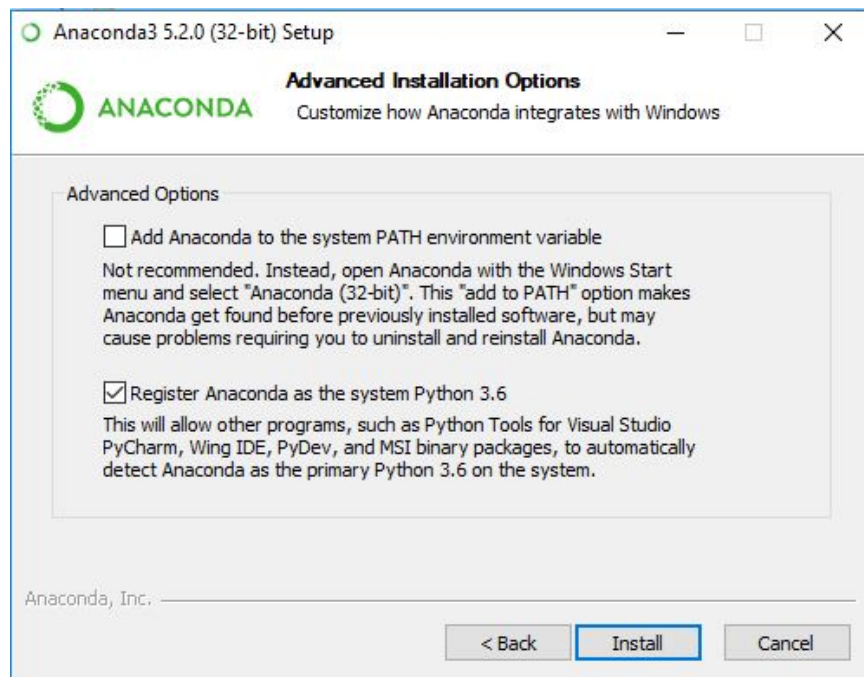


Figure 1.7: Langkah ketiga instalasi anaconda.

untuk memecahkan persolalan yang spesifik.

2. Supervised learning adalah sebuah pendekatan dengan syarat sudah terdapat data yang dilatih kemudian harus terdapat variable yang ditargetkan sehingga tujuan dari pendekatan ini adalah pengelompokan data terhadap data yang telah ada. Ciri khas dari Supervised learning yaitu terdapat label atau nama kelas pada data latih (supervisi) dan data baru di klasifikasikan berdasarkan data latih. Data latih dikelompokkan berdasarkan ukuran kemiripan pada suatu kelas. Berdasarkan keluaran dari fungsi, Supervised learning dibagi menjadi 2, regresi dan klasifikasi. Regresi terjadi jika output dari fungsi merupakan nilai yang kontinyu, sedangkan klasifikasi terjadi jika keluaran dari fungsi adalah nilai tertentu dari suatu atribut (tidak kontinyu). Tujuan dari Supervised learning adalah untuk memprediksi nilai dari fungsi untuk sebuah data masukanyang sah setelah melihat sejumlah data latih[3].

Adapun pengertian klasifikasi dan regresi adalah sebagai berikut :

- Klasifikasi merupakan pengelompokan berdasarkan parameter tertentu yang tidak konstan contoh pada mahluk hidup yaitu persamaan ciri cara hidup dan tempat hidup.

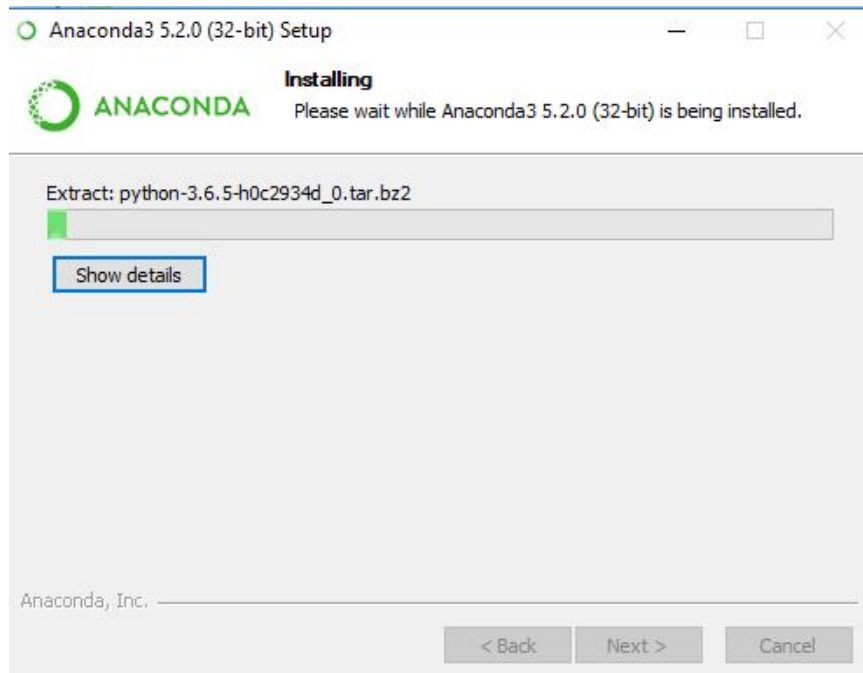


Figure 1.8: Langkah terakhir instalasi anaconda.

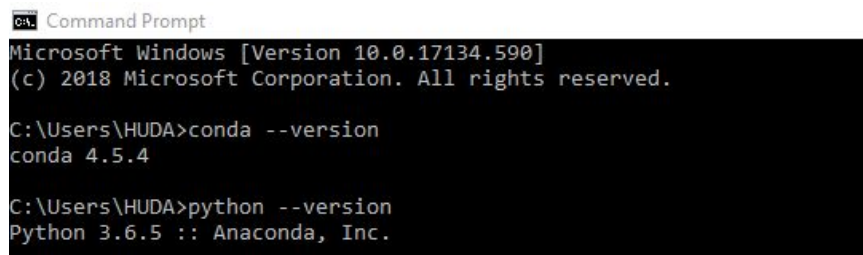


Figure 1.9: Langkah pertama instalasi scikit pada CMD.

- Regresi yaitu pengeluaran nilai output yang konstan jika dipicu dengan parameter tertentu biasanya regresi disini berbentuk regresi linier. Regresi linier yaitu metode statistika yang digunakan untuk membentuk model hubungan antara variabel terikat(dependen,respon,Y) dengan satu atau lebih variabel bebas(independent, prdikator, X). Apabila banyaknya variabel bebas hanya ada satu, disebut sebagai regresi linier sederhana, sedangkan apabila terdapat lebih dari satu variabel bebas, disebut sebagai regresi linier berganda [4].

unsupervised learning adalah pendekatan yang tidak memerlukan data latih atau data training untuk melakukan prediksi atau klasifikasi. Berdasarkan model secara matematisnya, algoritma ini tidak memiliki target variabel. Tu-

```
C:\Users\HUDA>pip install -U scikit-learn
Collecting scikit-learn
  Downloading https://files.pythonhosted.org/packages/ee/c8/c89ebdc0d7dbbafefcd222daabd257da3c28a967dd7c352d4272b2e1cef6/scikit_learn-0.20.2-cp36-cp36m-win32.whl (4.3MB)
    100% |#####| 4.3MB 425KB/s
Requirement not upgraded as not directly required: numpy>=1.8.2 in f:\anaconda\lib\site-packages (from scikit-learn) (1.14.3)
Requirement not upgraded as not directly required: scipy>=0.13.3 in f:\anaconda\lib\site-packages (from scikit-learn) (1.1.0)
Installing collected packages: scikit-learn
  Found existing installation: scikit-learn 0.19.1
    Uninstalling scikit-learn-0.19.1:
      Successfully uninstalled scikit-learn-0.19.1
Successfully installed scikit-learn-0.20.2
You are using pip version 18.0.1, however version 19.0.3 is available.
You should consider upgrading via the 'python -m pip install --upgrade pip' command.
```

Figure 1.10: Langkah kedua instalasi scikit pada CMD.

```
C:\Users\HUDA>conda install scikit-learn
Solving environment: done

## Package Plan ##

  environment location: F:\anaconda

  added / updated specs:
    - scikit-learn

The following packages will be downloaded:

  package | build | size
  -----|-----|-----
  conda-4.6.7 | py36_0 | 1.7 MB

The following packages will be UPDATED:

  conda: 4.5.4-py36_0 --> 4.6.7-py36_0

Proceed ([y]/n)? y

Downloading and Extracting Packages
conda-4.6.7 | 1.7 MB | ##### | 100%
Preparing transaction: done
Verifying transaction: done
Executing transaction: done
```

Figure 1.11: Langkah ketiga instalasi scikit pada CMD.

juan dari algoritma ini yaitu pengelompokan objek yang memiliki kesamaan atau hampir sama dalam satu cakupan wilayah tertentu. Kemudian pada unsupervised learning tidak terdapat label atau nama kelas pada data latih. Kemudian dataset merupakan objek yang menggambarkan data itu sendiri dan relasinya di memory. Struktur datanya mirip dengan struktur data di basis data. Jadi strikturnya terdiri atas baris kolom dan juga ada sejenis relasi data. Pada dataset terdiri bagian bagian yaitu training set dan Testing set. Adapun pengertian dari training set dan Testing set adalah sebagai berikut :

- Training set adalah bagian dari dataset itu sendiri yang dilatih untuk membuat prediksi atau algoritma mesin learning lainnya sesuai keinginan atau tujuan data itu dibuat.
- Testing set adalah bagian dari dataset yang di tes atau diujicoba untuk melihat keakuratannya dengan katalain melihat peformanya.

```

C:\Users\HUDA>python
Python 3.6.5 [Anaconda, Inc.] (default, Mar 29 2018, 13:23:52) [MSC v.1900 32 bit (Intel)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>> from sklearn.metrics import confusion_matrix
>>> y_true = [2, 0, 2, 2, 0, 1]
>>> y_pred = [0, 0, 2, 2, 0, 2]
>>> confusion_matrix(y_true, y_pred)
array([[2, 0, 0],
       [0, 0, 1],
       [1, 0, 2]], dtype=int64)
>>> y_true = ["cat", "ant", "cat", "cat", "ant", "bird"]
>>> y_pred = ["ant", "ant", "cat", "cat", "ant", "cat"]
>>> confusion_matrix(y_true, y_pred, labels=["ant", "bird", "cat"])
array([[2, 0, 0],
       [0, 0, 1],
       [1, 0, 2]], dtype=int64)
>>> tn, fp, fn, tp = confusion_matrix([0, 1, 0, 1], [1, 1, 1, 0]).ravel()
>>> (tn, fp, fn, tp)
(0, 2, 1, 1)
>>>

```

Figure 1.12: Langkah compile code pada python anaconda.

```

[[ 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.]]

```

Figure 1.13: Hasil Tampilan 1.

1.5.2 Instalikasi

Pada proses instalikasi ini langkah pertama yaitu mengakses website scikit dengan mengakses link berikut <https://scikit-learn.org/stable/tutorial/basic/tutorial.html> maka hasilnya dapat dilihat pada gambar 1.35 kemudian setelah itu klik button installation maka akan muncul tampilan yang dapat dilihat pada gambar 1.36.

1. cara instalikasi Instalasi library scikit dari anaconda langkah pertama instal terlebih dahulu anacondanya dikarenakan anaconda sudah include dengan python maka codingan python dapat digunakan di anaconda dan ketika diperikas versinya maka akan muncul tampilan seperti Gambar 1.51

kemudian pada cmd administrator install library scikit dengan cara memasukan codingan `pip install -U scikit-learn` maka hasilnya seperti seperti pada gambar 1.52 berikut.

setelah itu masukan kembali perintah berikut di cmd `conda install scikit-learn` jika hasilnya seperti pada gambar 1.53 maka librari scikit telah terinstal dan siap untuk di gunakan.

```
array([0, 1, 2, ..., 8, 9, 8])
```

Figure 1.14: Hasil Tampilan 2.

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

Figure 1.15: Hasil Tampilan 3.

kemudian untuk mencobanya tuliskan perintah python pada cmd lalu masukan codingan `print("Hello Anaconda!")` maka hasilnya terlihat seperti gambar 1.54 codingan `print("Hello Anaconda!")` yaitu berfungsi mencetak nilai yang ada di dalam kurung dan di antara kutip.

2. cara mencoba dataset yaitu dengan cara memasukan perintah berikut pada cmd seperti pada gambar 1.55

- pada codingan `from sklearn import datasets` menjelaskan inport librari dataset dari librari sikic pada gambar 1.55
- pada codingan `iris = datasets.load_iris` iris berarti parameter atau acuan bernama iris kemudian di load ke dalam dataset sebagai perbandingan kalau dalam diagram iris bisa disebut X nya pada gambar 1.55
- pada codingan `digits = datasets.load_digits` digits berarti parameter hampir mirip seperti iris tadi namun digits merupakan kebalikannya kalau di dalam diagram dia bernilai Y pada gambar 1.55
- kemudian pada codingan `print(digits.data)` yaitu mencetak data digits yang di bandingkan dengan data iris pada gambar 1.55
- pada codingan `print(iris.data)` yaitu mencetak data iris yang dibandingkan dengan data digits pada gambar 1.55

3. Mencoba Learning and Predicting

Pada kasus ini dataset digit digunakan untuk memprediksi yang mana telah di berikan gambar untuk mewakili sampel masing-masing dari 10 kelas yang dimulai dari digit nol hingga sembilan yang digunakan untuk memprediksi sampel

```
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)
```

Figure 1.16: Hasil Tampilan 4.

```
array([8])
```

Figure 1.17: Hasil Tampilan 5.

yang tidak terlihat untuk lebih jelasnya dapat di praktikan codingan berikut ini.

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

pada baris ini dapat diartikan bahwa librari sklearn mengimport package dataset

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

pada baris ini dimasukan parameter iris yang di sandingkan dengan dataset load sehingga iris berisi nilai dataset

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

pada baris ini dimasukan parameter digits yang di sandingkan dengan dataset load sehingga digits berisi nilai dataset

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

pada baris ini librari sklearn mengimport package svm

```
>>> clf = svm.SVC(gamma=0.0001, C=100.)
```

pada codingan diatas dibuat variabel clf yang di isi dengan nilai svm dengan nilai gama 0.0001 dan 100.

```
>>> clf.fit(digits.data[:-1], digits.target[:-1])
```

pada codingan clf di implementasikan dengan perintah fit


```
SVC(C=1.0, cache_size=200, class_weight=None, coef0=0.0,
    decision_function_shape='ovr', degree=3, gamma='scale', kernel='rbf',
    max_iter=-1, probability=False, random_state=None, shrinking=True,
    tol=0.001, verbose=False)
```

Figure 1.18: Hasil Tampilan 6.

```
array([0])
```

Figure 1.19: Hasil Tampilan 7.

```
SVC(C=100.0, cache_size=200, class_weight=None, coef0=0.0,
    decision_function_shape='ovr', degree=3, gamma=0.0001, kernel='rbf',
    max_iter=-1, probability=False, random_state=None, shrinking=True,
    tol=0.001, verbose=False)
```

yang hasilnya seperti codingan diatas yang menjabarkan isi dari SVC itu sendiri.

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

kemudain pada codingan diatas digunakan perintah predic yang merupakan perintah untuk mengimplementasikan method digits.

4. Mencoba model pertistence

model pertistence yaitu model yang digunakan untuk mengolah data sehingga data tersebut konstan atau konsisten terhadap parameter tertentu contoh pada codingan di bagawah nilai y akan konstan di nol walaupun telah di isi nilai lebih dari nol.

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

pada baris ini librari sklearn mengimport package svm.

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

pada baris ini librari sklearn mengimport package datasets.

```
>>> y[0]
0
```

Figure 1.20: Hasil Tampilan 8.

```
>>> from joblib import dump, load
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
ModuleNotFoundError: No module named 'joblib'
>>> dump(clf, 'filename.joblib')
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
NameError: name 'dump' is not defined
>>> clf = load('filename.joblib')
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
NameError: name 'load' is not defined
```

Figure 1.21: Hasil Tampilan 9.

```
>>> clf = svm.SVC(gamma='scale')
```

pada codingan diatas dibuat variabel clf yang di isi dengan nilai svm dengan gamma sama dengan scale.

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

pada baris ini dimasukan parameter iris yang di sandingkan dengan dataset load sehingga iris berisi nilai dataset.

```
>>> X, y = iris.data, iris.target
```

pada codingan diatas X berisi nilai iris.data dan y berisi nilai iris.target.

```
>>> clf.fit(X, y)
```

method clf di implementasikan dengan perintah fit dengan X, y sebagai nilai untuk implementasinya.

```
>>> X.dtype
dtype('float32')
```

Figure 1.22: Hasil Tampilan 10.

```
>>> X_new.dtype
dtype('float64')
```

Figure 1.23: Hasil Tampilan 11.

```
SVC(C=1.0, cache_size=200, class_weight=None, coef0=0.0,
    decision_function_shape='ovr', degree=3, gamma='scale', kernel='rbf',
    max_iter=-1, probability=False, random_state=None, shrinking=True,
    tol=0.001, verbose=False)
```

maka hasilnya penjabaran dari SVC seperti codingan diatas.

```
>>> import pickle
```

mengimport library atau package pickle.

```
>>> s = pickle.dumps(clf)
```

kemudian di buat variabel s yang di load oleh package pickle dengan di isi nilai clf.

```
>>> clf2 = pickle.loads(s)
```

setelah itu pada codingan diatas dibuat lagi variabel clf2 kemudian di load pickle.

```
>>> clf2.predict(X[0:1])
```

kemudian variabel clf2 di implementasikan dengan parameter X dengan nilai 0 berbanding 1 maka hasilnya nilainya array bernilai nol dan y bernilai nol.

```
>>> clf.fit(iris.data, iris.target)
SVC(C=1.0, cache_size=200, class_weight=None, coef0=0.0,
    decision_function_shape='ovr', degree=3, gamma='scale', kernel='rbf',
    max_iter=-1, probability=False, random_state=None, shrinking=True,
    tol=0.001, verbose=False)
```

Figure 1.24: Hasil Tampilan 12.

```
>>> list(clf.predict(iris.data[:3]))
[0, 0, 0]
```

Figure 1.25: Hasil Tampilan 13.

```
array([0])
```

```
>>> y[0]
```

```
0
```

5. mencoba conventions conventions merupakan aturan aturan dasar atau kesepakatan kesepakatan dalam pemrograman sikit python dan anaconda berikut merupakan jenis-jenis codingan conventions :

- Type casting yaitu tipe pelemparan parameter atau variabel kedalam variabel baru.

```
>>> import numpy as np
```

codingan diatas yaitu import librari numpy yang di inisialisasi menjadi np

```
>>> from sklearn import random_projection
```

import librari random_projection

```
>>> rng = np.random.RandomState(0)
```

membuat variabel baru dengan nama rng dengan nilai random

```
>>> X = rng.rand(10, 2000)
```

memasukan nilai rng kedalam variabel X dengan rad nilai 10 sampai 2000

```
>>> X = np.array(X, dtype='float32')
```

menambahkan nilai np berupa array yaitu X dan float 32

```
>>> X.dtype
```

```
dtype('float32')
```

X.dtype di running menghasilkan nilai dtype float

```
>>> clf.fit(iris.data, iris.target_names[iris.target])
SVC(C=1.0, cache_size=200, class_weight=None, coef0=0.0,
    decision_function_shape='ovr', degree=3, gamma='scale', kernel='rbf',
    max_iter=-1, probability=False, random_state=None, shrinking=True,
    tol=0.001, verbose=False)
```

Figure 1.26: Hasil Tampilan 14.

```
>>> list(clf.predict(iris.data[:3]))
['setosa', 'setosa', 'setosa']
```

Figure 1.27: Hasil Tampilan 15.

```
>>> transformer = random_projection.GaussianRandomProjection()
```

membuat variabel transformer dengan nilai random

```
>>> X_new = transformer.fit_transform(X)
```

membuat variabel X_new dan di isi nilai transformer kemudian di implementasikan

```
>>> X_new.dtype
```

merunning variabel X_new

```
dtype('float64')
```

hasil running X_new

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

mengimport library dataset

```
>>> from sklearn.svm import SVC
```

mengimport library SVC

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

membuat variabel iris dengan nilai load dataset

```
>>> clf = SVC(gamma='scale')
```

membuat variabel clf bernilai SVC dengan gamma menggunakan nilai sekala

```
>>> clf.fit(iris.data, iris.target)
```

```
>>> clf.set_params(kernel='linear').fit(X, y)
SVC(C=1.0, cache_size=200, class_weight=None, coef0=0.0,
    decision_function_shape='ovr', degree=3, gamma='auto_deprecated',
    kernel='linear', max_iter=-1, probability=False, random_state=None,
    shrinking=True, tol=0.001, verbose=False)
```

Figure 1.28: Hasil Tampilan 16.

```
>>> clf.predict(X_test)
array([1, 0, 1, 1, 0])
```

Figure 1.29: Hasil Tampilan 17.

merunning variabel atau method clf dengan isian nilai iris data dan iris target

```
SVC(C=1.0, cache_size=200, class_weight=None, coef0=0.0,
    decision_function_shape='ovr', degree=3, gamma='scale', kernel='rbf',
    max_iter=-1, probability=False, random_state=None, shrinking=True,
    tol=0.001, verbose=False)
```

detail hasil runing clf

```
>>> list(clf.predict(iris.data[:3]))
[0, 0, 0]
```

memunculkan detail atau lis sebanyak tiga nilai

```
>>> clf.fit(iris.data, iris.target_names[iris.target])
```

merunning kembali clf dengan nilai iris data, iris target name

```
SVC(C=1.0, cache_size=200, class_weight=None, coef0=0.0,
    decision_function_shape='ovr', degree=3, gamma='scale', kernel='rbf',
    max_iter=-1, probability=False, random_state=None, shrinking=True,
    tol=0.001, verbose=False)
```

hasil dari running clf

```
>>> list(clf.predict(iris.data[:3]))
['setosa', 'setosa', 'setosa']
```

memunculkan tiga nilai yang telah dilempar dari SVC

```
>>> clf.set_params(kernel='rbf', gamma='scale').fit(X, y)
SVC(C=1.0, cache_size=200, class_weight=None, coef0=0.0,
    decision_function_shape='ovr', degree=3, gamma='scale', kernel='rbf',
    max_iter=-1, probability=False, random_state=None, shrinking=True,
    tol=0.001, verbose=False)
```

Figure 1.30: Hasil Tampilan 18.

```
>>> clf.predict(X_test)
array([1, 0, 1, 1, 0])
```

Figure 1.31: Hasil Tampilan 19.

- Refitting and updating parameters atau pengisian ulang atau memperbarui parameter merupakan cara untuk merubah nilai dari sebuah parameter contoh nilai x adalah 10 jika di perbaharui bisa menjadi 15 begitu juga dalam codiangan berikut hal ini dapat dilakukan untuk lebih jelasnya dapat dilihat codingan dibawah ini :

```
>>> import numpy as np
```

codingan diatas yaitu import librari numpy yang di inisialisasi menjadi np

```
>>> from sklearn.svm import SVC
```

mengimport library SVC

```
>>> rng = np.random.RandomState(0).
```

membuat variabel baru dengan nama rng dengan nilai random.

```
>>> X = rng.rand(100, 10)
```

parameter X dengan nilai dari variabel rng dan rad dari 100 sampai 10.

```
>>> y = rng.binomial(1, 0.5, 100)
```

parameter y dengan nilai rng binominal dari 1 0,5 sampai 100.

```
>>> X_test = rng.rand(5, 10)
```

parameter X_test dengan nilai rng dan rad dari 5 ke 10

```
>>> clf = SVC()
```

parameter clf bernilai SVC

```
>>> clf.set_params(kernel='linear').fit(X, y)
```

```
>>> clf.fit(X, y).predict(X)
array([0, 0, 1, 1, 2])
```

Figure 1.32: Hasil Tampilan 20.

```
>>> clf.fit(X, y).predict(X)
array([[1, 0, 0],
       [1, 0, 0],
       [0, 1, 0],
       [0, 0, 0],
       [0, 0, 0]])
```

Figure 1.33: Hasil Tampilan 21.

parameter clf di set dengan mengkompilasi atau mengekstrak nilai X dan y dengan kernel linear.

```
SVC(C=1.0, cache_size=200, class_weight=None, coef0=0.0,
    decision_function_shape='ovr', degree=3, gamma='auto_deprecated',
    kernel='linear', max_iter=-1, probability=False, random_state=None,
    shrinking=True, tol=0.001, verbose=False)
```

penjabaran nilai SVC hasil running CLF

```
>>> clf.predict(X_test)
array([1, 0, 1, 1, 0])
```

merunning clf dengan nilai X_test

```
>>> clf.set_params(kernel='rbf', gamma='scale').fit(X, y)
```

parameter clf di set dengan kernel rbf dan gamma skala dan mengkompilasi nilai X dan y.

```
SVC(C=1.0, cache_size=200, class_weight=None, coef0=0.0,
    decision_function_shape='ovr', degree=3, gamma='scale', kernel='rbf',
    max_iter=-1, probability=False, random_state=None, shrinking=True,
    tol=0.001, verbose=False)
```

penjabaran nilai SVC hasil running CLF


```
>>> clf.fit(X, y).predict(X)
array([[1, 1, 0, 0, 0],
       [1, 0, 1, 0, 0],
       [0, 1, 0, 1, 0],
       [1, 0, 1, 0, 0],
       [1, 0, 1, 0, 0]])
```

Figure 1.34: Hasil Tampilan 22.

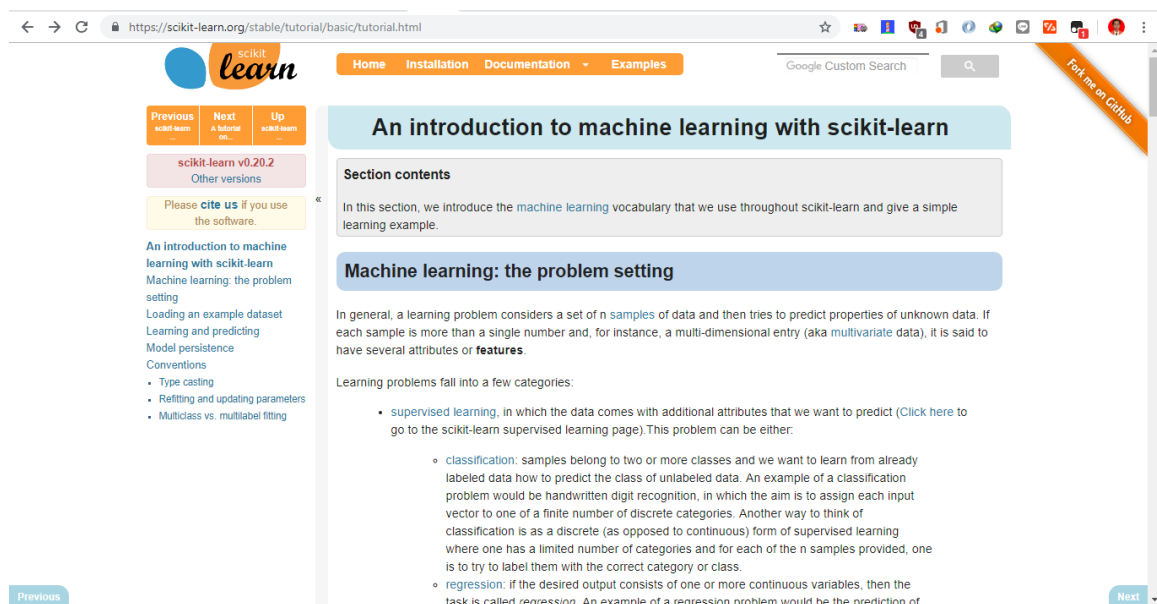


Figure 1.35: Tampilan website Scikit 1.

```
>>> clf.predict(X_test)
array([1, 0, 1, 1, 0])
```

Hasil dari running clf.

- Multiclass vs. multilabel fitting perbandingan antara banyak kelas dan pelabelan yang tepat berikut merupakan codingannya.

```
>>> from sklearn.svm import SVC
```

mengimport library SVC

```
>>> from sklearn.multiclass import OneVsRestClassifier
```

memasukan librari OneVsRestClassifier dengan kondisi multi class.

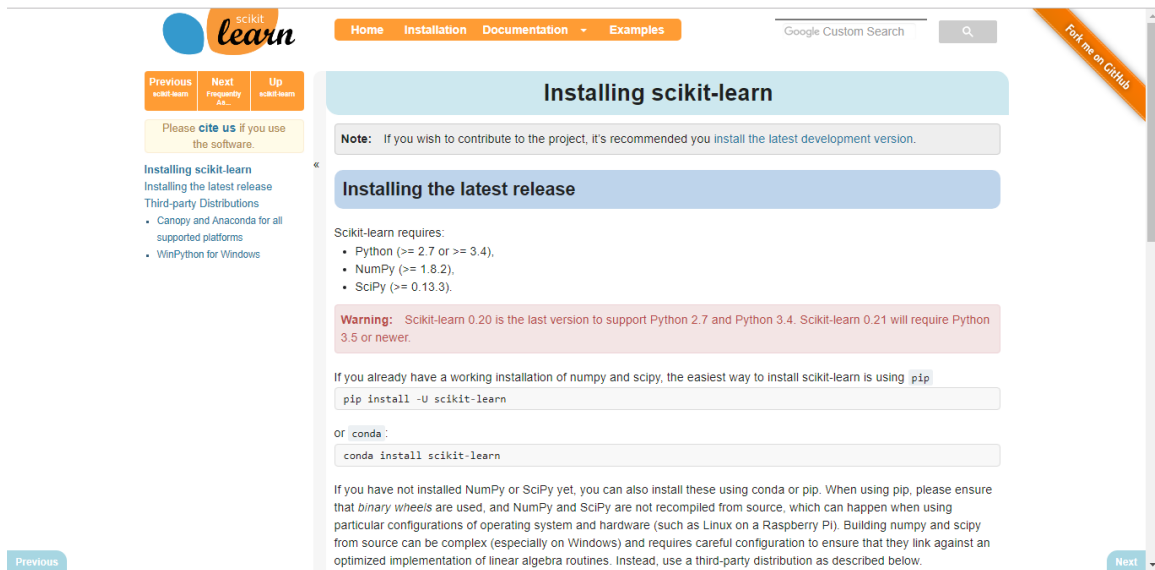


Figure 1.36: Tampilan website Scikit 2.

```
>>> from sklearn.preprocessing import LabelBinarizer
```

memasukan librari LabelBinarizer

```
>>> X = [[1, 2], [2, 4], [4, 5], [3, 2], [3, 1]]
```

pemberian nilai pada parameter X

```
>>> y = [0, 0, 1, 1, 2]
```

pemberian nilai pada parameter y

```
>>> classif = OneVsRestClassifier(estimator=SVC(gamma='scale',
... random_state=0))
```

opsi untuk class dengan ketentuan estimator SVC gamma berbentuk skala dan random (acak).

```
>>> classif.fit(X, y).predict(X)
```

```
array([0, 0, 1, 1, 2])
```

hasil array dari running classif

```
>>> y = LabelBinarizer().fit_transform(y)
```

memberikan nilai pada parameter y

```
>>> classif.fit(X, y).predict(X)
```

```
array([[1, 0, 0],
```

```
[1, 0, 0],  
[0, 1, 0],  
[0, 0, 0],  
[0, 0, 0]])
```

hasil running classif

```
>>> from sklearn.preprocessing import MultiLabelBinarizer  
  
import librari multi label  
  
>>> y = [[0, 1], [0, 2], [1, 3], [0, 2, 3], [2, 4]]  
  
memberikan nilai pada parameter y  
  
>>> y = MultiLabelBinarizer().fit_transform(y)  
  
membuat parameter y menjadi multi label.  
  
>>> classif.fit(X, y).predict(X)
```

1.6 Fathi Rabbani / 1164074

1.6.1 Teori

1. Sejarah dan Perkembangan Kecerdasan Buatan

Sejarah dari sebuah Artificial Intelligence atau dalam Bahasa indonesianya diterjemahkan sebagai Kecerdasan Buatan adalah sebuah usaha untuk dapat memodelkan sebuah mesin agar dapat berfikir dan menirukan tingkah laku dan cara berfikir manusia, ada beberapa jenis dari kecerdasan buatan, yaitu :

- Symbol Manipulating AI
- Nueral AI
- Neural Network

Peneliti yang selalu disebutkan sebagai Bapak AI adalah Jhon McCharty merupakan seorang dosen yang mengenalkan Kecerdasan Buatan kepada 2 lembaga penelitian hebat, yaitu Stanford Artificial Intelligence Laboratory dan MIT Artificial Intelligence Laboratory.

Sedangkan perkembangan kecerdasan buatan saat ini sudah mencapai tahap dimana manusia mulai membuat sebuah robot yang dapat menirukan hampir

90 persen dari keseharian mereka, mulai dari bidang kesehatan, koki, pabrik, kantor, hingga sebuah robot yang bertugas sebagai seorang pelayan di sebuah restoran. Dan dubai sebagai pengguna mobil tanpa pengemudi yang menerapkan AI dengan menggunakan data wilayah serta jarak kendaraan dengan pingir jalan.

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

- Supervised learning merupakan sebuah pendekatan AI dengan latihan yang sudah dilakukan dengan sebuah data yang lengkap, dan memiliki variable yang dapat digunakan sebagai target sehingga dapat menunjukan data agar menjadi kelompok dari sebuah data menjadi kelompok data yang baru.
- Unsupervised learning merupakan sebuah pendekatan AI tanpa menggunakan data yang lengkap dan ter-variable sehingga harus dilakukan pengelompokkan agar data tersebut dapat digunakan.
- Klasifikasi merupakan sebuah pengelompokkan suatu objek ke dalam kategori tertentu.
- Regresi merupakan pendekatan model matematika untuk mendeskripsikan hubungan dari beberapa variabel independen dengan variable dependen.
- Data Set, merupakan sebuah objek yang merepresentasikan data dan hubungannya di memory.
- Training Set, subset untuk melatih model.
- Testing Set, subset untuk menguji model yang sudah dilatih.

1.6.2 Praktikum

3. Instalasi Library Scikit dari Anaconda

Pertama Download terlebih dahulu anaconda-nya di <https://www.anaconda.com/distrib> pilih Operating Sistem yang kalian gunakan. lalu setelah download Install dengan proses berikut :

- Proses Instalasi Anaconda pada gambar 2.73 hingga proses 2.80.
- Proses Instalasi Scikit-Learn dengan menggunakan Conda pada gambar 2.81 hingga gambar 2.83.

- contoh dari Variable Explorer yang digunakan ada pada gambar 1.48.

4. Load Example Dataset dan Menjelaskan kegunaan barisan Code

berikut ini adalah contoh dataset yang digunakan untuk melakukan compile ada pada gambar 1.49 dan hasilnya ada pada gambar 1.50.

- dari code yang dicoba diketahui bahwa data set yang digunakan adalah data yang diambil dari SKLEARN yang ada pada gambar 1.49.

- Learning and Predicting

Dalam scikit-learn estimator untuk klasifikasi adalah sebuah objek data python yang memiliki implementasi method `fit(x, y)` dan `predict(T)`. Sebuah estimator dari class `sklearn.svm.SVC`, yang mana implementasi dari support vector classification. Estimator dari konstruktor mengambil argument dari model parameter.

```
from sklearn import svm
clf = svm.SVC(gamma=0.001, C=100.)
clf.fit(digits.data[:-1], digits.target[:-1])
clf.predict(digits.data[:-1])
```

mengambil nilai data svm ada pada class sklearn, lalu set nilai data dengan `clf = svm.SVC(gamma=0.001, C=100.)`. variable `clf` digunakan dengan method `fit` yang di set nilainya `[:-1]` yang memproduksi array baru dari data `digits.data`, dengan menggunakan `digits.data` sebagai acuan, sekarang tinggal melakukan prediksinya.

- Model Persistence

```
from sklearn import svm
from sklearn import datasets
clf = svm.SVC(gamma=0.001)
iris = datasets.load_iris()
X, y = iris.data, iris.target
clf.fit(X, y)
```

mengambil nilai data svm ada pada class sklearn dan mengambil nilai data `datasets` ada pada class sklearn, lalu buat variable `clf` dengan nilai data dan buat variable yang berisi nilai `load_iris`. variable `X` dan `y` yang

berisi nilai iris data dan iris target, lalu memanggil nilai variable X dan y dengan data variable clf dan method fit.

```
import pickle
s = pickle.dumps(clf)
clf2 = pickle.loads(s)
clf2.predict(X[0:1])
y[0]
```

mengambil nilai data pickle dan membuat variable s dengan data nilai pickle.dumps yang berisi data variable clf, membuat variable clf2 dengan data pickle.loads yang menggunakan variable s. menggunakan data variable clf2 dengan method predict dengan data variable X dan data variable y.

- Conventions

```
– import numpy as np
  from sklearn import random_projection
```

```
rng = np.random.RandomState(0)
X = rng.rand(10, 2000)
X = np.array(X, dtype='float32')
X.dtype
```

```
transformer = random_projection.GaussianRandomProjection()
X_new = transformer.fit_transform(X)
X_new.dtype
```

mengambil data numpy dan dialiaskan sebagai np. dari data sklearn mengambil data random_projection. lalu buat variable rng yang berisi nilai data np dengan random yang berawal 0. lalu variable X dengan data rng yang memiliki type rand berisi data 10 dan 2000. lalu dibuatkan arraynya dengan format `X = np.array(X, dtype='float32')`. dan nilai variable transform yang digunakan untuk menampilkan hasil random, dengan format Gaussian random projection. format penampilannya `X_new = transformer.fit_transform(X)` dan `X_new.dtype`

```
– from sklearn import datasets
  from sklearn.svm import SVC
```

```

iris = datasets.load_iris()
clf = SVC(gamma=0.001)
clf.fit(iris.data, iris.target)

list(clf.predict(iris.data[:3]))

clf.fit(iris.data, iris.target_names[iris.target])

list(clf.predict(iris.data[:3]))

```

dari data sklearn mengambil datasets dan SVC dari svm, selanjutnya membuat format data variable dengan nilai load_iris, clf dengan format `SVCgamma = 0.001`, lalu di jalankan dengan moethod `clf.fit` dengan `iris.data` dan `iris.target` sebagai nilainya.

buatkan tampilan datanya dengan list menampilkan data clf dengan `predict` pada `iris.data`, dan dilakukan selanjutnya dengan `clf.fit` dengan nilai data `iris.data` dan `iris.target_names[iris.target]`. tampilkan lagi dalam bentuk list data tersebut.

```

- import numpy as np
  from sklearn.svm import SVC

rng = np.random.RandomState(0)
X = rng.rand(100, 10)
y = rng.binomial(1, 0.5, 100)
X_test = rng.rand(5, 10)

clf = SVC()
clf.set_params(kernel='linear').fit(X, y)

clf.predict(X_test)

clf.set_params(kernel='rbf', gamma=0.001).fit(X, y)

clf.predict(X_test)

```

mengambil data numpy dan dialiaskan sebagai np dan dari sklearn.svm mengambil data SVC, lalu buat variable rng yang berisi nilai data np

dengan random yang berawal 0, lalu variable X dengan data rng yang memiliki type rand berisi data(100, 10) dan y yang memiliki data (1, 0.5, 100) dengan type data binomial lalu buat X_test untuk variable test.

```

- from sklearn.svm import SVC
  from sklearn.multiclass import OneVsRestClassifier
  from sklearn.preprocessing import LabelBinarizer

X = [[1, 2], [2, 4], [4, 5], [3, 2], [3, 1]]
y = [0, 0, 1, 1, 2]

classif = OneVsRestClassifier(estimator=SVC(gamma=1, random_state=0))

classif.fit(X, y).predict(X)

```

```

y = LabelBinarizer().fit_transform(y)
classif.fit(X, y).predict(X)

```

mengambil data sklearn SVC dari svm, OneVsRestClassifier dari multiclass, dan LabelBinarizer dari preprocessing. lalu buat variable X dan y yang berisi data nilai dan buat data variable classif untuk digunakan sebagai parameter bagi OneVsRestClassifier yang menghitung data estimator SVC. lalu jalankan dengan method fit dan predict, lalu variable y digunakan sebagai parameter yang menjalankan method fit_transform yang berisi data LabelBinarizer.

```

- from sklearn.preprocessing import MultiLabelBinarizer

y = [[0, 1], [0, 2], [1, 3], [0, 2, 3], [2, 4]]
y = MultiLabelBinarizer().fit_transform(y)
classif.fit(X, y).predict(X)

```

mengambil data dari sklearn datanya adalah MultiLabelBinarizer dari preprocessing, buat variable y yang berisikan nilai integer untuk di proses agar menghasilkan data seperti berikut

```

>>>>>> c6dbbab89a86daa2bf691a75d9a15c4b56fb15d7
array([[1, 1, 0, 0, 0],
       [1, 0, 1, 0, 0],
       [0, 1, 0, 1, 0],
       [1, 0, 1, 0, 0],

```



```
[1, 0, 1, 0, 0]])
```

iiiiii HEAD hasil running classif setelah nili parameter y telah di ganti.

1.6.3 Penanganan Error

(a) Screenshot Error untuk lebih jelasnya Screenshot codingan dapat dilihat pada gambar 1.56 dan 1.57

(b) Kode error pada screenshot untuk kode error pada gambar 1.56 yaitu pada source code berikut

```
clf.fit(digits.data[:-1], digits.target[:-1])
```

pada codingan tersebut menjadi error dikarenakan variabel atau method digits belum di definisikan. sedangkan untuk kode error pada gambar 1.57 yaitu pada source code

```
from joblib import dump, load
```

pada codingantersebut terjadi error dikarenakan module joblib belum di install atau modul tersebut tidak ada di library python.

(c) Solusi Pemecahan Masalah Error

- untuk memperbaiki error pada gambar 1.56 tinggal mendefinisikan variabel atau method digits, untuk lebih lengkapnya dapat dilihat pada gambar 1.58 dengan cara tersebut maka masalah error dapat diselesaikan.
- sedangkan untuk error joblib bisa dilakukan dengan cara masuk ke cmd administrator kemudian isikan perintah pip install joblib kemudiantekan enter sehingga hasilnya terlihat seperti gambar 1.59 setelah itu coba masuk kembali ke python di cmd dan ketikan perintah from joblib import dump, load maka hasilnya seperti gambar 1.60.

=====

5. Error Handling

(a) Screenshot

(b) Error Code and Error Type

- Module Not Found

module yang dicari tidak ditemukan, karena file yang dicari tidak ada atau belum di instal.

- Type Data Error

type data yang seharusnya diisi oleh data number namun diisi oleh data str/character sedangkan nilai yang bisa dibaca adalah number.

(c) Solution

- For Module Not Found

lakukan instalasi dengan memasukan code berikut untuk download dan instalasi module JOBLIB

```
conda install -c anaconda joblib
```

- For Data Type Error

ganti isi data menjadi data number, contoh :

```
clf = SVC(gamma='scale')
```

ganti menjadi

```
clf = SVC(gamma=0.5)
```

~~~~~ c6dbbab89a86daa2bf691a75d9a15c4b56fb15d7 ~~~~~ b1d7802739e393f94f14f567f8e8c88ea

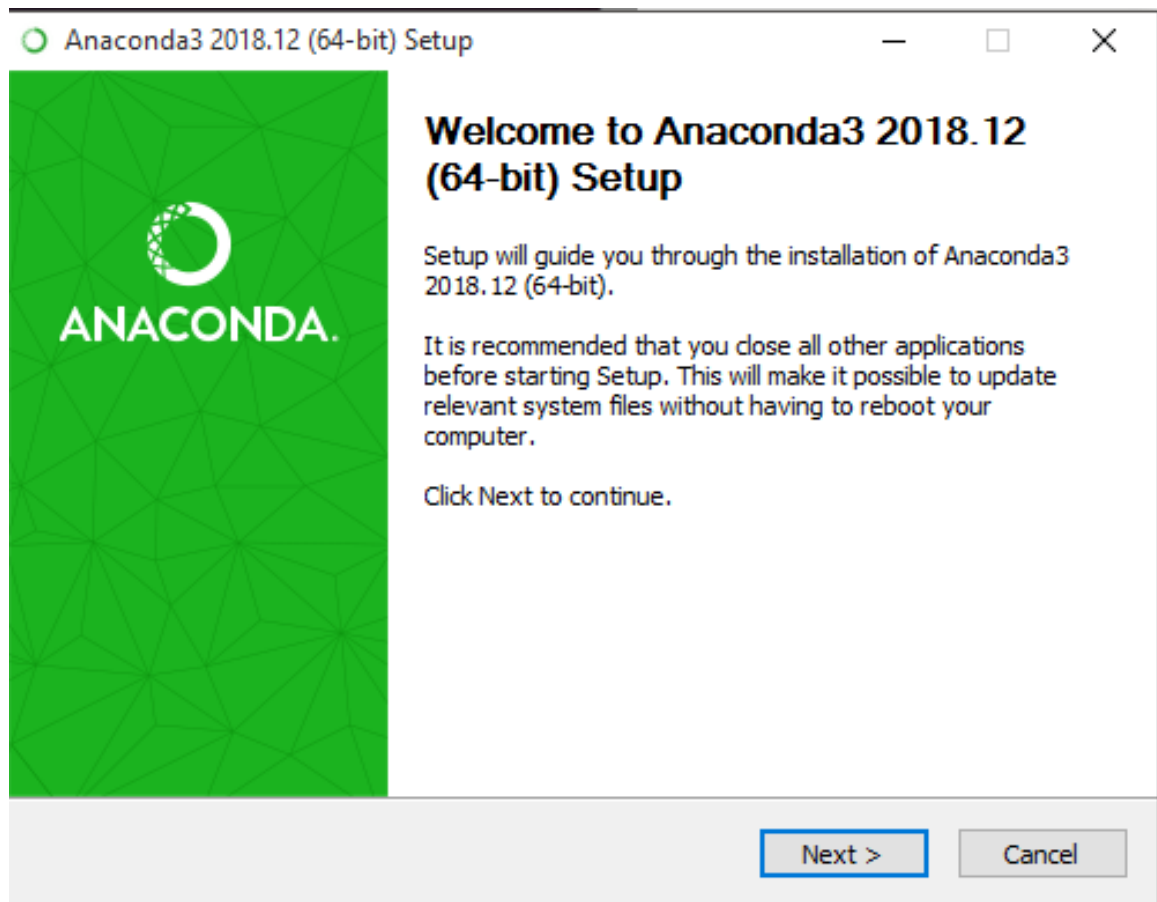


Figure 1.37: setelah membuka data instalasi klik next

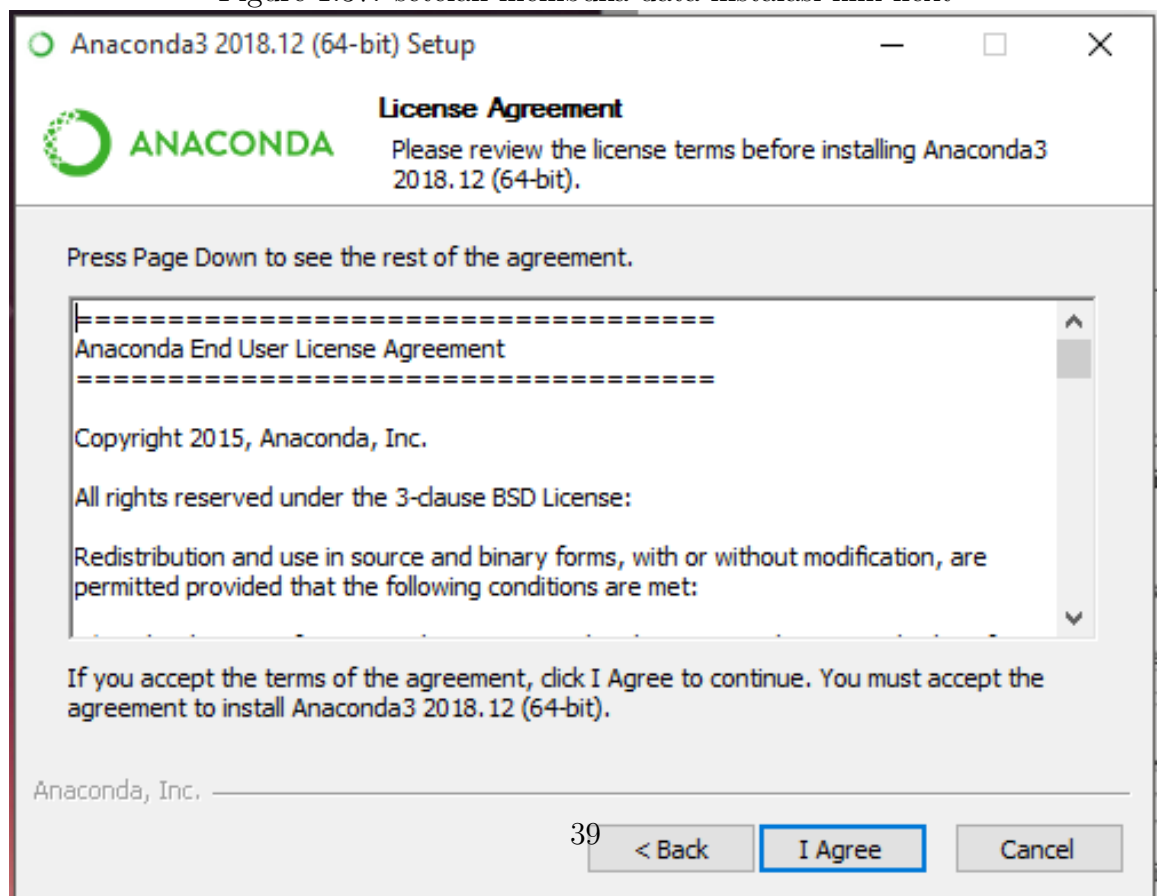


Figure 1.38: pilih i agree

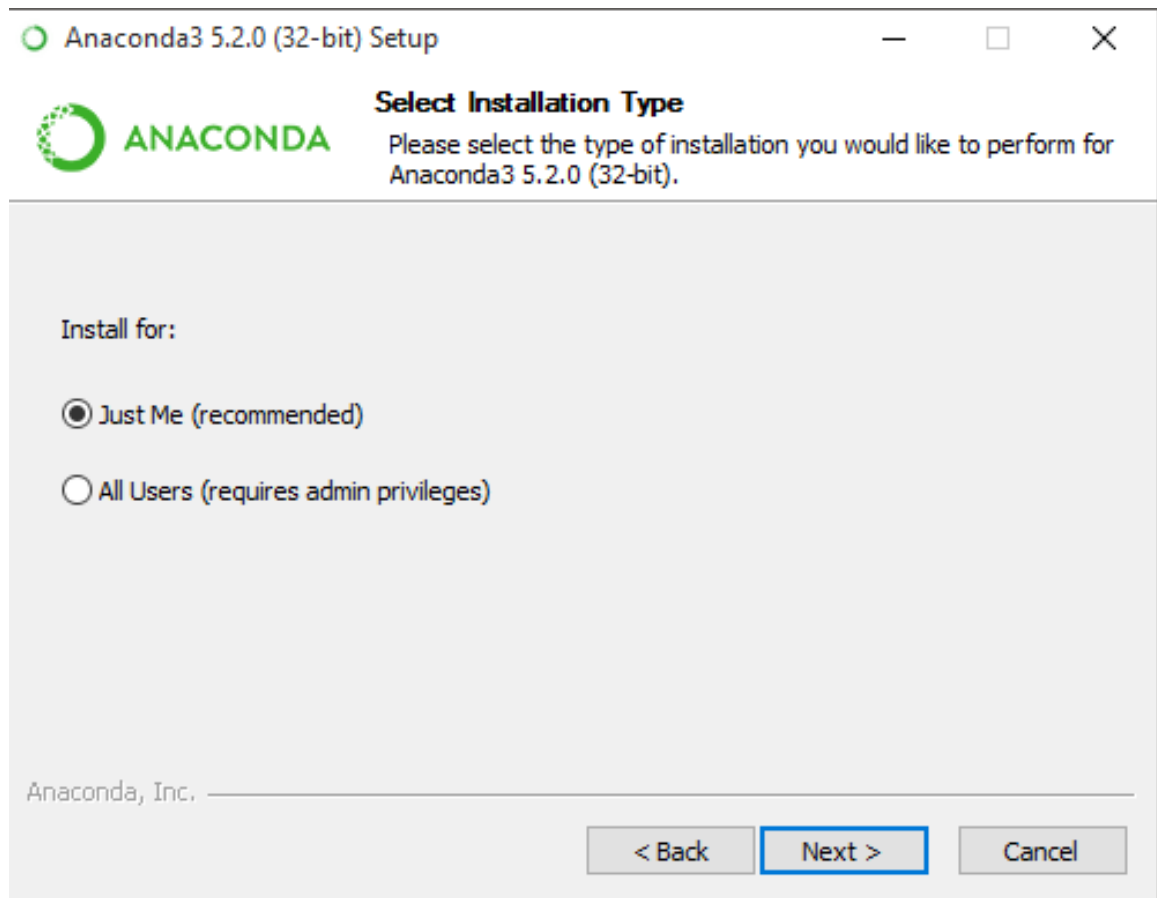


Figure 1.39: pilih instalasi Just Me

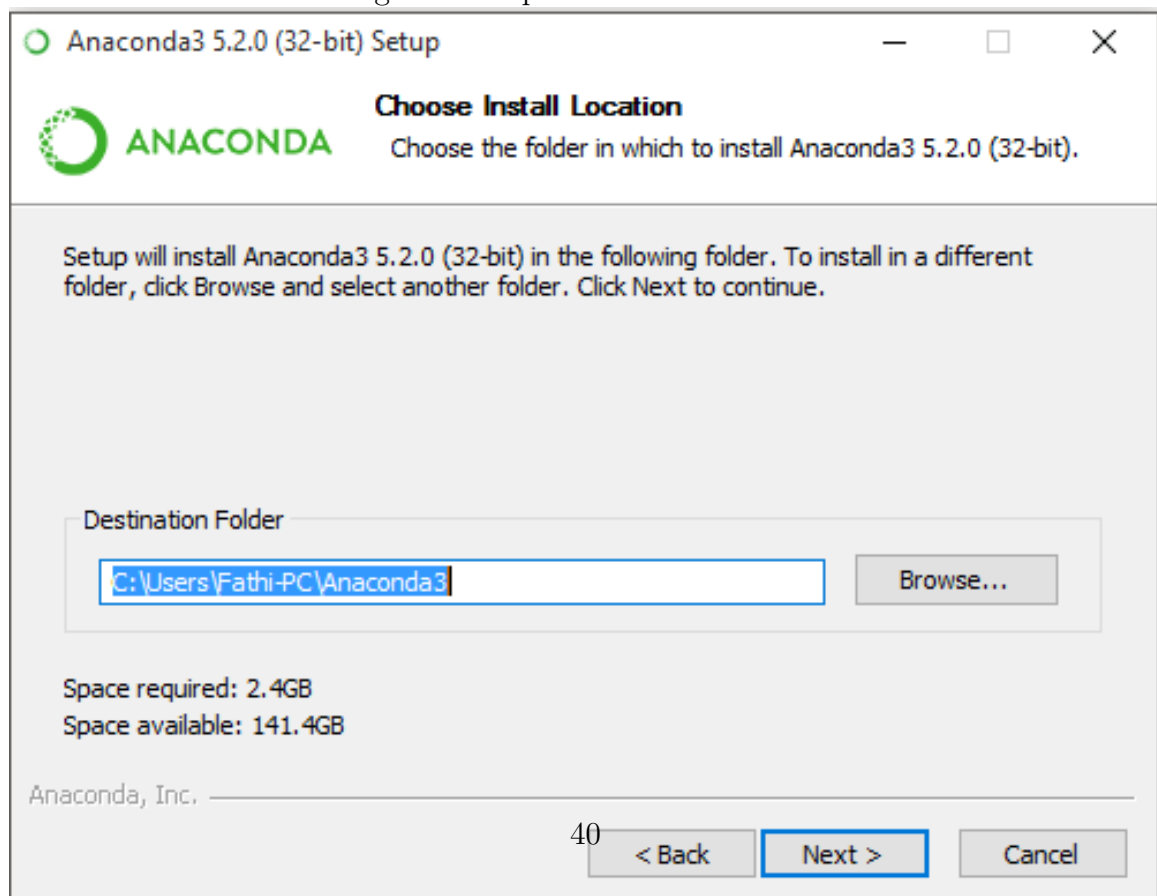


Figure 1.40: langsung saja next

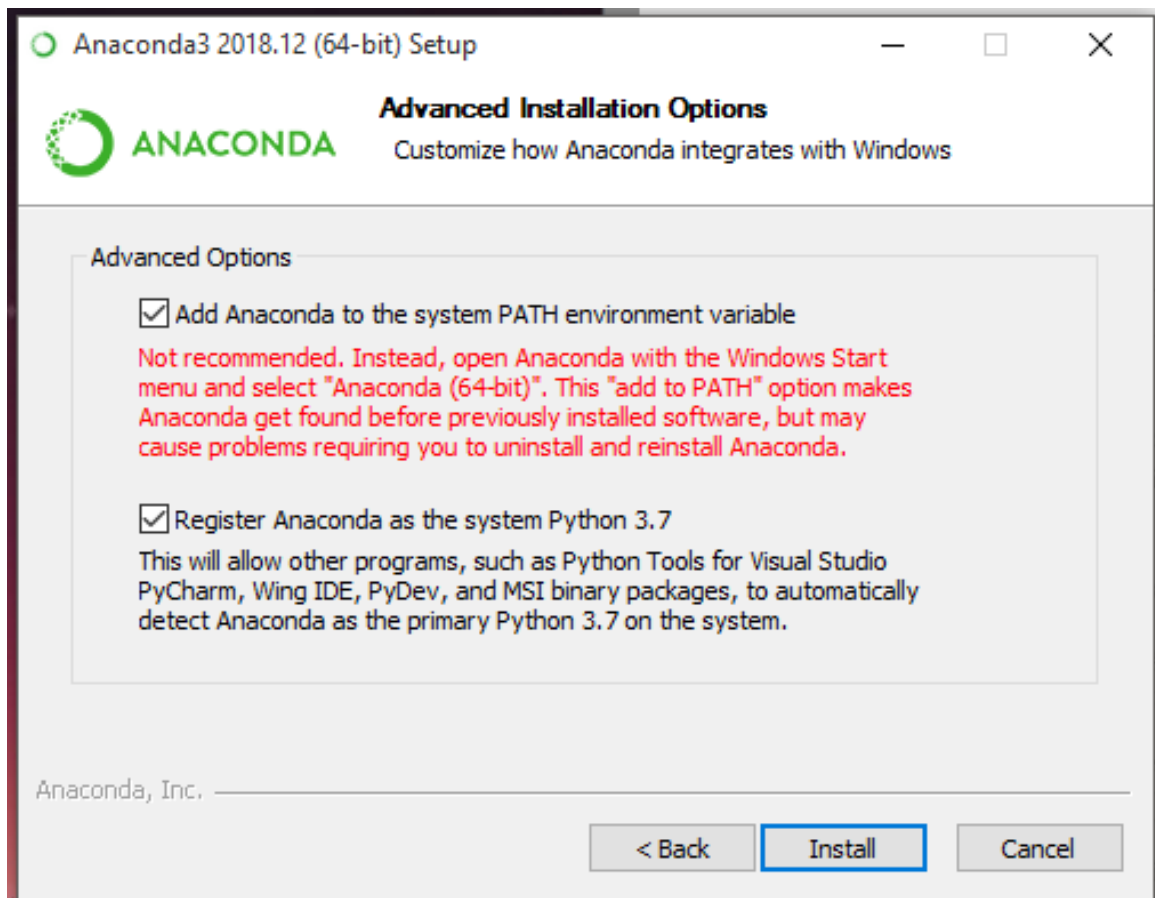


Figure 1.41: cek kedua pilihan tersebut

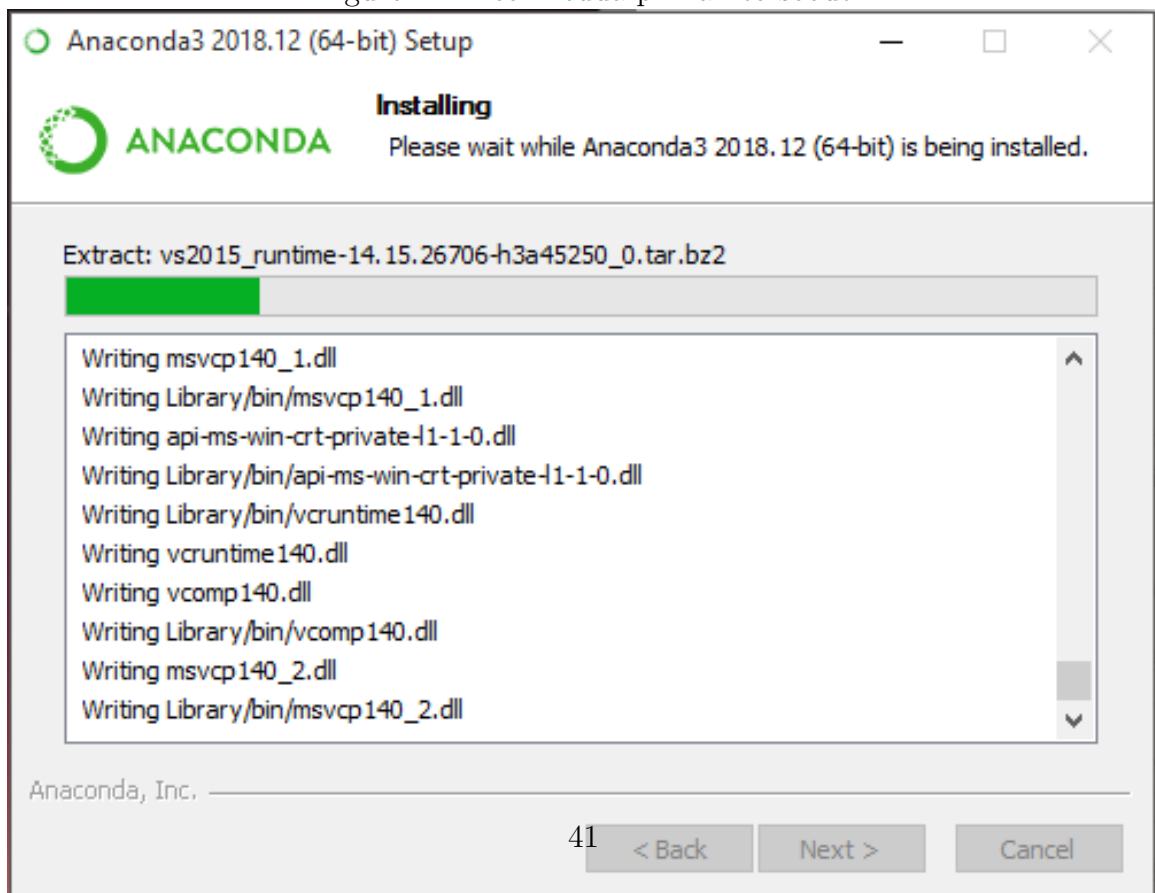


Figure 1.42: proses Instalasi

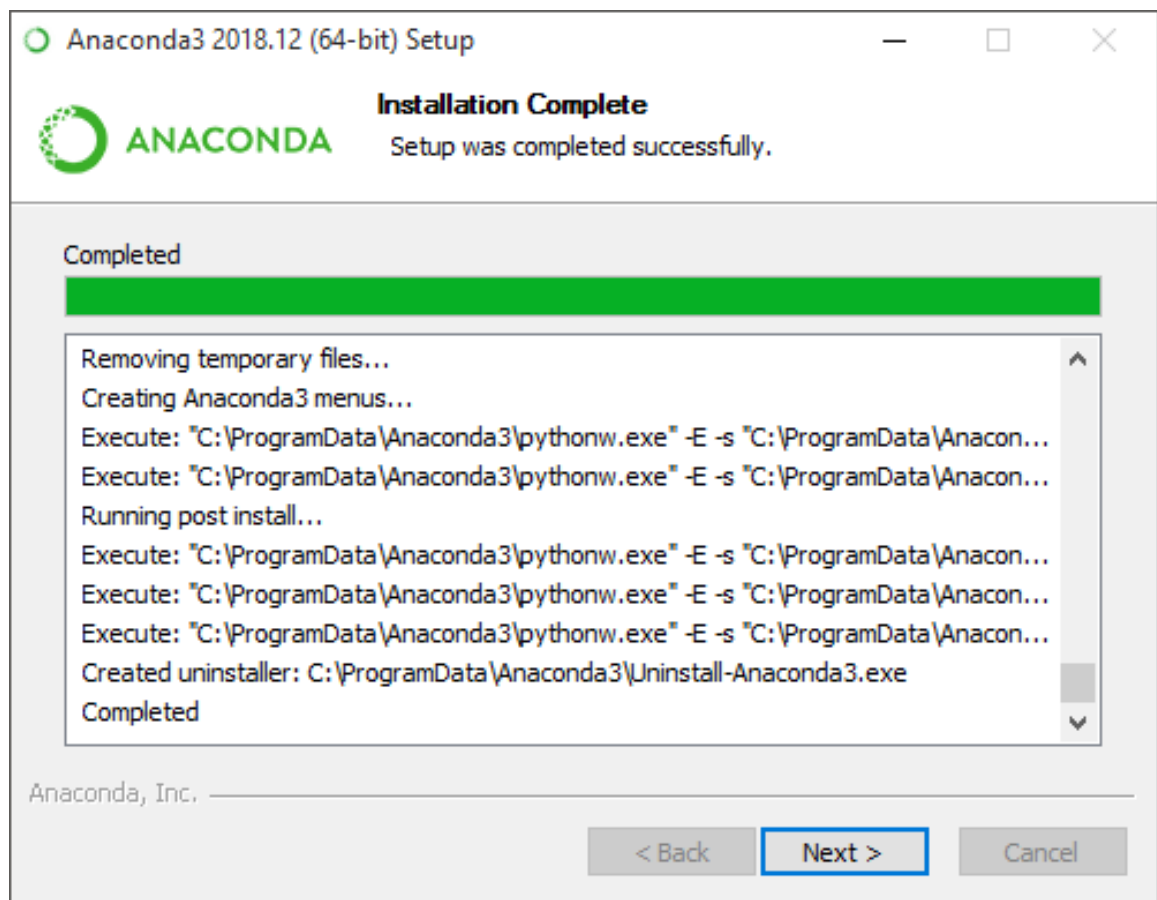


Figure 1.43: klik next

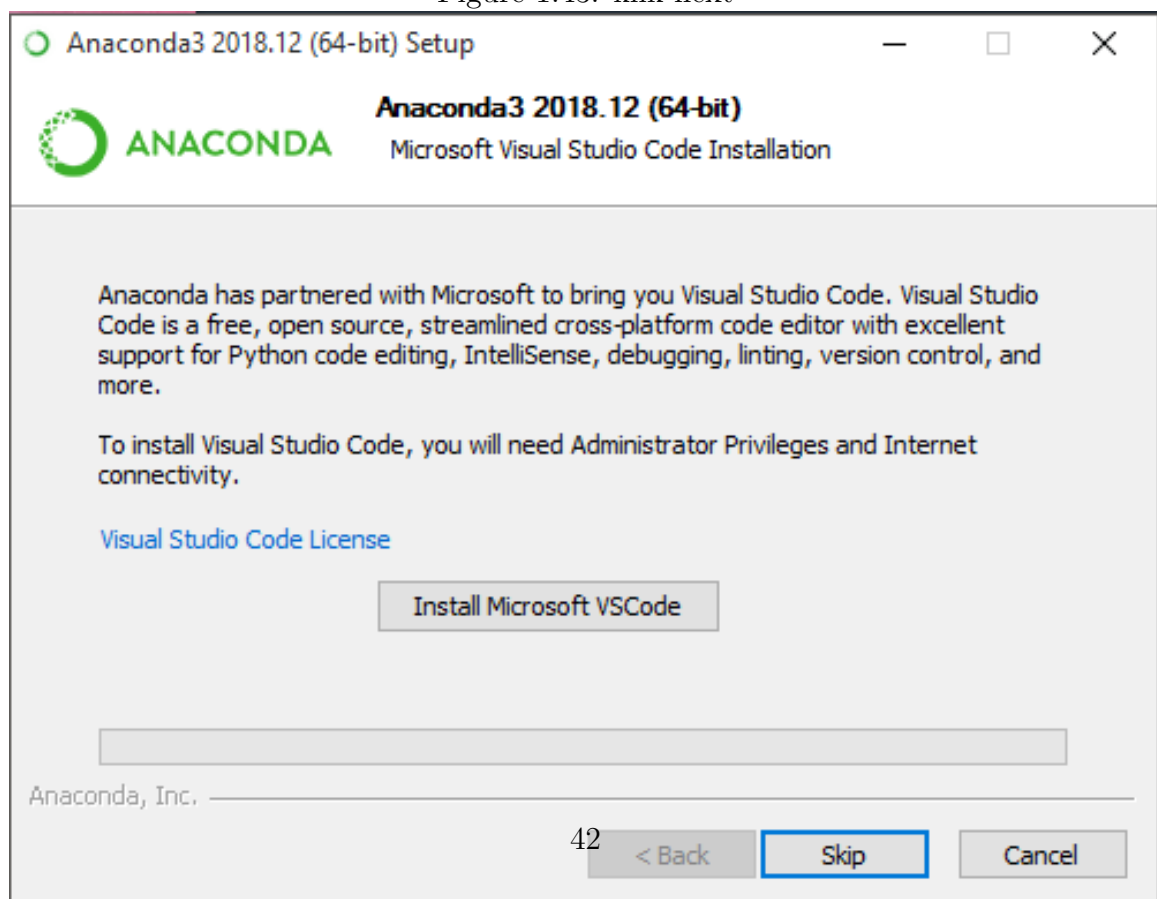


Figure 1.44: selesai instalasi anaconda

```

C:\Users\Fathi-PC>conda --version
conda 4.5.4

C:\Users\Fathi-PC>python --version
Python 3.6.5 :: Anaconda, Inc.

C:\Users\Fathi-PC>conda install scikit-learn
Solving environment: done

```

Figure 1.45: Instalasi SCIKIT dengan menggunakan anaconda

```

Solving environment: done

## Package Plan ##

  environment location: C:\Users\Fathi-PC\Anaconda3

  added / updated specs:
    - scikit-learn

The following packages will be downloaded:

  package                        | build                |
  -----|-----|
  conda-4.6.7                    | py36_0              | 1.7 MB

The following packages will be UPDATED:

  conda: 4.5.4-py36_0 --> 4.6.7-py36_0

Proceed ([y]/n)? y

```

Figure 1.46: Konfirmasi Instalasi

```

Downloading and Extracting Packages
conda-4.6.7 | 1.7 MB | ##### | 100%
Preparing transaction: done
Verifying transaction: done
Executing transaction: done

```

Figure 1.47: hasil dari instalasi SCIKIT

| Variable explorer |             |      |                                      |
|-------------------|-------------|------|--------------------------------------|
| Name              | Type        | Size | Value                                |
| digits            | utils.Bunch | 5    | Bunch object of sklearn.utils module |
| iris              | utils.Bunch | 5    | Bunch object of sklearn.utils module |

Figure 1.48: data variable explorer

```

from sklearn import datasets
iris = datasets.load_iris()
digits = datasets.load_digits()

print(digits.data)

```

Figure 1.49: code example dataset yang digunakan

```

In [14]: runfile('C:/Users/Fathi-PC/.spyder-py3/temp.py', wdir='C:/Users/Fathi-PC/.spyder-py3')
[[ 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.]]

```

Figure 1.50: data hasil dari code example dataset yang digunakan



```
C:\> Command Prompt

Microsoft Windows [Version 10.0.17134.590]
(c) 2018 Microsoft Corporation. All rights reserved.

C:\Users\COKRO>conda --version
conda 4.6.7

C:\Users\COKRO>python --version
Python 3.6.5 :: Anaconda, Inc.

C:\Users\COKRO>
```

Figure 1.51: Tampilan Versi Python dan Anaconda .

```
Administrator: Command Prompt

Microsoft Windows [Version 10.0.17134.590]
(c) 2018 Microsoft Corporation. All rights reserved.

C:\WINDOWS\system32>python --version
Python 3.6.5 :: Anaconda, Inc.

C:\WINDOWS\system32>conda --version
conda 4.5.4

C:\WINDOWS\system32>pip install -U scikit-learn
Collecting scikit-learn
  Using cached https://files.pythonhosted.org/packages/ee/c8/c89ebdc0d7dbba6e6fd222daabd257da3c28a967dd7c352d4272b2e1cef6/scikit_learn-0.20.2-cp36-cp36m-win32.whl
Requirement not upgraded as not directly required: numpy>=1.8.2 in c:\programdata\anaconda3\lib\site-packages (from scikit-learn) (1.14.3)
Requirement not upgraded as not directly required: scipy>=0.13.3 in c:\programdata\anaconda3\lib\site-packages (from scikit-learn) (1.1.0)
distributed 1.21.8 requires msgpack, which is not installed.
Installing collected packages: scikit-learn
  Found existing installation: scikit-learn 0.19.1
    Uninstalling scikit-learn-0.19.1:
      Successfully uninstalled scikit-learn-0.19.1
Successfully installed scikit-learn-0.20.2
You are using pip version 10.0.1, however version 19.0.3 is available.
You should consider upgrading via the 'python -m pip install --upgrade pip' command.
```

Figure 1.52: Instalikasi Library Sikic.

```

C:\WINDOWS\system32>conda install scikit-learn
Solving environment: done

## Package Plan ##

  environment location: C:\ProgramData\Anaconda3

  added / updated specs:
    - scikit-learn

The following packages will be UPDATED:

    conda: 4.5.4-py36_0 --> 4.6.7-py36_0

Proceed ([y]/n)? y

Preparing transaction: done
Verifying transaction: done
Executing transaction: done

C:\WINDOWS\system32>

```

Figure 1.53: Instalasi Library Sikic Melalui Conda

```

C:\Users\COKRO>python
Python 3.6.5 [Anaconda, Inc.] (default, Mar 29 2018, 13:23:52) [MSC v.1900 32 bit (Intel)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> print("Hello Anaconda!")
Hello Anaconda!
>>>

```

Figure 1.54: Console Python Include Anaconda

```

Command Prompt - python

C:\Users\COKRO>python
Python 3.6.5 [Anaconda, Inc.] (default, Mar 29 2018, 13:23:52) [MSC v.1900 32 bit (Intel)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> from sklearn import datasets
>>> iris = datasets.load_iris()
>>> digits = datasets.load_digits()
>>> print(digits.data)
[[ 0.  0.  5. ...  0.  0.  0.]
 [ 0.  0.  0. ... 10.  0.  0.]
 [ 0.  0.  0. ... 16.  9.  0.]
 ...
 [ 0.  0.  1. ...  6.  0.  0.]
 [ 0.  0.  2. ... 12.  0.  0.]
 [ 0.  0. 10. ... 12.  1.  0.]]
>>> print(iris.data)
[[5.1 3.5 1.4 0.2]
 [4.9 3.  1.4 0.2]
 [4.7 3.2 1.3 0.2]
 [4.6 3.1 1.5 0.2]
 [5.  3.6 1.4 0.2]
 [5.4 3.9 1.7 0.4]
 [4.6 3.4 1.4 0.3]
 [5.  3.4 1.5 0.2]
 [4.4 2.9 1.4 0.2]
 [4.9 3.1 1.5 0.1]
 [5.4 3.7 1.5 0.2]
 [4.8 3.4 1.6 0.2]
 [4.8 3.  1.4 0.1]
 [4.3 3.  1.1 0.1]
 [5.8 4.  1.2 0.2]
 [5.7 4.4 1.5 0.4]
 [5.4 3.9 1.3 0.4]
 [5.1 3.5 1.4 0.3]
 [5.7 3.8 1.7 0.3]
 [5.1 3.8 1.5 0.3]
 [5.4 3.4 1.7 0.2]
 [5.1 3.7 1.5 0.4]
 [4.6 3.6 1.  0.2]
 [5.1 3.3 1.7 0.5]
 [4.8 3.4 1.9 0.2]
 [5.  3.  1.6 0.2]
 [5.  3.4 1.6 0.4]
 [5.2 3.5 1.5 0.2]]

```

Figure 1.55: Contoh Codingan Dataset

```
Command Prompt - python
C:\Users\COKRO>python
Python 3.6.5 [Anaconda, Inc.] (default, Mar 29 2018, 13:23:52) [MSC v.1900 32 bit (Intel)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> from sklearn import svm
>>> clf = svm.SVC(gamma=0.001, C=100.)
>>> clf.fit(digits.data[:-1], digits.target[:-1])
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
NameError: name 'digits' is not defined
>>>
```

Figure 1.56: Error Coding 1

```
Command Prompt - python
Microsoft Windows [Version 10.0.17134.590]
(c) 2018 Microsoft Corporation. All rights reserved.

C:\Users\COKRO>python
Python 3.6.5 [Anaconda, Inc.] (default, Mar 29 2018, 13:23:52) [MSC v.1900 32 bit (Intel)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> from joblib import dump, load
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
ModuleNotFoundError: No module named 'joblib'
>>>
```

Figure 1.57: Error Coding 2

```
Command Prompt - python
C:\Users\COKRO>python
Python 3.6.5 [Anaconda, Inc.] (default, Mar 29 2018, 13:23:52) [MSC v.1900 32 bit (Intel)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> from sklearn import datasets
>>> iris = datasets.load_iris()
>>> digits = datasets.load_digits()
>>> from sklearn import svm
>>> clf = svm.SVC(gamma=0.0001, C=100.)
>>> 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=0.0001, kernel='rbf',
    max_iter=-1, probability=False, random_state=None, shrinking=True,
    tol=0.001, verbose=False)
>>> clf.predict(digits.data[-1:])
array([8])
>>>
```

Figure 1.58: Codingan Solusi Untuk Error digits

```
Administrator: Command Prompt
Microsoft Windows [Version 10.0.17134.590]
(c) 2018 Microsoft Corporation. All rights reserved.

C:\WINDOWS\system32>pip install joblib
Collecting joblib
  Downloading https://files.pythonhosted.org/packages/cd/c1/50a758e8247561e58cb87305b1e90b171b8c767b15b12a1734001f41d356
/joblib-0.13.2-py2.py3-none-any.whl (278kB)
    100% |#####| 286kB 2.3MB/s
distributed 1.21.8 requires msgpack, which is not installed.
Installing collected packages: joblib
Successfully installed joblib-0.13.2
You are using pip version 10.0.1, however version 19.0.3 is available.
You should consider upgrading via the 'python -m pip install --upgrade pip' command.

C:\WINDOWS\system32>
```

Figure 1.59: Codingan Solusi Untuk Error Joblib

```

Command Prompt - python
Microsoft Windows [Version 10.0.17134.590]
(c) 2018 Microsoft Corporation. All rights reserved.

C:\Users\COKRO>python
Python 3.6.5 [Anaconda, Inc.] (default, Mar 29 2018, 13:23:52) [MSC v.1900 32 bit (Intel)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> from joblib import dump, load
>>>

```

Figure 1.60: Hasil Solusi Error Joblib

```

>>> from sklearn import svm
>>> from sklearn import datasets
>>> clf = svm.SVC(gamma='scale')
>>> iris = datasets.load_iris()
>>> X, y = iris.data, iris.target
>>> clf.fit(X, y)
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
    File "C:\Users\Fathi-PC\Anaconda3\lib\site-packages\sklearn\svm\base.py", line 187,
in fit
    fit(X, y, sample_weight, solver_type, kernel, random_seed=seed)
    File "C:\Users\Fathi-PC\Anaconda3\lib\site-packages\sklearn\svm\base.py", line 254,
in _dense_fit
    max_iter=self.max_iter, random_seed=random_seed)
    File "sklearn\svm\libsvm.pyx", line 58, in sklearn.svm.libsvm.fit
TypeError: must be real number, not str

```

Figure 1.61: Error Type data, yang harus digunakan number sedangkan isinya 'SCALE' pada gamma

```

>>> from joblib import dump, load
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
ModuleNotFoundError: No module named 'joblib'

```

Figure 1.62: Error no Module found, modul yang dicari tidak ditemukan atau tidak ada 'JOBLIB'

# Chapter 2

## Related Works

Your related works, and your purpose and contribution which must be different as below.

### 2.1 Cokro Edi Prawiro/ 1164069

#### 2.1.1 Teori

1. Jelaskan Apa Itu binari calssification drlengkapi ilustrasi gambar sendiri.

Binary Classification atau binomial adalah tugas mengklasifikasikan unsur-unsur dari himpunan yang diberikan ke dalam kedua kelompok berdasarkan aturan klasifikasi yang telah ditetapkan. binari classification juga dapat diartikan sebagai pembagi yang hanya memberikan dua pilihan contohnya benar dan salah atau klasifikasi tingkat panjang atau pendek. penjelasan lebih singkatnya binari classification merupakan kegiatan mengklasifikasikan yang hanya memberikan dua class. contoh pada gambar 2.53 klasifikasi antara betuk kotak dan segitiga.

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

supervised learning adalah cara untuk mengklasifikasikan suatu objek atau data yang telah ditentukan kelas-kelasnya contoh pada sayuran tumbuhan wortel termasuk yang mengandung vitamin A berarti tumbuhan wortel telah di kategorikan ke dalam sayuran yang mengandung vitamin A. sedangkan kangkung mengandung zat besi yang berarti tumbuhan kangkung telah di kategorikan ke dalam sayuran yang mengandung zat besi untuk lebih jelasnya dapat dilihat pada gambar 2.54.

unsupervised learning merupakan cara untuk mengklasifikasi tanpa adanya kelas untuk menentukan jenisnya contoh sayuran berarti semua objek yang memiliki ciri ciri sayuran di kategorikan kedalam sayuran untuk lebih jelasnya dapat dilihat pada gambar 2.55.

clustering merupakan peroses mengklasifikasikan yang berdasarkan suatu parameter dalam penentuannya contoh pada berat sayuran sayuran A memiliki berat 100 gr dan sayuran B memiliki berat 120 gr yang berarti berat sayuran dibagi dua parameter yaitu lebih kecil samadengan 100 gram dan lebih besar dari gram contoh pada gambar 2.56.

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

evaluasi adalah pengumpulan pengumpulan dan pengamatan dari berbagai macam bukti untuk mengukur dampak efektifitas dari suatu objek, program, atau proses berkaitan dengan spesifikasi atau persyaratan yang telah di tetapkan sebelumnya. sedangkan akurasi itu sendiri merupakan bagian dari evaluasi yang merupakan ketepatan data terhadap suatu objek berdasarkan kriteria tertentu. kita dapat mengevaluasi seberapa baik model bekerja dengan mengukur akurasinya. ketepatan akan di definisikan sebagai presentase kasus yang di klasifikasikan dengan benar. hal ini berkaitan dengan confusion matrix pada materi selanjutnya. contoh evaluasi untuk membedakan burung dengan ayam terdapat parameter yaitu ukuran badan dan fungsi sayap pada hewan tersebut. lebih jelanya pada gambar 2.57 berikut:

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

Dalam pembuatan confusion matrix tentukan parameter atau objek yang akan di evaluasi contoh bunga melati , bunga mawar, dan bunga kenangan buat tabel dengan baris dan kolom berjumlah tiga kemudian tentukan nilai miring pada setiap kolom tersebut disini saya memberi nilai 30 dengan ketentuan setiap baris harus berisi nilai 30 nilai tersebut jika terbagi ke kolom lain maka jumlahnya harus bernilai 30 jika tidak berarti data tersebut tidak akurat. untuk lebih jelanya dapat dilihat pada gambar 2.58 berikut :

5. Jelaskan bagaimana K-fold cross validation bekerja dengan gambar ilustrasi contoh buatan sendiri. K-fold Cross Validation merupakan cara untuk melatih suatu mesin dimana di dalamnya terdapat data set yang dibagi menjadi dua

yaitu untuk data testing dan data training contoh 1000 data merupakan data set dan 200 data digunakan untuk data testing kemudian 800 datanya digunakan untuk data training dimana data training tersebut digunakan untuk menentukan nilai bobot yang dimasukan kedalam rumus regresi linier. sedangkan nilai testing akan dijadikan nilai inputan untuk rumus regresi linier. contohnya dapat dilihat pada gambar 2.59 berikut :

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

Decision tree (pohon keputusan) merupakan implementasi dari binari classification dimana pada pohon keputusan akan terdapat root atau akar dan cabang cabangnya yang nilainya seperti if contoh pada root berisi nilai jenis kelamin, apakah perempuan pada cabang satu bernilai iya dan pada cabang dua bernilai tidak jika nilainya iya berarti jenis kelaminnya perempuan dan jika tidak maka bernilai laki-laki. agar lebih jelas dapat dilihat pada gambar 2.60 decision tree berikut:

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

informasion gain merupakan informasi atau kriteria dalam pembagian sebuah objek contoh information gain pada laki-laki yaitu berrambut pendek, memiliki jakun, berjenggot, berkumis, dan mempunyai bahu yang lebar. pada kriteria tersebut seringkali terdapat bias misalkan ada perempuan yang berrambut pendek atau berkumis namun dari parameter tersebut dapat dilihat bahwa 60 persen parameter tersebut tepat pada sasarnya selama parameter itu bernilai tinggi untuk tepat maka dapat digunakan itulah information gain untuk lebih jelasnya dapat dilihat pada gambar 2.61 berikut :

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

### 2.1.2 Sikic-Learn /Cokro Edi Prawiro/1164069

1. pada surcode pertama yang dapat dilihat pada gambar 2.1 pada baris pertama di tuliskan

```
import pandas as baso
```

yang berarti mengimport library pandas yang di inialisasi namanya menjadi baso. selanjutnya pada baris ke dua codingan tersebut berisi

```
cireng = baso.read_csv  
( 'E:\KULIAH\semester_6\AI\Buku\Chapter01\dataset\student-por.csv', sep=';')
```

pada code tersebut terdapat variabel cireng yang berisi inialisasi pandas (baso) dengan aksi untuk membaca vfile csv yang terdapat pada direktori pada komputer kemudian terdapat sep samadengan titik koma yang berarti pemisah field di dalam vile tersebut merupakan titik koma. kemudian pada bagian akhir terdapat code len (nama variabel) yang berarti akan menghitung jumlah baris pada file tersebut. untuk hasilnya dapat dilihat pada gambar 2.2

```
# In[1]:  
  
# Load dataset (student Portuguese scores)  
import pandas as baso  
cireng = baso.read_csv('E:\KULIAH\semester_6\AI\Buku\Chapter01\dataset\student-por.csv', sep=';')  
len(cireng)
```

Figure 2.1: Source Code Load Dataset

```
In [1]: import pandas as baso  
...: cireng = baso.read_csv('E:\KULIAH\semester_6\AI\Buku\Chapter01\dataset\student-por.csv', sep=';')  
...: len(cireng)  
Out[1]: 649
```

Figure 2.2: Hasil Load Dataset

2. Pada code selanjutnya akan ditambahkan fungsi untuk lulus atau gagal yang dibuat berupa kolom kolom yang di dalamnya terdapat nilai 0 dan 1 dimana 0 berarti gagal dan 1 berarti lulus. kemudian variabel pada codingan sebelumnya masih digunakan yaitu pada codingan berikut

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

. dimana variabel cireng digunakan karena berisi nilai file csv kemudian dilakukan eksekusi dengan parameter G1, G2, dan G3 dengan fungsi lambda yang berarti if bersarang atau if didalam if yang berarti nilai yang di eksekusi akan dilempar ke dalam setiap paramater sesuai dengan kriteria dan axis=1



yaitu nilai tersebut akan digunakan dari tiap baris data. sedangkan pada codingan

```
cireng = cireng.drop(['G1', 'G2', 'G3'], axis=1)
```

variabel cireng di berikan aksi drop yaitu penurunan nilai pada bagian baris data. dan pada code cireng.head () yaitu untuk mengeksekusi codingan sebelumnya untuk lebih jelasnya dapat dilihat pada gambar 2.3 dan hasilnya seperti pada gambar 2.4 berikut :

```
# In[2]:

# generate binary label (pass/fail) based on G1+G2+G3 (test grades, each 0-20 pts); threshold for passing
cireng['pass'] = cireng.apply(lambda row: 1 if (row['G1']+row['G2']+row['G3']) >= 35 else 0, axis=1)
cireng = cireng.drop(['G1', 'G2', 'G3'], axis=1)
cireng.head()
```

Figure 2.3: memberikan nilai satau atau nol

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

[5 rows x 31 columns]
```

Figure 2.4: hasil dari memberikan nilai nol dan satu

3. pada kodingan selanjutnya diguanakn untuk menambahkan nilai numerik berupa 0 dan 1 yang dirubah dari nilai tidak dan iya hal ini merupakan fungsi dari codingan get\_dummies pada baris pertama pada gambar 2.5 yang nilainya diambil dari variabel cireng yang telah di dekralasikan tadi banyaknya data numerik itu sendiri tergantung pada field dari kolom yang di camtumkan pada codingan dengan catatan field tersebut harus ada dalam data csv yang di import oleh codingan pertama tadi maka hasilnya akam merubah data dalam field tersebut menjadi 0 dan 1 untuk lebih jelasnya dapat di lihat pada gambar 2.6 berikut;
4. selanjutnya pada codingan selanjutnya yaitu menentukan data training dan data testing dari dataset dimana variabel cireng yang berisi data csv tadi dibuat perbandingan 500 untuk data training dan sisanya yaitu 149 digunakan untuk

```
# In[3]:

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

cireng.head()
```

Figure 2.5: Penambahan nilai numerik

```
In [3]: cireng = baso.get_dummies(cireng, columns=['sex', 'school', 'address', 'famsize', 'Pstatus', 'Mjob', 'Fjob',
...:                                     'reason', 'guardian', 'schoolsup', 'famsup', 'paid', 'activities',
...:                                     'nursery', 'higher', 'internet', 'romantic'])
...: cireng.head()
Out[3]:
```

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

```
[5 rows x 57 columns]
```

Figure 2.6: hasil Penambahan nilai numerik

data testing hal ini di lakukan pada baris ke 1 sampai ke 3 pada gambar 2.7 kemudian nilai tersebut di turunkan brdasarkan baris pada data set hal itu dapat dilihat pada baris ke 4 sampai ke 9 pada gambar 2.7 kemudian selanjutnya mengimport library numpy yang berguna untuk oprasi vektor dan matrix karna data diatas berupa data matrix maka library ini di gunakan. untuk hasilnya dapat dilihat pada gambar 2.8 dan 2.9

```
# In[4]:

# shuffle rows
cireng = cireng.sample(frac=1)
# split training and testing data
cireng_train = cireng[:500]
cireng_test = cireng[500:]

cireng_train_att = cireng_train.drop(['pass'], axis=1)
cireng_train_pass = cireng_train['pass']

cireng_test_att = cireng_test.drop(['pass'], axis=1)
cireng_test_pass = cireng_test['pass']

cireng_att = cireng.drop(['pass'], axis=1)
cireng_pass = cireng['pass']

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

Figure 2.7: penentuan data training dan data testing

5. selanjutnya yaitu membuat pohon keputusan dapat lebih jelasnya dapat di lihat pada gambar 2.10. pada baris pertama yairu memasukan librari tree kemudian pada baris kedua dibuat variabel tempe dengan nilai DecisionTreeClassifier yang merupakan paket atau fungsi dari scikit-learn yang merupakan class

```

In [4]: cireng = cireng.sample(frac=1)
...: # split training and testing data
...: cireng_train = cireng[:500]
...: cireng_test = cireng[500:]
...:
...: cireng_train_att = cireng_train.drop(['pass'], axis=1)
...: cireng_train_pass = cireng_train['pass']
...:
...: cireng_test_att = cireng_test.drop(['pass'], axis=1)
...: cireng_test_pass = cireng_test['pass']
...:
...: cireng_att = cireng.drop(['pass'], axis=1)
...: cireng_pass = cireng['pass']
...:
...: # number of passing students in whole dataset:
...: import numpy as nasipadang
...: print("Passing: %d out of %d (%.2f%%)" % (nasipadang.sum(cireng_pass), len(cireng_pass),
100*float(nasipadang.sum(cireng_pass)) / len(cireng_pass)))
Passing: 328 out of 649 (50.54%)

```

Figure 2.8: Hasil 1 penentuan data training dan data testing

| Name              | Type      | Size      | Value                                                                      |
|-------------------|-----------|-----------|----------------------------------------------------------------------------|
| cireng            | DataFrame | (649, 57) | Column names: age, Medu, Fedu, traveltime, studytime, failures, famrel ... |
| cireng_att        | DataFrame | (649, 56) | Column names: age, Medu, Fedu, traveltime, studytime, failures, famrel ... |
| cireng_pass       | Series    | (649,)    | Series object of pandas.core.series module                                 |
| cireng_test       | DataFrame | (149, 57) | Column names: age, Medu, Fedu, traveltime, studytime, failures, famrel ... |
| cireng_test_att   | DataFrame | (149, 56) | Column names: age, Medu, Fedu, traveltime, studytime, failures, famrel ... |
| cireng_test_pass  | Series    | (149,)    | Series object of pandas.core.series module                                 |
| cireng_train      | DataFrame | (500, 57) | Column names: age, Medu, Fedu, traveltime, studytime, failures, famrel ... |
| cireng_train_att  | DataFrame | (500, 56) | Column names: age, Medu, Fedu, traveltime, studytime, failures, famrel ... |
| cireng_train_pass | Series    | (500,)    | Series object of pandas.core.series module                                 |

Figure 2.9: hasil 2 penentuan data training dan data testing

yang mampu melakukan multi class. sedangkan max\_depth=5 merupakan untuk penyesuaian data terhadap pohon keputusan itu sendiri. untuk hasilnya dapat dilihat pada gambar 2.11

- selanjutnya yaitu pembuatan gambar dari pohon keputusan yang tadi di buta pada codingan sebelumnya pada baris pertama codingan ya itu mengimport library graphviz kemudian pada baris ke dua yaitu pemberian nilai pada variabel baru dot data nilainya diambil dari pembuatan pohon keputusan tadi kemudian di tentukannya nilai benar dan salah dari codingan tersebut setelah itu dibuatlah sebuah variabel untuk menampung hasil eksekusi tersebut kemudian variabel tersebut di running untuk lebih jelasnya dapat di lihat pada gambar 2.12 kemudian hasilnya dapat dilihat pada gambar 2.13.

```
# In[5]:

# fit a decision tree
from sklearn import tree
tempe = tree.DecisionTreeClassifier(criterion="entropy", max_depth=5)
tempe = tempe.fit(cireng_train_att, cireng_train_pass)
```

Figure 2.10: memberikan nilai pada pohon keputusan

```
In [5]: from sklearn import tree
...: tempe = tree.DecisionTreeClassifier(criterion="entropy", max_depth=5)
...: tempe = tempe.fit(cireng_train_att, cireng_train_pass)
```

Figure 2.11: hasil memberikan nilai pada pohon keputusan

7. selanjutnya pembuatan method untuk menyimpan data pohon dengan menarik data langsung dari pohon keputusan di buat tadi untuk code lebih jelasnya dapat dilihat pada gambar 2.14. kemudian untuk hasilnya dapat dilihat pada gambar 2.15 berikut.
8. selanjutnya pada codingan berikut yaitu digunakan untuk mencetak nilai score rata-rata dari ketepatan data yang telah di olah tadi lebih jelasnya dapat dilihat pada gambar 2.16 kemudian untuk hasilnya dapat dilihat pada gambar 2.17 tersebut:
9. selanjutnya yaitu digunakan untuk memeriksa akurasi dari ketepatan hasil pengolahan data tersebut maka akan didapat nilai rata-rata 60 persen dari hasil pengolahan data tersebut untuk lebih jelasnya dapat di lihat pada gambar 2.18 pada codingan tersebut pada baris ke satu melakukan import library dari sklearn kemudian pada baris selanjutnya mengisi nilai skor dengan nilai pada variabel tempe setelah hal tersebut dilakukan kemudian data tersebut di eksekusi. berikut merupakan hasil dari code tersebut dapat dilihat pada gambar 2.19.
10. membuat rank akurasi dari 1 sampai 20 untuk melihat akurasi data apakah data tersebut terdapat di rata rata 60 persen atau tidak dengan cara membuat lagi variabel baru dengan nilai tree diadalamnya jadi hampir mirip seperti membuat pohon keputusan namun ini langsung dalam bentuk rata rata akurasi yang lebih spesifik untuk lebih jelasnya dapat dilihat pada gambar 2.20 codingan berikut. dan untuk hasilnya dapat dilihat pada gambar 2.21 berikut.

```
# In[6]:

# visualize tree
import graphviz
dot_data = tree.export_graphviz(tempe, out_file=None, label="all", impurity=False, proportion=True,
                                feature_names=list(cireng_train_att), class_names=["fail", "pass"],
                                filled=True, rounded=True)
graph = graphviz.Source(dot_data)
graph
```

Figure 2.12: pembuatan diagram pohon keputusan

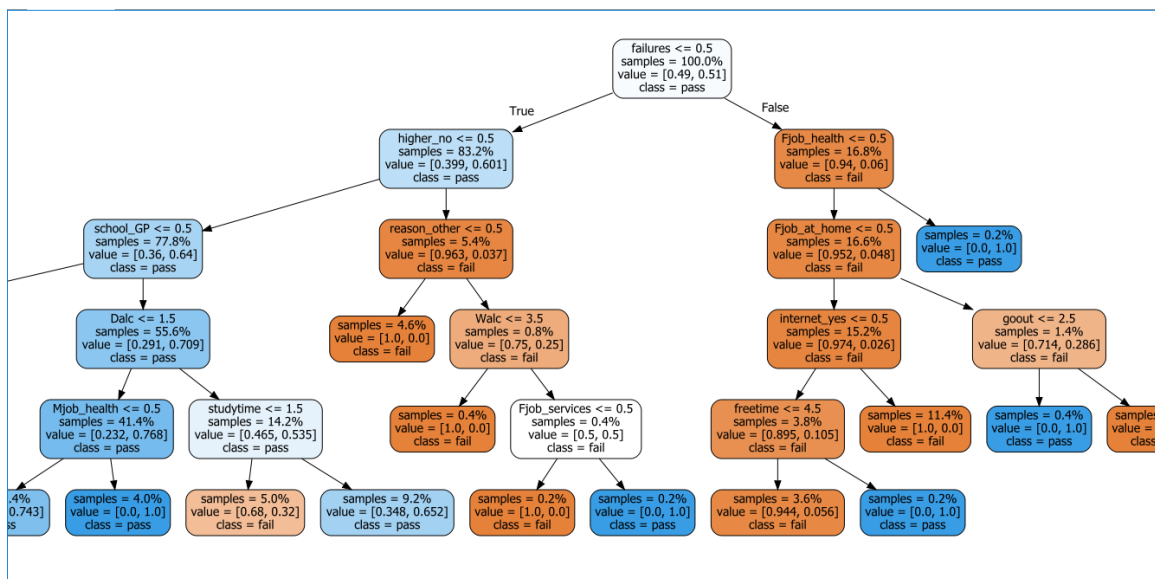


Figure 2.13: hasil pembuatan pohon keputusan

11. untuk selanjutnya yaitu menentukan nilai untuk grafik hampirsama dengan nilai akurasi tadi pertama tentukan rank atau batasan nilai itu disini batasannya di mulai dari 1 sampai 20 dengan menggunakan nilai tree tadi maka hasilnya dapat ditentukan kemudian buat variabel i untuk penomoran tiap record yang keluar atau record hasil dari eksekusi tree tersebut. untuk lebih jelasnya dapat dilihat pada gambar 2.22 dan untuk hasilnya dapat dilihat pada gambar 2.23.
12. terakhir yaitu pembuatan grafik untuk pembuatan grafik diambil data dari codingan sebelumnya yang 20 record tadi dengan cara mengimport library matplotlib.pyplot yang di inialisasi menjadi barwankemudian inialisasi tersebut di eksekusi. untuk lebih jelasnya codingannya seperti gambar 2.24 dan untuk hasilnya seperti gambar 2.25 berikut.

```
# In[7]:

# save tree
tree.export_graphviz(tempe, out_file="student-performance.dot", label="all", impurity=False, proportion=True,
                      feature_names=list(cireng_train_att), class_names=["fail", "pass"],
                      filled=True, rounded=True)
```

Figure 2.14: coding save

```
In [7]: tree.export_graphviz(tempe, out_file="student-performance.dot", label="all", impurity=False,
proportion=True,
...:                      feature_names=list(cireng_train_att), class_names=["fail", "pass"],
...:                      filled=True, rounded=True)
```

Figure 2.15: hasil coding save

### 2.1.3 Penanganan Error /Cokro Edi Prawiro/1164069

1. skrinshot error dapat dilihat pada gambar 2.26
2. kode error dan jenis errornya .

```
import graphviz
dot_data = tree.export_graphviz(tempe, out_file=None, label="all", impurity=False,
                                feature_names=list(cireng_train_att), class_names=["fail", "pass"],
                                filled=True, rounded=True)

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

pada codingan tersebut error karena graphviznya belum di install

3. Solusi pemecahan masalah

buka CMD komputer anda run as administrator kemudian masukan perintah conda install graphviz kemudian tekan enter ingat hal ini dilakukan harus terkoneksi dengan jaringan internet. untuk hasilnya dapat dilihat pada gambar 2.27 dan gambar 2.28 setelah itu masukan PATH graphviz dengan cara masuk ke direktori graphviz itu di simpan kalau di komputer saya disimpan di

C:\ProgramData\Anaconda3\Library\bin\graphviz

kemudian setelah itu copy alamat direktori tersebut dan masukan kedalam path seperti pada gambar 2.29 dan pada gambar 2.30.

```
# In[8]:  
  
tempe.score(cireng_test_att, cireng_test_pass)
```

Figure 2.16: Contoh Binary Classification

```
In [8]: tempe.score(cireng_test_att, cireng_test_pass)  
Out[8]: 0.6577181208053692
```

Figure 2.17: hasil coding save

## 2.2 Ahmad Syafrizal Huda/1164062

### 2.2.1 Teori

1. Binary Classification yaitu katakanlah kita memiliki tugas untuk mengklasifikasi objek menjadi dua kelompok berdasarkan beberapa fitur. Sebagai contoh, katakanlah kita diberi beberapa pena dan pensil dari berbagai jenis dan merek, kita dapat dengan mudah memisahkannya menjadi dua kelas, yaitu pena dan pensil.

Contoh ilustrasi gambar bisa dilihat pada gambar 2.45.

2. Supervised learning merupakan sebuah pendekatan dimana sudah terdapat data yang dilatih, dan terdapat variable yang ditargetkan sehingga tujuan dari pendekatan ini adalah mengelompokkan suatu data ke data yang sudah ada. Sedangkan unsupervised learning tidak memiliki data latih, sehingga dari data yang ada, kita mengelompokkan data tersebut menjadi 2 bagian atau 3 bagian dan seterusnya. Dan clustering adalah proses pengelompokan entitas yang sama bersama-sama. Tujuan dari teknik pembelajaran mesin tanpa pengawasan ini adalah untuk menemukan kesamaan pada titik data dan mengelompokkan titik data yang serupa secara bersamaan[6].

Contoh ilustrasi gambar bisa dilihat pada gambar 2.46.

Contoh ilustrasi gambar bisa dilihat pada gambar 2.47.

```
# In[9]:

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

Figure 2.18: Akurasi perhitungan pohon keputusan

```
In [9]: from sklearn.model_selection import cross_val_score
...: scores = cross_val_score(tempe, Cireng_att, Cireng_pass, cv=5)
...: # show average score and +/- two standard deviations away (covering 95% of scores)
...: print("Accuracy: %0.2f (+/- %0.2f)" % (scores.mean(), scores.std() * 2))
Accuracy: 0.71 (+/- 0.06)
```

Figure 2.19: hasil Akurasi perhitungan pohon keputusan

3. Evaluasi dan akurasi adalah bagaimana cara kita bisa mengevaluasi seberapa baik model mengerjakan pekerjaannya dengan cara mengukur akurasi. Akurasi nantinya didefinisikan sebagai presentase kasus yang telah diklasifikasikan dengan benar. Kita dapat melakukan analisis kesalahan yang telah di buat oleh model.

Contoh ilustrasi gambar bisa dilihat pada gambar 2.49.

4. Cara membuat dan membaca confusion matrix yaitu, menentukan pokok permasalahan serta atributnya, membuat Decision Tree, membuat Data Testing, mencari nilai variabelnya misal a,b,c, dan d, mencari nilai recall, precision, accuracy, dan error rate.

Contoh Confusion Matrix.

$$\text{Recall} = 3/(1+3) = 0,75$$

$$\text{Precision} = 3/(1+3) = 0,75$$

$$\text{Accuracy} = (5+3)/(5+1+1+3) = 0,8$$

$$\text{Error Rate} = (1+1)/(5+1+1+3) = 0,2$$

5. Berikut ini tata cara kerja K-fold Cross Validation;

- Total instance akan dibagi menjadi N bagian.
- Fold yang pertama adalah bagian pertama menjadi testing data dan sisanya menjadi training data.



```
# In[10]:

for max_depth in range(1, 20):
    tempeenak = tree.DecisionTreeClassifier(criterion="entropy", max_depth=max_depth)
    scores = cross_val_score(tempeenak, cireng_att, cireng_pass, cv=5)
    print("Max depth: %d, Accuracy: %0.2f (+/- %0.2f)" % (max_depth, scores.mean(), scores.std() * 2))
```

Figure 2.20: Contoh Binary Classification

```
In [10]: for max_depth in range(1, 20):
...:     tempeenak = tree.DecisionTreeClassifier(criterion="entropy", max_depth=max_depth)
...:     scores = cross_val_score(tempeenak, cireng_att, cireng_pass, cv=5)
...:     print("Max depth: %d, Accuracy: %0.2f (+/- %0.2f)" % (max_depth, scores.mean(), scores.std() * 2))
Max depth: 1, Accuracy: 0.64 (+/- 0.05)
Max depth: 2, Accuracy: 0.69 (+/- 0.07)
Max depth: 3, Accuracy: 0.70 (+/- 0.10)
Max depth: 4, Accuracy: 0.70 (+/- 0.07)
Max depth: 5, Accuracy: 0.71 (+/- 0.06)
Max depth: 6, Accuracy: 0.69 (+/- 0.08)
Max depth: 7, Accuracy: 0.68 (+/- 0.04)
Max depth: 8, Accuracy: 0.70 (+/- 0.07)
Max depth: 9, Accuracy: 0.69 (+/- 0.07)
Max depth: 10, Accuracy: 0.68 (+/- 0.09)
Max depth: 11, Accuracy: 0.68 (+/- 0.09)
Max depth: 12, Accuracy: 0.68 (+/- 0.07)
Max depth: 13, Accuracy: 0.67 (+/- 0.06)
```

Figure 2.21: hasil Akurasi perhitungan pohon keputusan

- Hitung akurasi berdasarkan porsi data tersebut dengan menggunakan persamaan.
- Fold yang ke dua adalah bagian ke dua menjadi testing data dan sisanya training data.
- Hitung akurasi berdasarkan porsi data tersebut.
- Lakukan step secara berulang hingga habis mencapai fold ke-K.
- Terakhir hitung rata-rata akurasi K buah.

Contoh ilustrasi gambar bisa dilihat pada gambar 2.50.

6. Decision Tree adalah sebuah metode pembelajaran yang digunakan untuk melakukan klarifikasi dan regresi. Decision Tree digunakan untuk membuat sebuah model yang dapat memprediksi sebuah nilai variabel target dengan cara mempelajari aturan keputusan dari fitur data. Contoh Decision Tree adalah untuk melakukan prediksi apakah Kuda termasuk hewan mamalia atau bukan.

Contoh ilustrasi gambar bisa dilihat pada gambar 2.51.

7. Gain adalah pengurangan yang diharapkan dalam entropy. Dalam machine learning, gain dapat digunakan untuk menentukan sebuah urutan atribut atau

```
# In[11]:

depth_acc = nasipadang.empty((19,3), float)
i = 0
for max_depth in range(1, 20):
    tempemendoan = tree.DecisionTreeClassifier(criterion="entropy", max_depth=max_depth)
    scores = cross_val_score(tempemendoan, cireng_att, cireng_pass, cv=5)
    depth_acc[i,0] = max_depth
    depth_acc[i,1] = scores.mean()
    depth_acc[i,2] = scores.std() * 2
    i += 1

depth_acc
```

Figure 2.22: Contoh Binary Classification

memperkecil atribut yang telah dipilih. Urutan ini akan membentuk decision tree, atribut gain dipilih yang paling besar. Dan Entropi adalah ukuran ketidakpastian sebuah variabel acak sehingga dapat di artikan entropi adalah ukuran ketidakpastian dari sebuah atribut.

Contoh ilustrasi gambar bisa dilihat pada gambar 2.52.

### 2.2.2 Scikit-learn

1. Penjelasan Codingan ini akan menampilkan data pada file yang ditentukan. Untuk codingan ini file yang dieksekusi untuk digunakan ialah student-mat.csv. Secara jelasnya, dalam codingan dapat dilihat bahwa variabel buahpir didefinisikan untuk pembacaan csv dari " buahnaga dimana untuk pemisahannya yaitu separation berupa ; . Setelah itu variabel buahpir di tampilkan dengan perintah menampilkan len panjang ataupun jumlah dan hasilnya berupa angka 395 .

Gambar Screenshootan codingan dan hasil bisa dilihat pada gambar 2.31.

2. Penjelasan codingan ini berfungsi untuk menampilkan baris G1, G2 dan G3 ( berdasarkan kriterianya ) untuk kolom PASS pada variabel buahpir. Untuk lebih jelasnya, pada codingan terdapat pendefinisian pembacaan lamda ( panjang gelombang ) dari baris G1, G2 dan G3. Apabila row-row tersebut bernilai lebih dari 35 maka akan terdefiniskan angka 1 apabila tidak, maka akan terdefiniskan angka 0 pada kolom PASS ( sesuai permintaan awal ). Selanjutnya variabelnya di ditampilkan sehingga menampilkan keluaran. Tidak lupa terdapat juga jumlah dari baris dan kolom yang berubah sesuai dengan baris yang dieksekusi.

```

In [11]: depth_acc = nasipadang.empty((19,3), float)
...: i = 0
...: for max_depth in range(1, 20):
...:     tempemendoan = tree.DecisionTreeClassifier(criterion="entropy", max_depth=max_depth)
...:     scores = cross_val_score(tempemendoan, cireng_att, cireng_pass, cv=5)
...:     depth_acc[i,0] = max_depth
...:     depth_acc[i,1] = scores.mean()
...:     depth_acc[i,2] = scores.std() * 2
...:     i += 1
...:
...:
...: depth_acc
Out[11]:
array([[ 1.      ,  0.63779741,  0.05447395],
       [ 2.      ,  0.68706505,  0.07197636],
       [ 3.      ,  0.69774069,  0.09857052],
       [ 4.      ,  0.70088863,  0.07289402],
       [ 5.      ,  0.71017885,  0.05597996],
       [ 6.      ,  0.68705331,  0.08098517],
       [ 7.      ,  0.68095891,  0.03757078],
       [ 8.      ,  0.69627251,  0.07383519],
       [ 9.      ,  0.68700597,  0.07922729],
       [10.      ,  0.67468617,  0.07965841],
       [11.      ,  0.67927752,  0.10138467],
       [12.      ,  0.69017812,  0.05085986],
       [13.      ,  0.66856774,  0.05944621],
       [14.      ,  0.65625968,  0.05003322],
       [15.      ,  0.66554953,  0.07030686],
       [16.      ,  0.6748403 ,  0.02486122],
       [17.      ,  0.66244894,  0.05728159],
       [18.      ,  0.65016492,  0.04288463],
       [19.      ,  0.66713588,  0.03161508]])

```

Figure 2.23: hasil Akurasi perhitungan pohon keputusan

Gambar Screenshootan codingan dan hasil bisa dilihat pada gambar 2.32.

3. Penjelasan codingan ini mendefinisikan pemanggilan get dummies dari buah-naga dalam variabel buahpir. Di dalam get dummies sendiri akan terdefiniskan variabel buahpir dengan kolom-kolom yang akan dieksekusi seperti school, address dll. Kemudian variabel tersebut diartikan untuk mendapatkan kembalian berupa keluaran dari eksekusi perintah variabel buahpir beserta dengan jumlah baris dan kolom data yang dieksekusi.

Gambar Screenshootan codingan dan hasil bisa dilihat pada gambar 2.33.

4. Penjelasan codingan ini difungsikan untuk mengartikan pembagian data yang berupa training dan testing data. pertama-tama variabel buahpir akan mengartikan sampel yang akan digunakan ( berupa shuffle row ). Nah kemudian masing-masing parameter yaitu buahpir train dan buahpir test akan berjumlah 500 data ( telah dibagi untuk training dan testing ). Selanjutnya dilakukan pengekseskuan untuk kolom Pass, apabila sesuai dengan axis=1 maka eksekusi

```
# In[12]:

import matplotlib.pyplot as bakwan
fig, ax = bakwan.subplots()
ax.errorbar(depth_acc[:,0], depth_acc[:,1], yerr=depth_acc[:,2])
bakwan.show()
```

Figure 2.24: codingan pembuatan grafik

fungsi berhasil. Selain itu juga disertakan jumlah dari peserta yang lolos dari semua nilai data setnya.

Gambar Screenshootan codingan dan hasil bisa dilihat pada gambar 2.34.

5. Penjelasan codingan ini hanya membuktikan pengujian dari Klasifikasi Decision Tree yang ada, apakah true atau tidak dan hasilnya true. Pada codingan ini di definisikan library sklearn untuk mengimpor atau menampilkan tree. Variabel buahapel difungsikan untuk membaca klasifikasi decision tree dari tree itu sendiri dengan 2 parameternya yaitu kriteria=entropy dan max depth=5. Maka selanjutnya variabel buahapel akan masuk dan terbaca dalam module fit dengan 2 parameter yaitu buahpir trai att dan buahpir train pass.

Gambar Screenshootan codingan dan hasil bisa dilihat pada gambar 2.35.

6. Penjelasan codingan ini memberikan gambaran dari klasifikasi decision tree yaitu pengolahan parameter yang dieksekusi kedalam variabel buahapel. Tentunya dengan pemanfaatan library graphviz yang telah diimport dan difungsikan.

Gambar Screenshootan codingan dan hasil bisa dilihat pada gambar 2.36.

7. Penjelasan codingan ini membahas tentang penyimpanan tree dari library graphviz yang dieksekusi bersamaan dengan variabel buahapel dan parameter lainnya. Dilakukan pengecekan dan pengujian apakah klasifikasi decision treenya dapat berjalan atau tidak. Apabila tidak berjalan, maka akan terjadi error, namun codingan ini berfungsi.

Gambar Screenshootan codingan dan hasil bisa dilihat pada gambar 2.37.

8. Penjelasan codingan ini membaca score dari variabel buahapel dimana terdapat 2 parameter yang dihitung dan diuji yaitu buahpir test att dan buahpir test pass. Untuk hasilnya sendiri mengapa berupa angka, dikarenakan pada parameter

```
In [12]: import matplotlib.pyplot as bakwan
...: fig, ax = bakwan.subplots()
...: ax.errorbar(depth_acc[:,0], depth_acc[:,1], yerr=depth_acc[:,2])
...: bakwan.show()
```

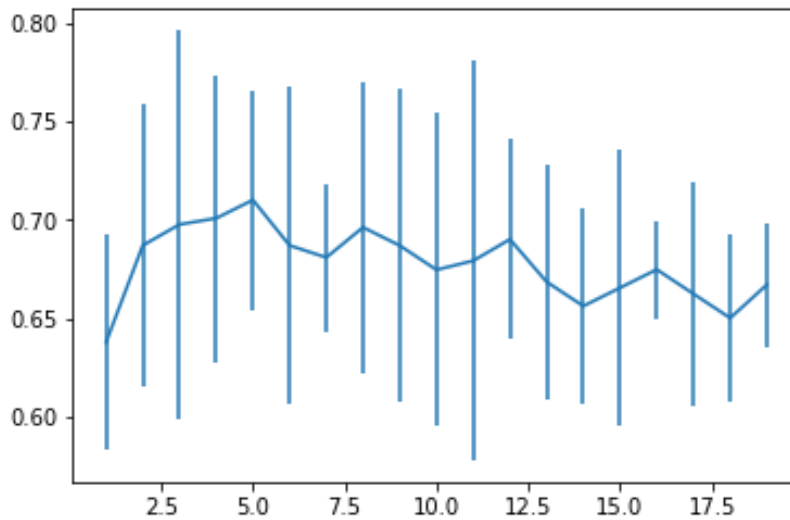


Figure 2.25: hasil grafik

yang dieksekusi memang memiliki data sehingga dieksekusi dan menghasilkan keluaran dari score tersebut.

Gambar Screenshootan codingan dan hasil bisa dilihat pada gambar 2.38.

9. Penjelasan codingan ini membahas mengenai pengkesekusan fungsi dan variabel dari library yang didefinisikan dan yang diimport. Penjelasan lebih jelasnya ialah codingan ini mendefinisikan library sklearn.model.selection kemudian mengimport cross val score. Kemudian variabel score mendefinisikan cross val score yang telah diimport tadi dengan 4 parameter yaitu buahapel, buahpir att, buahpir pass dan cv=5 untuk dieksekusi. Setelah semua pemrosesan tersebut maka hasil yang di tampilkan ialah rata2 perhitungan dari variabel score dimana dan standar dari plus minusnya tentunya dengan ketentuan parameter Accuracy .

Gambar Screenshootan codingan dan hasil bisa dilihat pada gambar 2.39.

10. Penjelasan Codingan ini mendefinisikan max depth dalam jarak angka antara parameter 1 dan 20. Variabel buahapel mendefinisikan klasifikasi decision tree dengan 2 parameter. Kemudian variabel score mengeksekusi parameter lainnya yaitu seperti buahapel, buahpir att, buahpir pass dan cv=5 ) . Hasil yang

```

import graphviz
dot_data = tree.export_graphviz(t, out_file=None, label="all", impurity=False, proportion=True,
                               feature_names=list(d_train_att), class_names=["fail", "pass"],
                               filled=True, rounded=True)
graph = graphviz.Source(dot_data)
graph
Traceback (most recent call last):
  File "<ipython-input-21-461c085f0565>", line 1, in <module>
    import graphviz
ModuleNotFoundError: No module named 'graphviz'

```

Figure 2.26: Screenshot error

ditampilkan ialah dari max depth, accuracy dan plus minusnya dan akhirnya hasil outputannya keluar.

Gambar Screenshootan codingan dan hasil bisa dilihat pada gambar 2.40.

11. Penjelasan codingan ini mengartikan bahwa variabel `depth_acc` akan mengeksekusi `empty` dari importan library `numpy` yang dinamakan `buahpepaya` dengan 2 parameter yaitu `19,3` dan `float`. `i` didefinisikan dengan angka `0` kemudian untuk perhitungan jarak max depth diantara parameter `1` dan `20`. Variabel `buahapel` mengartikan klasifikasi decision tree dengan 2 parameter. setelah itu, variabel `score` mendefinisikan variabel `depth_acc` dengan `i` dan `0`, variabel kedua dari `depth_acc` dengan `i` dan `1` serta variabel ketiga dari `depth_acc` dengan `i` dan `2`, maka pengeksekusian akhir bahwa variabel `i` akan ditambah dengan angka `1` untuk hasil akhirnya. Keluarannya akan berupa array dari perhitungan parameter dan variabel yang telah didefinisikan sebelumnya.

Gambar Screenshootan codingan dan hasil bisa dilihat pada gambar 2.41.

12. Penjelasan codingan ini mendefinisikan pemanggilan dari library `matplotlib.pyplot` sebagai `buahanggur` sehingga nanti hasilnya akan berbentuk gambar grafik/gelombang. Untuk variabel `fig` dan `ax` akan mendefinisikan subplots dari `buahanggur`. Setelah itu ketentuan dari parameter `depth acc = 0`, `depth acc = 1` dan `depth acc 2`. Selanjutnya untuk menampilkan gelombang maka panggil variabel `buahanggur` dengan perintah `show`.

Gambar Screenshootan codingan dan hasil bisa dilihat pada gambar 2.42.

### 2.2.3 Penanganan Error

1. Screenshot Error pada codingan No 8 dapat dilihat pada gambar 2.43.

```
C:\Users\COKRO>conda install graphviz
Collecting package metadata: done
Solving environment: done

## Package Plan ##

  environment location: C:\ProgramData\Anaconda3

added / updated specs:
- graphviz

The following packages will be downloaded:

package | build
-----|-----
graphviz-2.38 | hfa6e2cd_3 37.7 MB
-----|-----
Total: 37.7 MB

The following NEW packages will be INSTALLED:

graphviz pkgs/main/win-32::graphviz-2.38-hfa6e2cd_3

Proceed ([y]/n)? y

Downloading and Extracting Packages
graphviz-2.38 | 37.7 MB | ##### | 100%
Preparing transaction: done
Verifying transaction: failed

EnvironmentNotWritableError: The current user does not have write permissions to the target environment.
  environment location: C:\ProgramData\Anaconda3

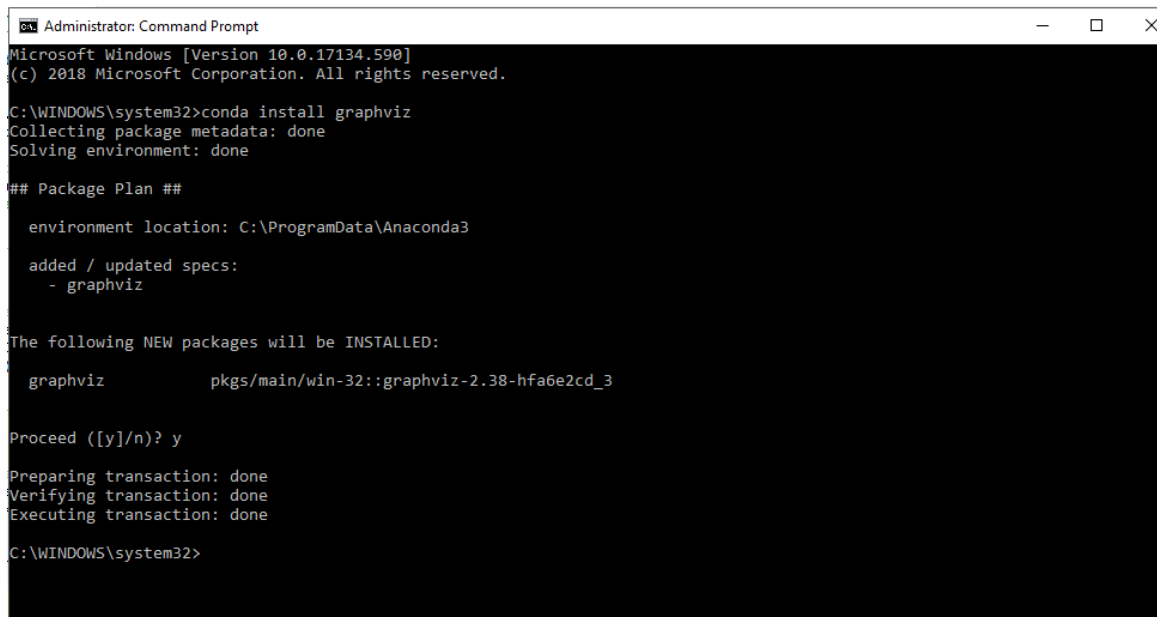
C:\Users\COKRO>
```

Figure 2.27: Proses instalasi graphviz

2. Codingan eror dan jenis erornya : sebenarnya tidak terdapat eror pada codingan ini namun saat pertama kali di run current cell codingan ini akan eror dan tidak keluar outputannya dikarenakan library graphviz sebelumnya tidak ditemukan atau belum di install terlebih dahulu.

```
import graphviz
dot_data = tree.export_graphviz(buahapel, out_file=None, label="all", impurity=
                                feature_names=list(buahpir_train_att), class_
                                filled=True, rounded=True)
graph = graphviz.Source(dot_data)
graph
```

3. Solusi pemecahan masalah eror tersebut yaitu dengan cara menginstall terlebih dahulu library graphviznya pada anaconda prompt atau command prompt anda dengan perintah `conda install graphviz` setelah itu run kembali codingan No 8 maka akan muncul outputan atau tampilan keluarannya.



```
Administrator: Command Prompt
Microsoft Windows [Version 10.0.17134.590]
(c) 2018 Microsoft Corporation. All rights reserved.

C:\WINDOWS\system32>conda install graphviz
Collecting package metadata: done
Solving environment: done

## Package Plan ##

  environment location: C:\ProgramData\Anaconda3

  added / updated specs:
    - graphviz

The following NEW packages will be INSTALLED:

  graphviz          pkgs/main/win-32::graphviz-2.38-hfa6e2cd_3

Proceed ([y]/n)? y
Preparing transaction: done
Verifying transaction: done
Executing transaction: done

C:\WINDOWS\system32>
```

Figure 2.28: Proses instalisasi graphviz di user

Berikut gambar cara menginstall graphviz dapat dilihat pada gambar 2.44

Contoh ilustrasi gambar bisa dilihat pada gambar 2.48.

## 2.3 Fathi Rabbani / 1164074

### 2.3.1 Teori

#### 1. Binary Classification

membuat sebuah klasifikasi dengan menggunakan 2 buah hasil data yang menghasilkan himpunan data dalam dua kelompok yang berbeda. berikut adalah contohnya 2.72.

#### 2. Supervised, Unsupervised and Clustering

- Supervised Learning supervised learning adalah cara untuk mengklasifikasi suatu objek atau data yang telah di tentukan kelasnya, berikut adalah contohnya 2.73.
- Unsupervised Learning unsupervised learning merupakan cara untuk mengklasifikasi tanpa adanya kelas untuk menentukan jenis datanya, berikut ini contohnya 2.74.



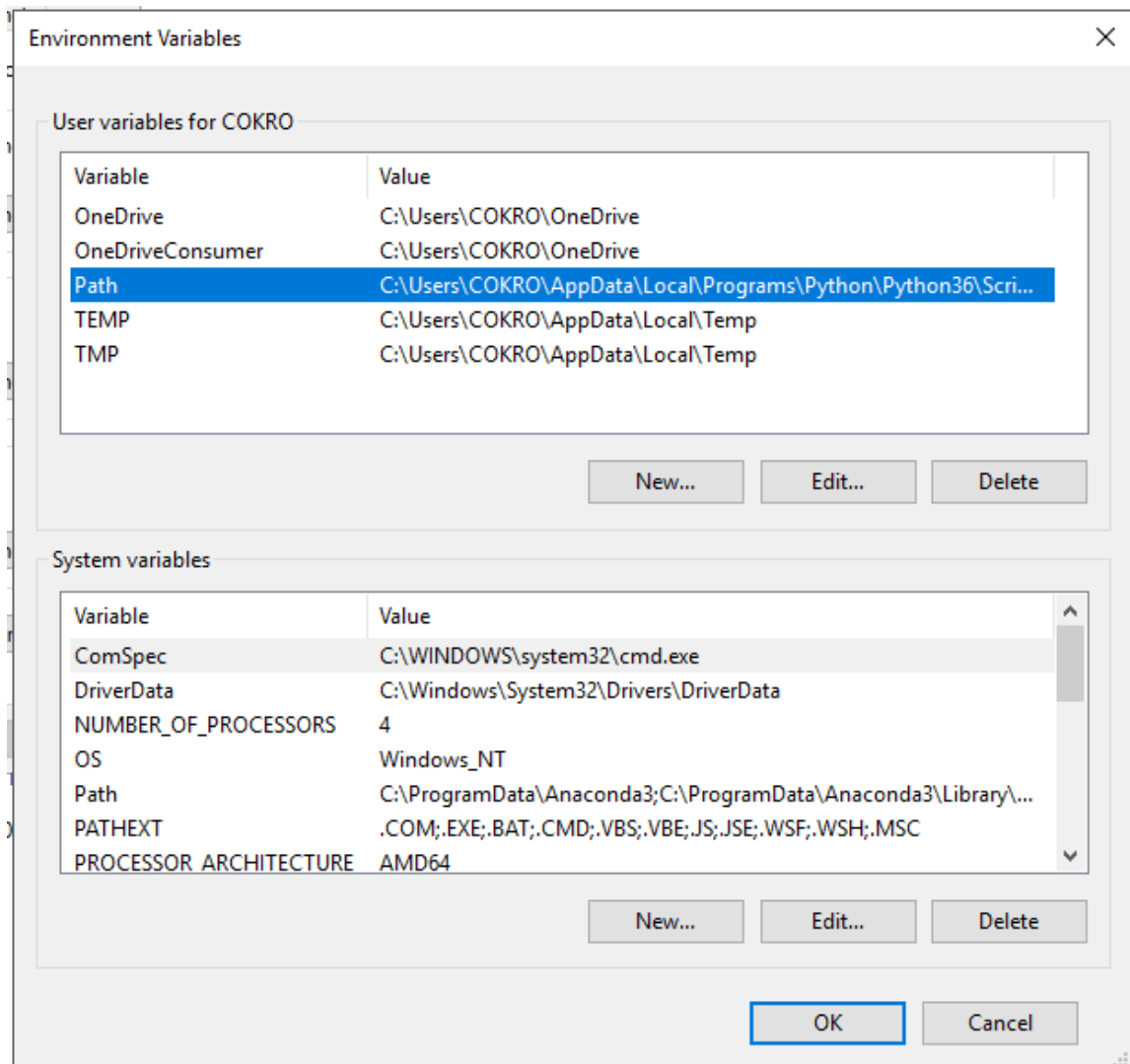


Figure 2.29: Path Komputer

- Clustering clustering merupakan proses mengklasifikasikan yang berdasarkan suatu parameter dalam penentuan hasilnya, berikut contohnya 2.75

### 3. Evaluasi dan Akurasi

- Evaluasi evaluasi adalah sebuah proses dalam mengumpulkan serta mengamati bukti untuk mengukur dampak dari suatu objek, data, program atau proses yang berkaitan itu sendiri, berikut contohnya 2.76.
- Akurasi akurasi merupakan ketepatan dalam sebuah proses dalam melakukan perhitungan akan proses yang sedang berlangsung.

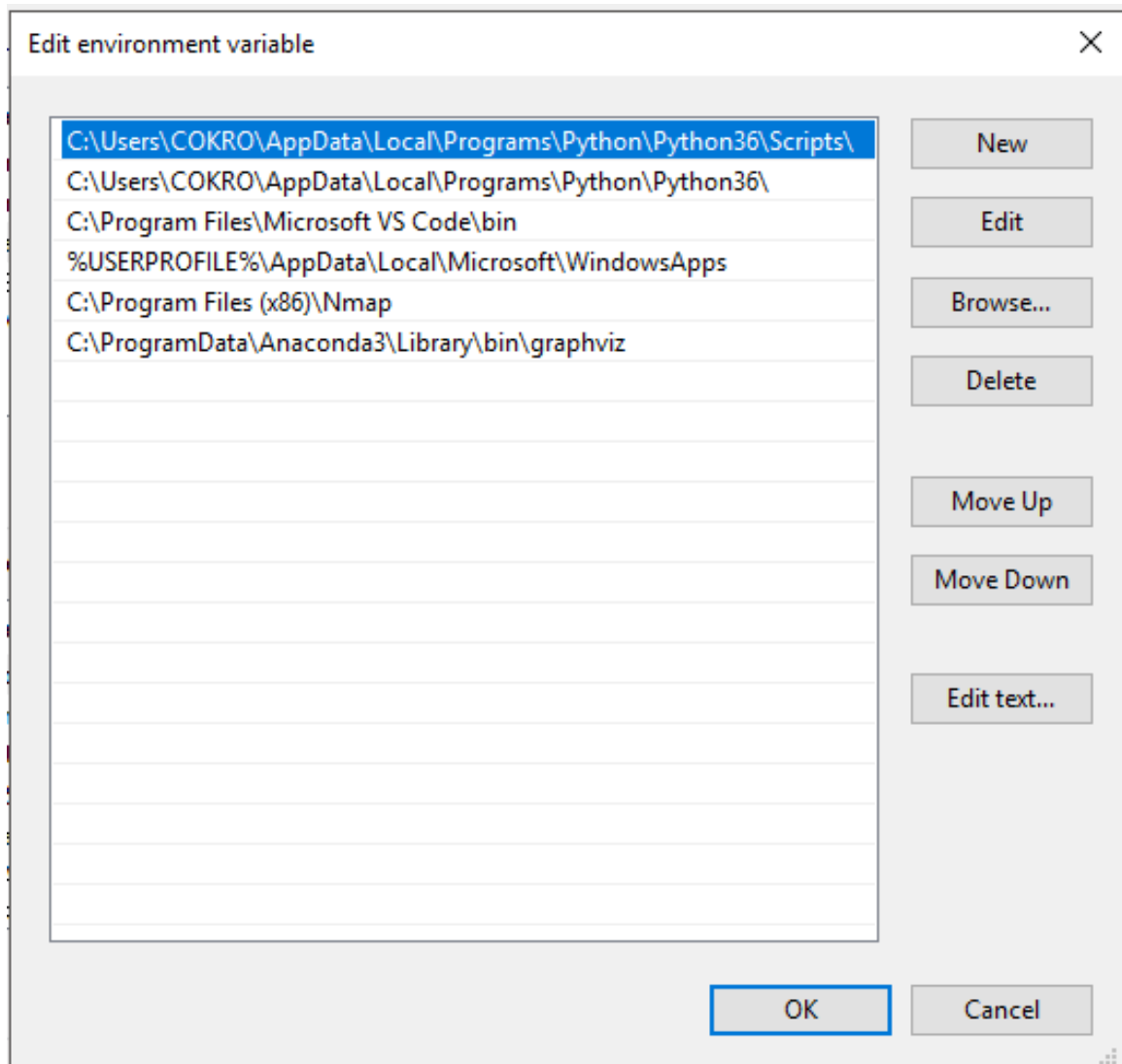


Figure 2.30: Memasukan Direktori Ke Path

4. Membuat dan Membaca Confusion Matrix menentukan objek yang digunakan sebagai bahan uji, sebagai contoh usia 20, 30 dan 40 dengan membuat sebuah tabel yang dapat menampung data dengan nilai 10 pada gambar 2.77 . lalu data tersebut bisa juga berupa data sebagai berikut 2.78. membaca data Matrix tersebut dengan menggunakan Usia sebagai data standarnya dengan rentang usia 20 hingga 40 tahun, lalu jumlah orang yang dapat di ketahui adalah 10 dan data yang ternilai haruslah berisi 10 agar data tersebut valid.
5. K-Fold Cross Validation K-fold Cross Validation merupakan cara untuk melatih suatu mesin dimana di dalamnya terdapat data set yang dibagi menjadi dua yaitu untuk data testing dan data training contoh 100 datasebesar 20 data digu-

```
In [10]: import pandas as buahnaga
...: buahpir = buahnaga.read_csv('E:\DATA KULIAH\SEMESTER 6\KECERDASAN BUATAN(PAK
ROLLY)\Python-Artificial-Intelligence-Projects-for-Beginners\Chapter01\dataset\student-
mat.csv', sep=';')
...: len(buahpir)
Out[10]: 395
```

Figure 2.31: Hasil Codingan No 1.

```
In [11]: buahpir['pass'] = buahpir.apply(lambda row: 1 if (row['G1']+row['G2']+row['G3'])
...: >= 35 else 0, axis=1)
...: buahpir = buahpir.drop(['G1', 'G2', 'G3'], axis=1)
...: buahpir.head()
Out[11]:
```

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

```
[5 rows x 31 columns]
```

Figure 2.32: Hasil Codingan No 2.

nakan untuk data testing kemudian 80 datanya digunakan untuk data training dimana data training tersebut digunakan untuk menentukan nilai bobot yang dimasukan kedalam rumus regresi linier. seperti berikut 2.79.

- Decision Tree merupakan implementasi dari binari clasification dimana akan terdapat akar dan cabang data yang memiliki nilai if...else contoh pada akar data berisi nilai jenis kelamin, apakah pria pada cabang satu bernilai iya dan pada cabang dua bernilai tidak jika nilainya iya berarti jenis kelaminnya pria dan jika tidak maka bernilai wanita, lebih jelasnya dapat dilihat pada gambar2.80.
- Information Gain dan Entropi informasion gain proses dengan mempraktekkan sistem decision tree menggunakan prinsip if...else yang berlangsung hingga menghasilkan data yang diinginkan. untuk contohnya seperti berikut ini sedangkan entropi merupakan ukuran keacakan dari informasi. semakin tinggi entropi maka semakin sulit dalam menentukan keputusan, berikut contohnya 2.81.

## 2.3.2 Praktek

- Code

- import pandas as plum  
durian = plum.read\_csv('dataset/student-mat.csv', sep=';')

```

In [12]: buahpir = buahnaga.get_dummies(buahpir, columns=['sex', 'school', 'address',
...: 'famsize',
...: 'Pstatus', 'Mjob', 'Fjob',
...: 'reason', 'guardian', 'schoolsup',
...: 'famsup', 'paid', 'activities',
...: 'nursery', 'higher', 'internet',
...: 'romantic'])
...: buahpir.head()
Out[12]:
   age  Medu  Fedu  ...  internet_yes  romantic_no  romantic_yes
0   18     4     4  ...             0             1             0
1   17     1     1  ...             1             1             0
2   15     1     1  ...             1             1             0
3   15     4     2  ...             1             0             1
4   16     3     3  ...             0             1             0

[5 rows x 57 columns]

```

Figure 2.33: Hasil Codingan No 3.

```
len(durian)
```

pada code berikut menjelaskan data dari pandas dengan menggunakan nama alias plum yang akan membaca data student-mat.csv yang berada pada folder dataset. hasilnya terdapat pada gambar 2.72

2. 

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

pada code berikut ini menjelaskan bahwa slot data pada student-mat.csv akan ditambah dengan kolom pass dan menggunakan proses lambda yang akan menghasilkan data berupa array dengan struktur G1, G2 dan G3. hasilnya bisa dilihat pada gambar 2.73

3. 

```
durian = plum.get_dummies(durian, columns=['sex', 'school', 'address', 'reason', 'guardian', 'schoolsup', 'famsup', 'nursery', 'higher', 'internet', 'romantic'])
durian.head()
```

pada code berikut menerangkan bahwa variable durian memiliki proses yang akan memanggil variable plum untuk mendapatkan data pada kolom tersebut. hasilnya adalah 2.74

4. 

```
durian = durian.sample(frac=1)
# split training and testing data
d_train = durian[:500]
```

```

In [5]: buahpir = buahpir.sample(frac=1)
...: # split training and testing data
...: buahpir_train = buahpir[:500]
...: buahpir_test = buahpir[500:]
...:
...: buahpir_train_att = buahpir_train.drop(['pass'], axis=1)
...: buahpir_train_pass = buahpir_train['pass']
...:
...: buahpir_test_att = buahpir_test.drop(['pass'], axis=1)
...: buahpir_test_pass = buahpir_test['pass']
...:
...: buahpir_att = buahpir.drop(['pass'], axis=1)
...: buahpir_pass = buahpir['pass']
...:
...: # number of passing students in whole dataset:
...: import numpy as buahpepaya
...: print("Passing: %d out of %d (%.2f%%)" % (buahpepaya.sum(buahpir_pass),
len(buahpir_pass), 100*float(buahpepaya.sum(buahpir_pass)) / len(buahpir_pass)))
Passing: 328 out of 649 (50.54%)

```

Figure 2.34: Hasil Codingan No 4.

```

In [6]: from sklearn import tree
...: buahapel = tree.DecisionTreeClassifier(criterion="entropy", max_depth=5)
...: buahapel = buahapel.fit(buahpir_train_att, buahpir_train_pass)

```

Figure 2.35: Hasil Codingan No 5.

```

d_test = durian[500:]

d_train_att = d_train.drop(['pass'], axis=1)
d_train_pass = d_train['pass']

d_test_att = d_test.drop(['pass'], axis=1)
d_test_pass = d_test['pass']

d_att = durian.drop(['pass'], axis=1)
d_pass = durian['pass']

# number of passing students in whole dataset:
import numpy as nanas
print("Passing: %d out of %d (%.2f%%)" % (nanas.sum(d_pass), len(d_pass))

```

code berikut menjelaskan bahwa data durian akan diproses untuk di-

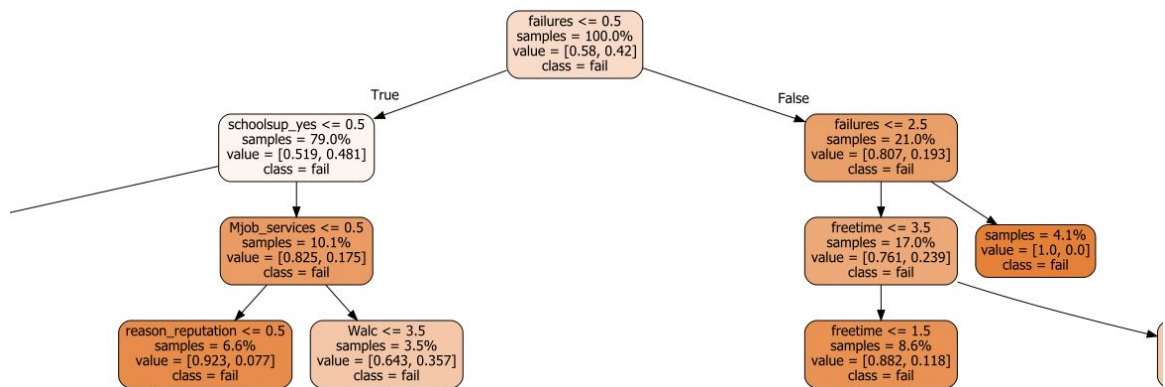


Figure 2.36: Hasil Codingan No 6.

```
In [7]: tree.export_graphviz(buahapel, out_file="student-performance.dot",
label="all", impurity=False, proportion=True,
...:                        feature_names=list(buahpir_train_att),
class_names=["fail", "pass"],
...:                        filled=True, rounded=True)
```

Figure 2.37: Hasil Codingan No 7.

dapatkan hasil dari penggunaan k-fold cross yang membagi data dengan training dan testing dengan hasilnya adalah seperti pada gambar 2.75

```
5. from sklearn import tree
tomat = tree.DecisionTreeClassifier(criterion="entropy", max_depth=5)
tomat = tomat.fit(d_train_att, d_train_pass)
```

code ini mengambil data dari sklearn berupa data tree dengan variabel tomat yang menampung data proses penggunaan decisiontree dan di proses dengan menggunakan data d\_train\_att dan d\_train\_pass hasilnya ada digambar 2.76

```
6. import graphviz
dot_data = tree.export_graphviz(tomat, out_file=None, label="all", impurity=False,
                                feature_names=list(d_train_att), class_names=["fail", "pass"],
                                filled=True, rounded=True)

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

pada code berikut menjelaskan data graphviz yang diambil untuk menampilkan data 2.77

```
In [8]: buahapel.score(buahpir_test_att, buahpir_test_pass)
Out[8]: 0.6644295302013423
```

Figure 2.38: Hasil Codingan No 8.

```
In [8]: from sklearn.model_selection import cross_val_score
...: scores = cross_val_score(buahapel, buahpir_att, buahpir_pass, cv=5)
...: # show average score and +/- two standard deviations away (covering 95% of
...: scores)
...: print("Accuracy: %0.2f (+/- %0.2f)" % (scores.mean(), scores.std() * 2))
Accuracy: 0.68 (+/- 0.07)
```

Figure 2.39: Hasil Codingan No 9.

```
7. tree.export_graphviz(tomat, out_file="student-performance.dot", label="a
    feature_names=list(d_train_att), class_names=["fail
    filled=True, rounded=True)
```

pada code ini digunakan untuk memproses data pada code 6 yang akan menghasilkan sebuah file bernama student-performance.dot hasilnya ada pada gambar 2.78

```
8. tomat.score(d_test_att, d_test_pass)
```

pada code ini dihasilkan data seperti berikut 2.79 yang artinya adalah data score dari d\_test\_att dan d\_test\_pass

```
9. from sklearn.model_selection import cross_val_score
    apel = cross_val_score(tomat, d_att, d_pass, cv=5)

    print("Accuracy: %0.2f (+/- %0.2f)" % (apel.mean(), apel.std() * 2))
```

mengambil data dari sklearn.model\_selection dan mengambil data cross\_val\_score yang digunakan untuk menghitung data akurasi dari code 5 hasilnya ada digambar 2.80

```
10. for max_depth in range(1, 20):
    tomat = tree.DecisionTreeClassifier(criterion="entropy", max_depth=max_depth)
    apel = cross_val_score(tomat, d_att, d_pass, cv=5)
    print("Max depth: %d, Accuracy: %0.2f (+/- %0.2f)" % (max_depth, apel.mean(), apel.std() * 2))
```



```

In [10]: for max_depth in range(1, 20):
...:     buahapel = tree.DecisionTreeClassifier(criterion="entropy", max_depth=max_depth)
...:     scores = cross_val_score(buahapel, buahpir_att, buahpir_pass, cv=5)
...:     print("Max depth: %d, Accuracy: %0.2f (+/- %0.2f)" % (max_depth, scores.mean(),
scores.std() * 2))
Max depth: 1, Accuracy: 0.64 (+/- 0.02)
Max depth: 2, Accuracy: 0.69 (+/- 0.02)
Max depth: 3, Accuracy: 0.69 (+/- 0.07)
Max depth: 4, Accuracy: 0.69 (+/- 0.05)
Max depth: 5, Accuracy: 0.67 (+/- 0.03)
Max depth: 6, Accuracy: 0.66 (+/- 0.08)
Max depth: 7, Accuracy: 0.67 (+/- 0.07)
Max depth: 8, Accuracy: 0.67 (+/- 0.05)
Max depth: 9, Accuracy: 0.65 (+/- 0.04)
Max depth: 10, Accuracy: 0.66 (+/- 0.05)
Max depth: 11, Accuracy: 0.66 (+/- 0.07)
Max depth: 12, Accuracy: 0.62 (+/- 0.09)
Max depth: 13, Accuracy: 0.64 (+/- 0.08)
Max depth: 14, Accuracy: 0.63 (+/- 0.09)
Max depth: 15, Accuracy: 0.64 (+/- 0.07)
Max depth: 16, Accuracy: 0.63 (+/- 0.07)
Max depth: 17, Accuracy: 0.62 (+/- 0.09)
Max depth: 18, Accuracy: 0.63 (+/- 0.08)
Max depth: 19, Accuracy: 0.63 (+/- 0.09)

```

Figure 2.40: Hasil Codingan No 10.

code ini menjelaskan hasil dari akurasi pada code 9 yang dibreakdown untuk tampil sebagai data yang terhitung seperti pada gambar 2.81

```

11. semangka = nanas.empty((19,3), float)
    i = 0
    for max_depth in range(1, 20):
        tomat = tree.DecisionTreeClassifier(criterion="entropy", max_depth=max_depth)
        apel = cross_val_score(tomat, d_att, d_pass, cv=5)
        semangka[i,0] = max_depth
        semangka[i,1] = apel.mean()
        semangka[i,2] = apel.std() * 2
        i += 1

```

semangka

pada code ini data pada code 10 di ubah menjadi seurutan array yang ada pada gambar 2.82

```

12. import matplotlib.pyplot as melon
    fig, ax = melon.subplots()
    ax.errorbar(semangka[:,0], semangka[:,1], yerr=semangka[:,2])
    melon.show()

```



```

In [12]: depth_acc = buahpepaya.empty((19,3), float)
...: i = 0
...: for max_depth in range(1, 20):
...:     buahapel = tree.DecisionTreeClassifier(criterion="entropy", max_depth=max_depth)
...:     scores = cross_val_score(buahapel, buahpir_att, buahpir_pass, cv=5)
...:     depth_acc[i,0] = max_depth
...:     depth_acc[i,1] = scores.mean()
...:     depth_acc[i,2] = scores.std() * 2
...:     i += 1
...:
...:
...:
...: depth_acc
Out[12]:
array([[ 1.,      0.63795172,  0.02257095],
       [ 2.,      0.68720762,  0.02295034],
       [ 3.,      0.69178704,  0.06621384],
       [ 4.,      0.69181089,  0.05105217],
       [ 5.,      0.67026014,  0.02969374],
       [ 6.,      0.66411731,  0.06858724],
       [ 7.,      0.68263885,  0.06498788],
       [ 8.,      0.67186961,  0.04285183],
       [ 9.,      0.65182173,  0.03602718],
       [10.,      0.65173861,  0.04048406],
       [11.,      0.65169145,  0.07924518],
       [12.,      0.63476783,  0.04265046],
       [13.,      0.62398685,  0.06238911],

```

Figure 2.41: Hasil Codingan No 11.

code ini menampilkan data grafik yang digunakan untuk melihat data pada code sebelumnya hasilnya adalah seperti pada gambar 2.83

- Hasil

### 2.3.3 Penanganan Error

- Error Path

cara membenarkan error yang terdapat pada gambar 2.84 adalah dengan menginstall ulang graphviz dengan format

```
conda install graphviz
```

atau

```
pip install graphviz
```

yang terdapat pada gambar 2.85

```
In [13]: import matplotlib.pyplot as buahanggur
...: fig, ax = buahanggur.subplots()
...: ax.errorbar(depth_acc[:,0], depth_acc[:,1], yerr=depth_acc[:,2])
...: buahanggur.show()
```

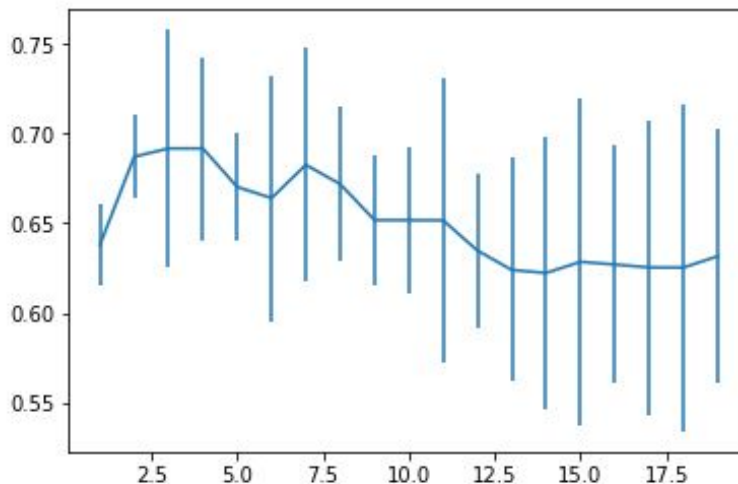


Figure 2.42: Hasil Codingan No 12.

```
In [25]: import graphviz
...: dot_data = tree.export_graphviz(buahpepaya, out_file=None, label="all",
...: impurity=False, proportion=True,
...: feature_names=list(buahpir_train_att),
...: class_names=["fail", "pass"],
...: filled=True, rounded=True)
...: graph = graphviz.Source(dot_data)
...: graph
Traceback (most recent call last):
  File "F:\anaconda\lib\site-packages\IPython\core\formatters.py", line 345, in __call__
    return method()
  File "F:\anaconda\lib\site-packages\graphviz\files.py", line 106, in _repr_svg_
    return self.pipe(format='svg').decode(self._encoding)
  File "F:\anaconda\lib\site-packages\graphviz\files.py", line 128, in pipe
    out = backend.pipe(self._engine, format, data, renderer, formatter)
  File "F:\anaconda\lib\site-packages\graphviz\backend.py", line 206, in pipe
    out, _ = run(cmd, input=data, capture_output=True, check=True, quiet=quiet)
  File "F:\anaconda\lib\site-packages\graphviz\backend.py", line 150, in run
    raise ExecutableNotFound(cmd)
ExecutableNotFound: failed to execute ['dot', '-Tsvg'], make sure the Graphviz executables are on your systems' PATH
```

Figure 2.43: Hasil Gambar Error No 6.

```

C:\Users\HUAO>conda install graphviz
Collecting package metadata: done
Solving environment: done

## Package Plan ##

  environment location: F:\anaconda

added / updated specs:
- graphviz

The following packages will be downloaded:

package | build
-----|-----
graphviz-2.38 | hfa6e2cd_3 37.7 MB
-----|-----
Total: 37.7 MB

The following NEW packages will be INSTALLED:

graphviz pkgs/main/win-32::graphviz-2.38-hfa6e2cd_3

Proceed ([y]/n)?

Downloading and Extracting Packages
graphviz-2.38 | 37.7 MB | ##### | 100%
Preparing transaction: done
Verifying transaction: done
Executing transaction: done

```

Figure 2.44: Hasil Gambar Penanganan Error No 6.

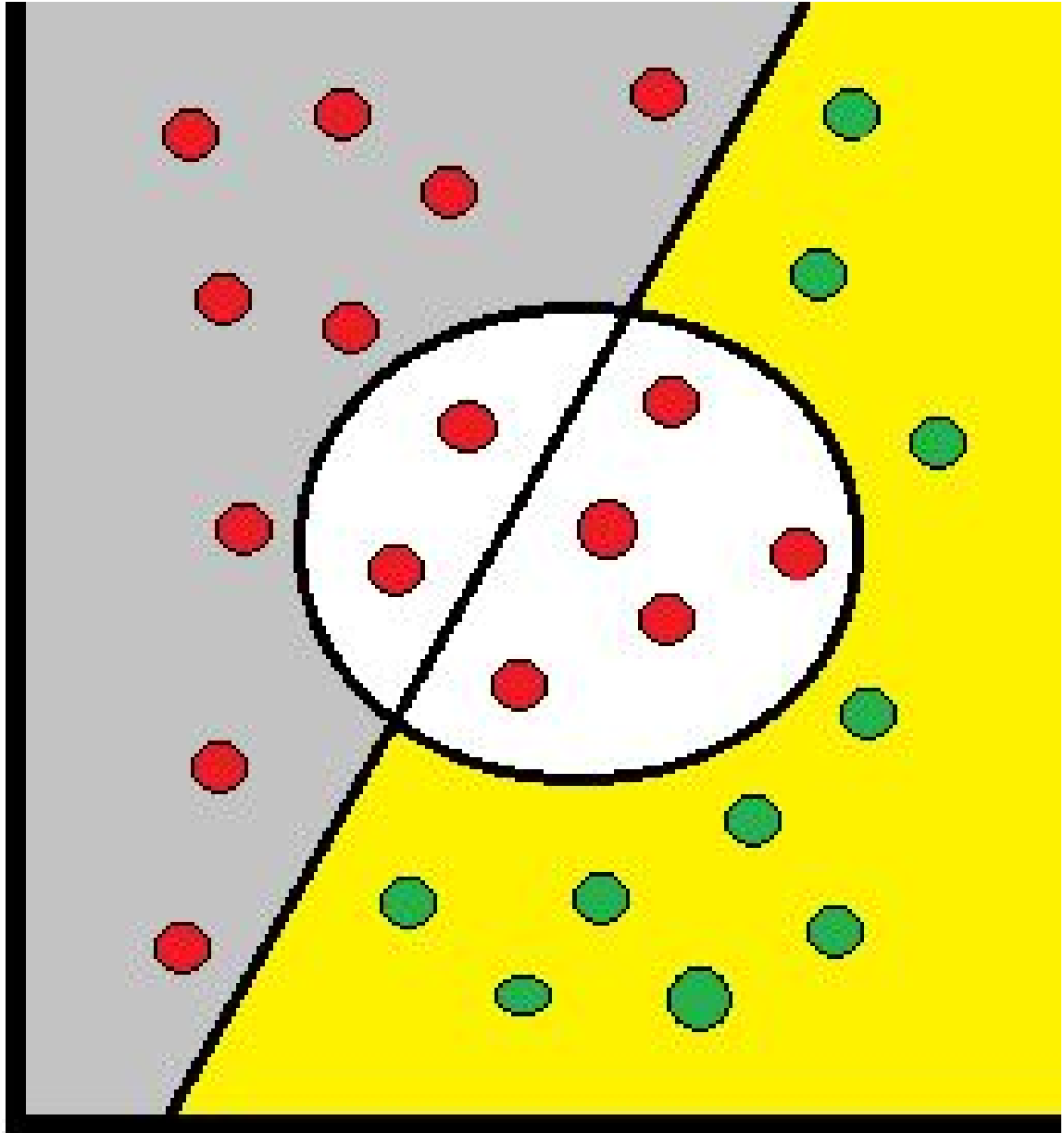


Figure 2.45: Binary Classification.

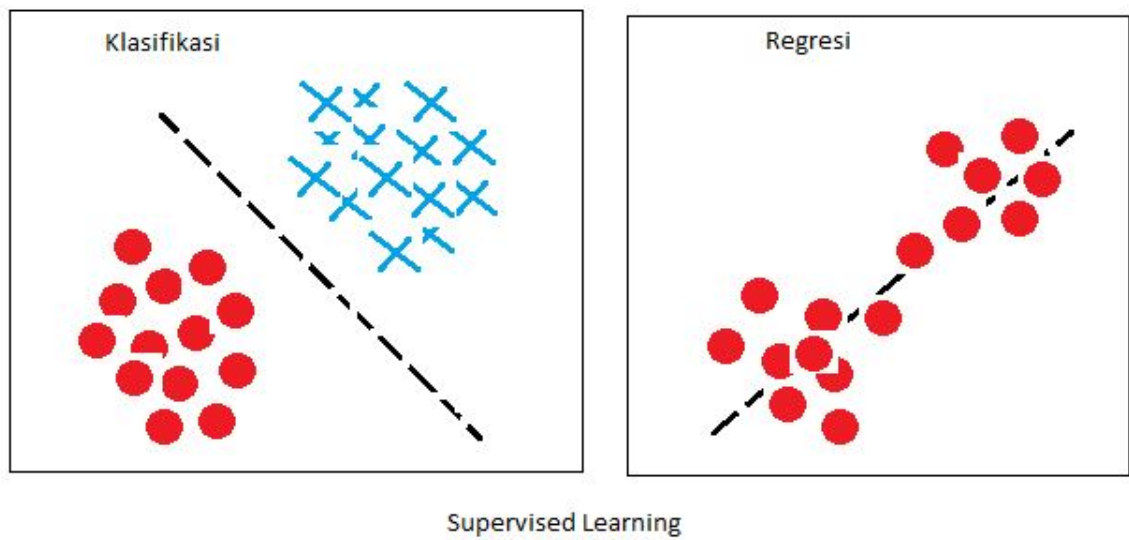


Figure 2.46: Supervised Learning.

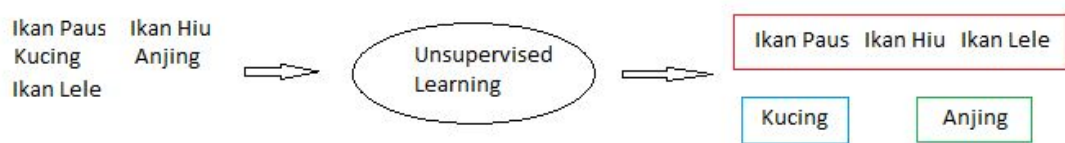


Figure 2.47: Unsupervised Learning.

## Clustering

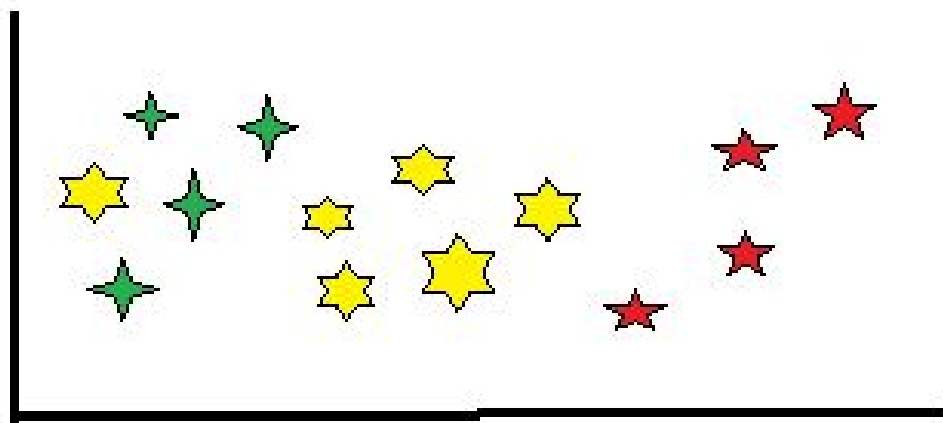


Figure 2.48: Clustering.

|             | Diprediksi Pena | Diprediksi Pensil |
|-------------|-----------------|-------------------|
| True Pena   | 10              | 5                 |
| True Pensil | 7               | 8                 |

Figure 2.49: Evaluasi dan Akurasi.

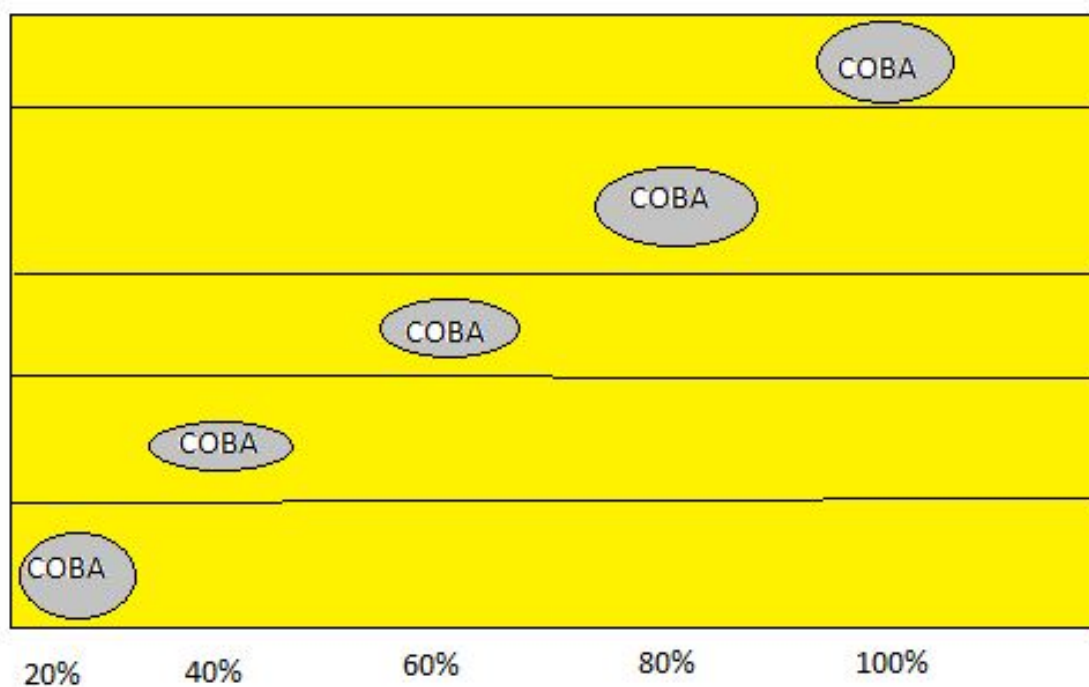


Figure 2.50: K-fold Cross Validation.

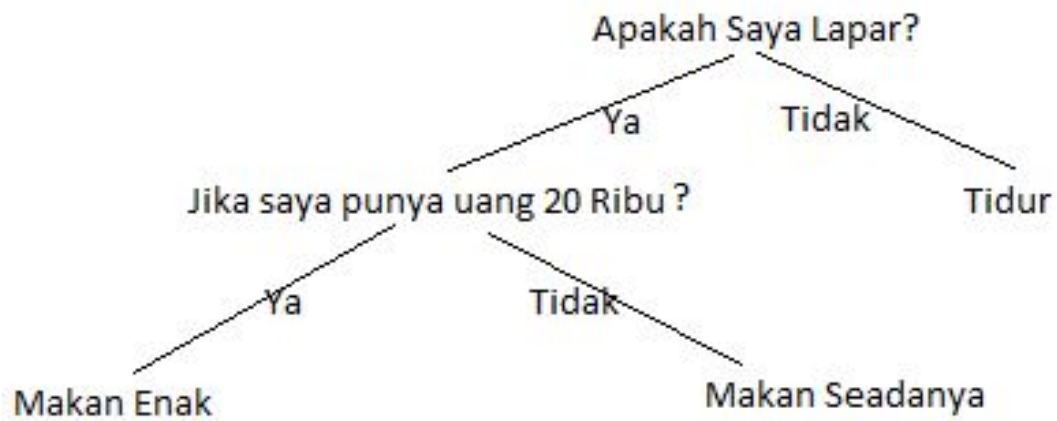


Figure 2.51: Decision Tree.

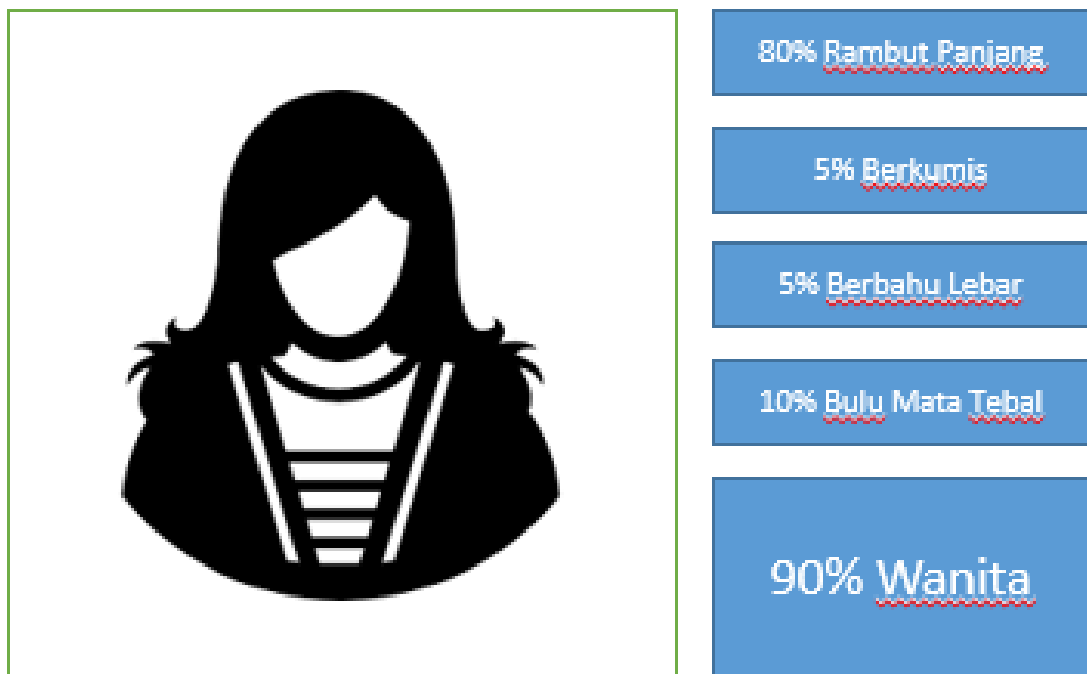


Figure 2.52: Gain.

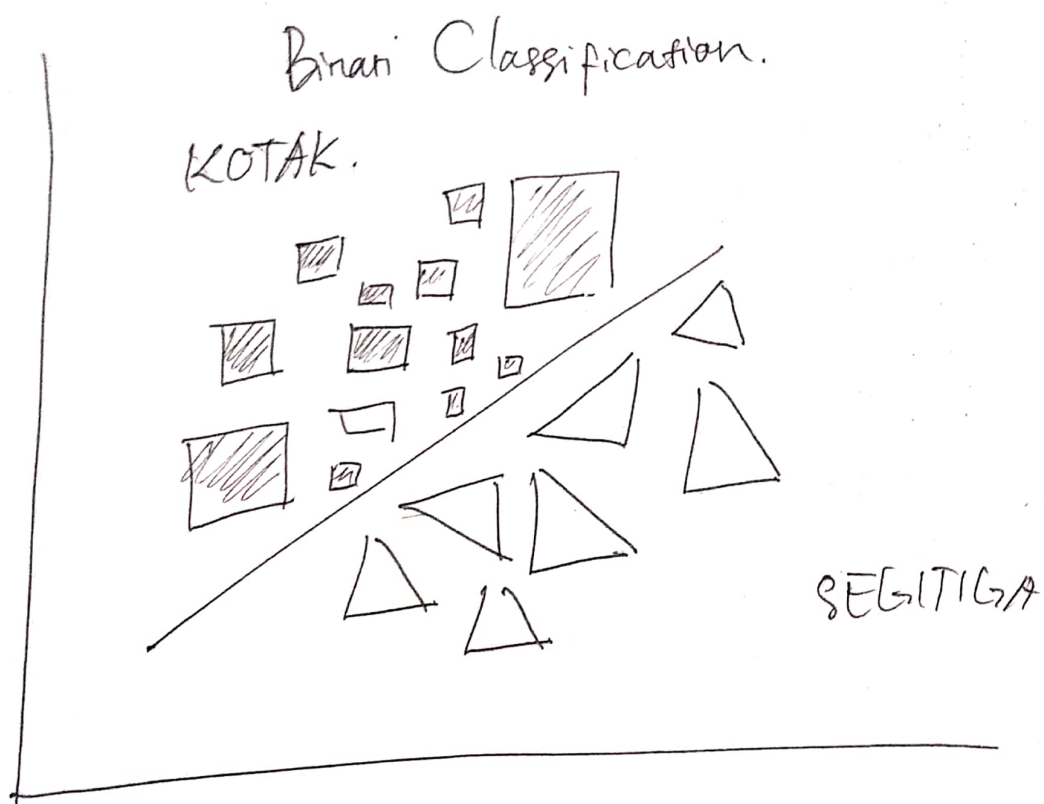


Figure 2.53: Contoh Binary Classification



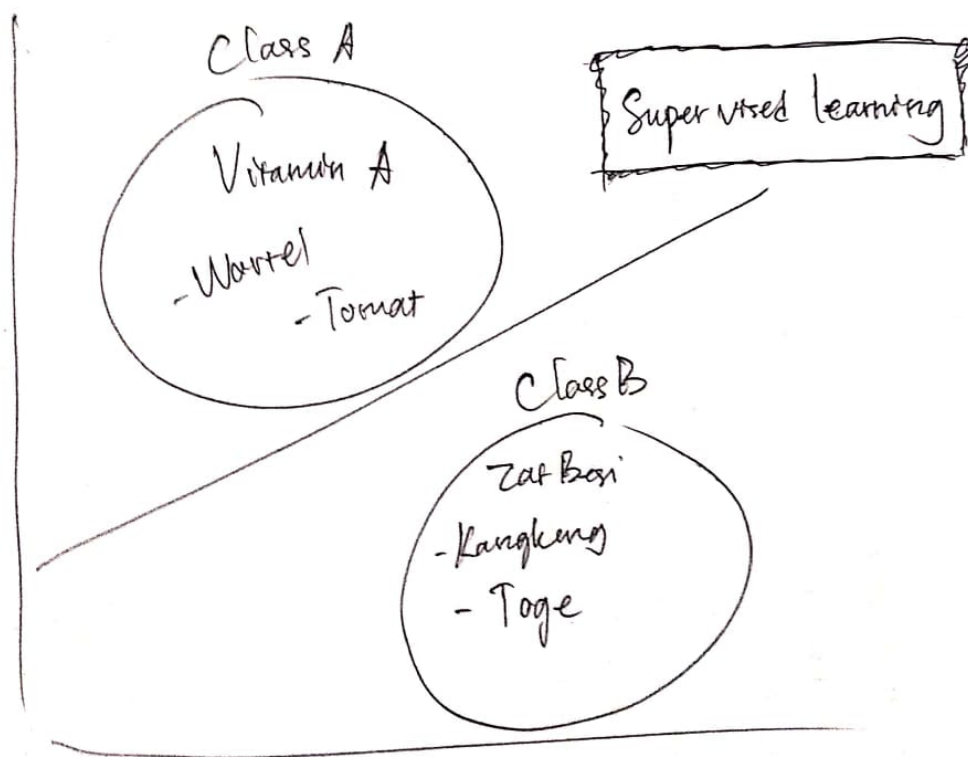


Figure 2.54: Ilustrasi Suvervised Learning

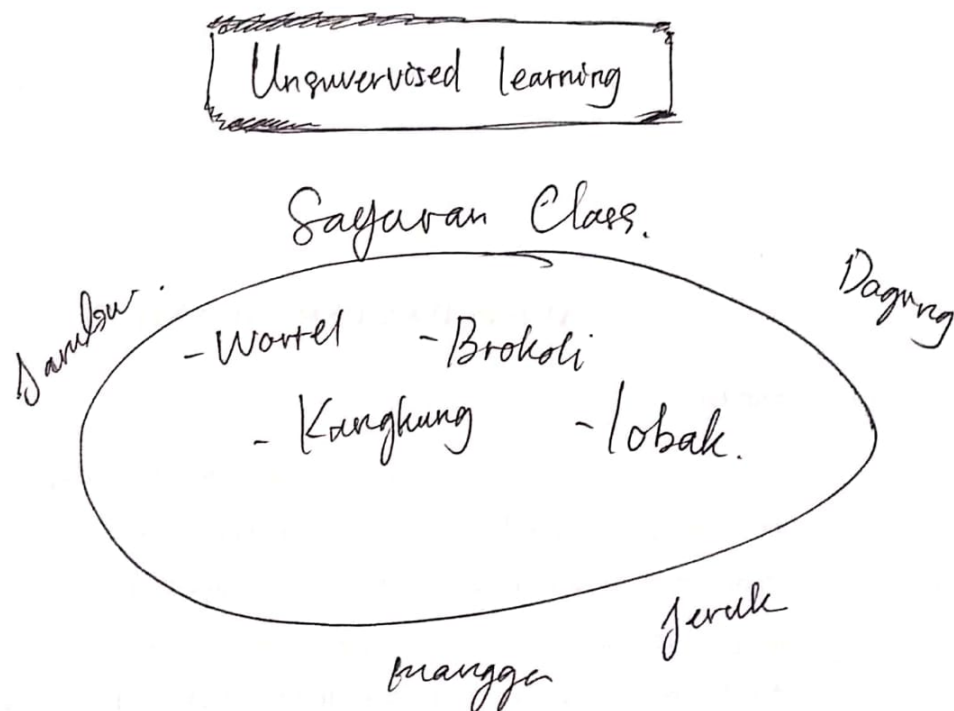


Figure 2.55: Ilustrasi Unsupervised Learning

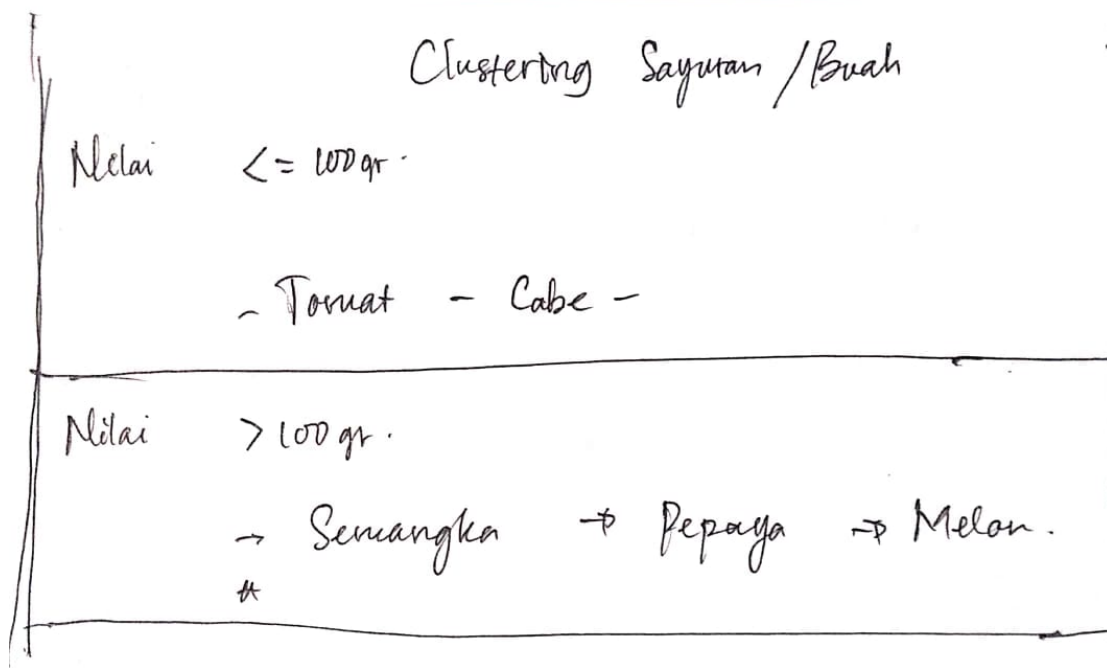


Figure 2.56: Ilustrasi Clustering

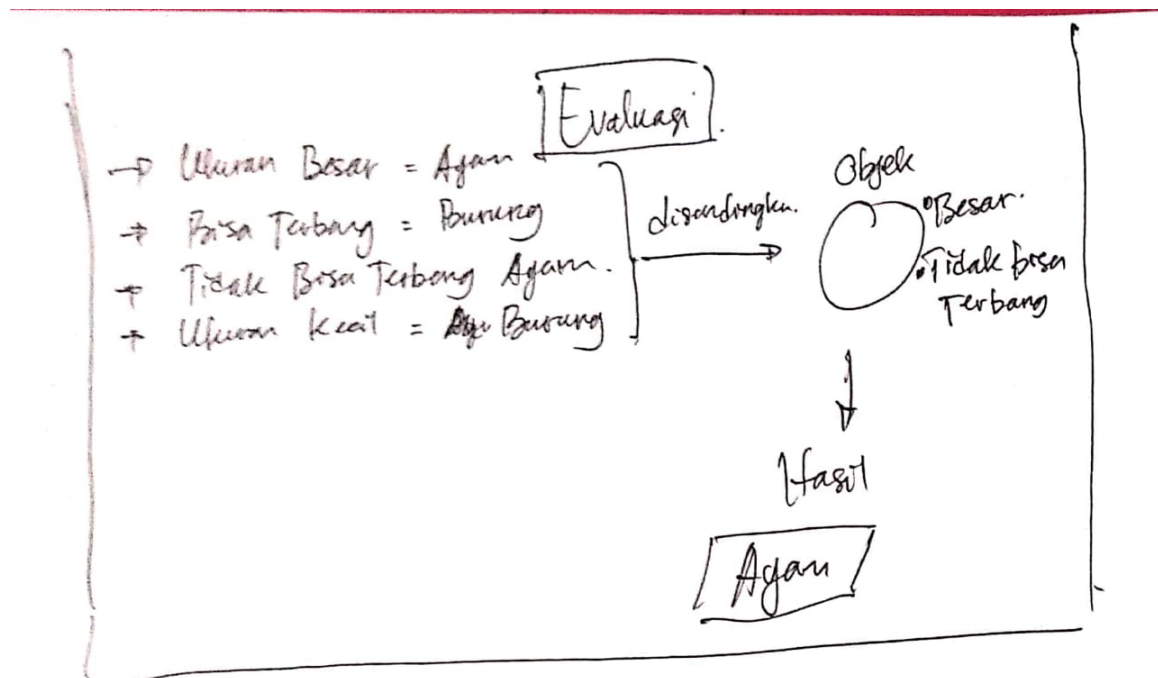


Figure 2.57: Ilustrasi Evaluasi

## Confusion Matrix.

|          |              |       |          |
|----------|--------------|-------|----------|
| Kenanga. |              |       | 30       |
| Mawar    |              | 30    |          |
| Melati   | 30           |       |          |
|          | Bunga Melati | Mawar | Kenanga. |

## Confusion Matrix.

|          |        |       |          |
|----------|--------|-------|----------|
| Kenanga. | 2      | 2     | 26       |
| Mawar    | 2      | 27    | 1        |
| Melati   | 25     | 3     | 2        |
|          | Melati | Mawar | Kenanga. |

Figure 2.58: Ilustrasi Confusion Matrix

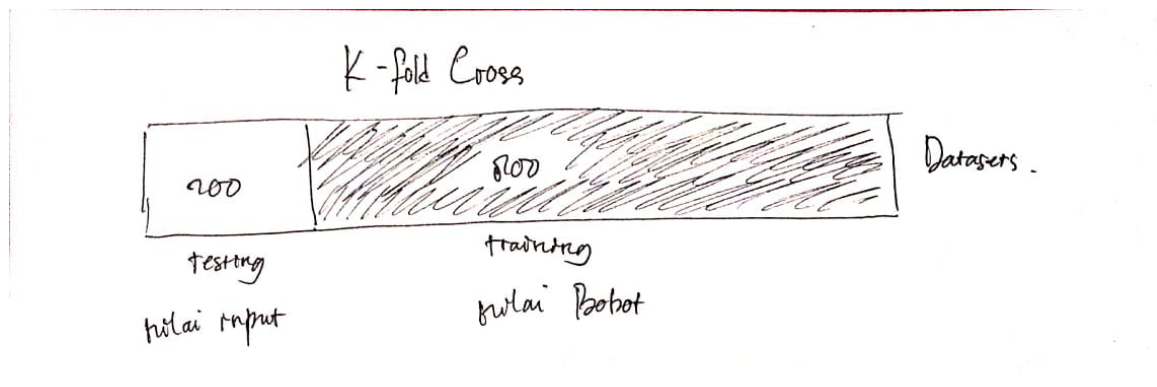


Figure 2.59: Ilustrasi K-Fold

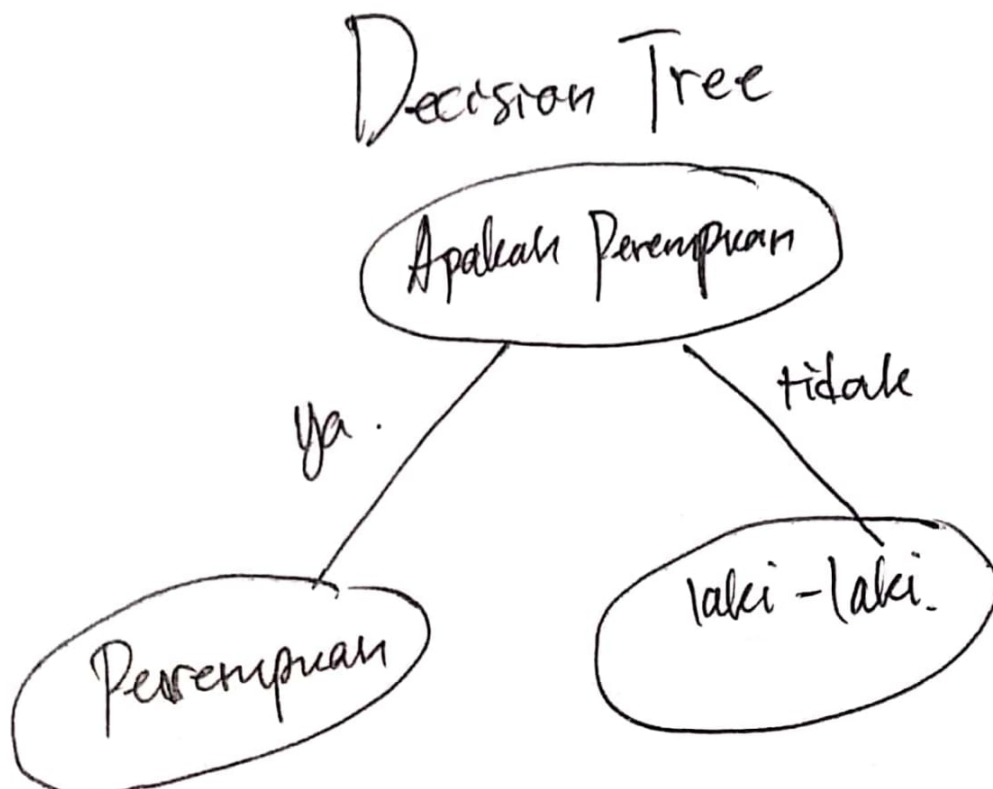


Figure 2.60: Ilustrasi Decision Tree

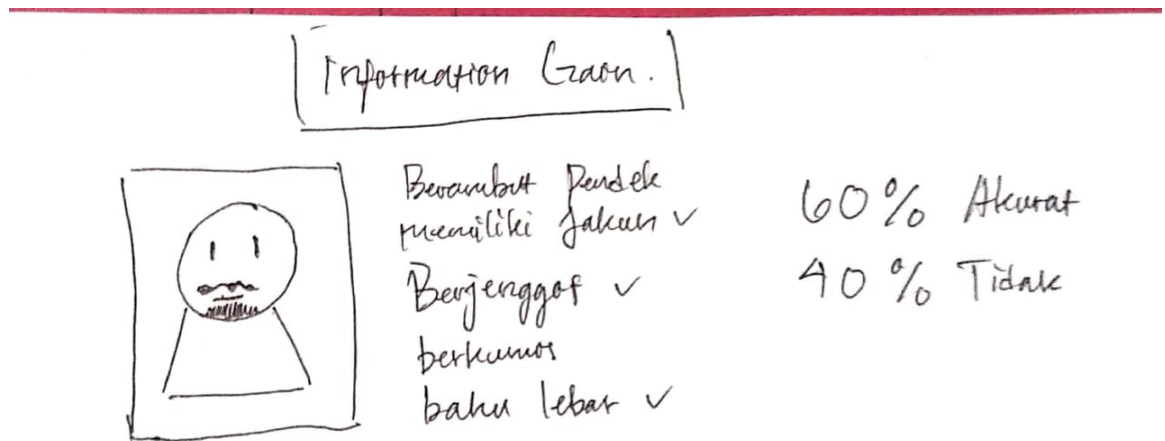


Figure 2.61: Contoh Ilustrasi Information Gain.

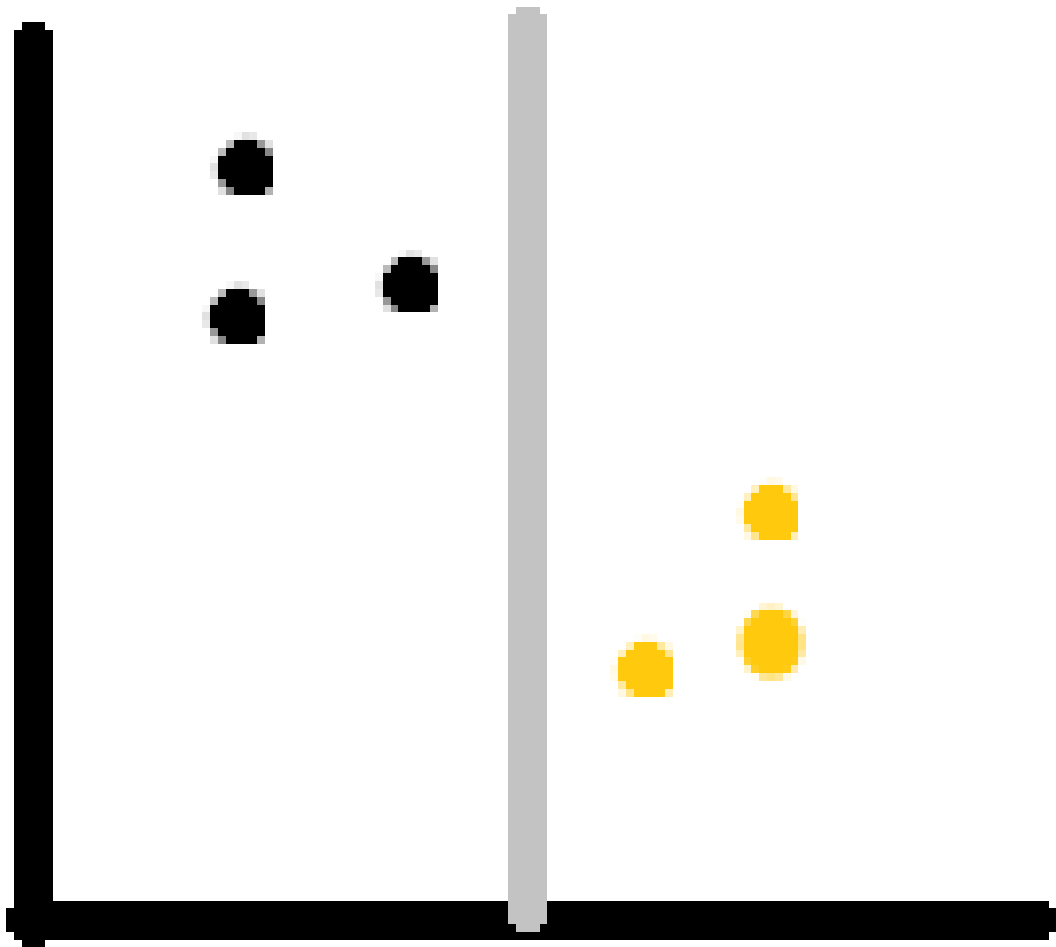


Figure 2.62: Contoh Penggunaan Binary Classification

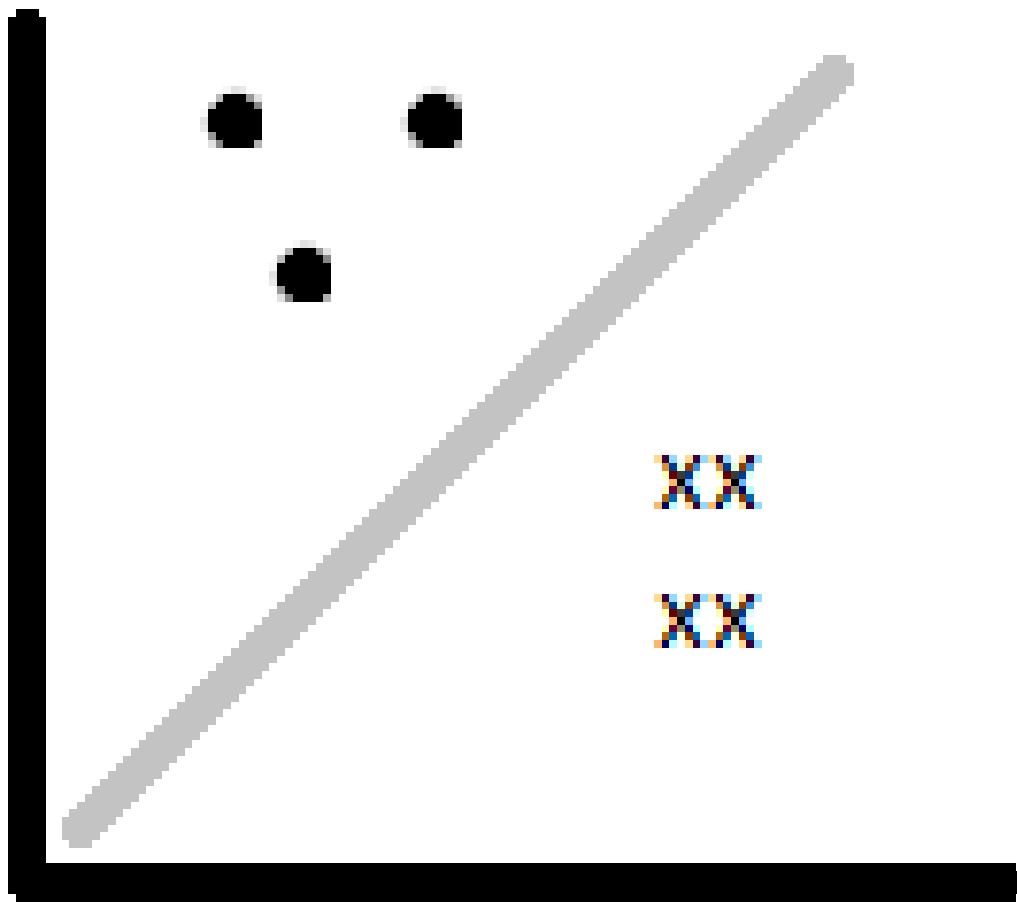
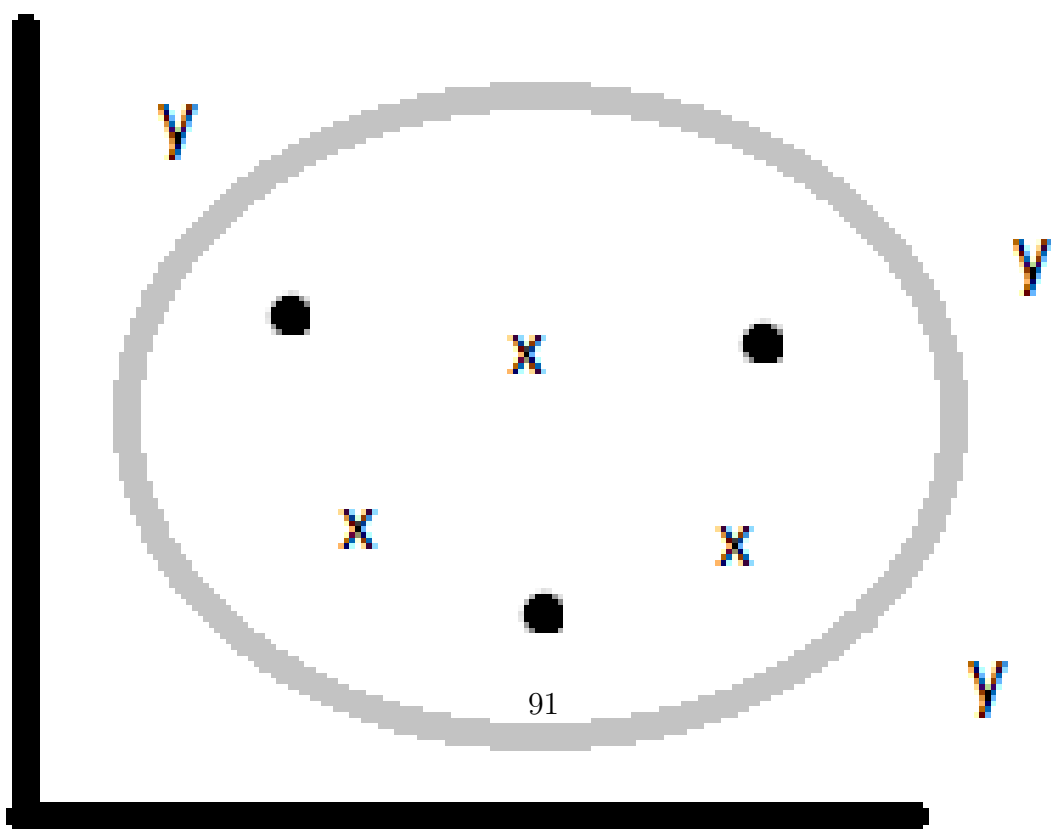


Figure 2.63: Contoh Penggunaan Supervised Learning



## EVALUASI

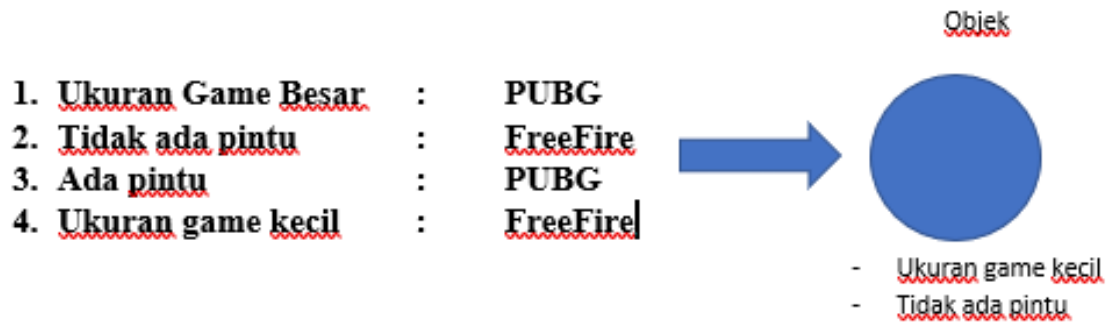


Figure 2.66: Contoh Penggunaan Evaluasi dan Akurasi

|      |    |    |    |
|------|----|----|----|
| 40   |    |    | 10 |
| 30   |    | 10 |    |
| 20   | 10 |    |    |
| usia | 20 | 30 | 40 |

Figure 2.67: Contoh Matrix Confusion

|      |    |    |    |
|------|----|----|----|
| 40   | 2  | 1  | 7  |
| 30   | 1  | 9  | 0  |
| 20   | 8  | 1  | 1  |
| usia | 20 | 30 | 40 |

Figure 2.68: Contoh Matrix Confusion



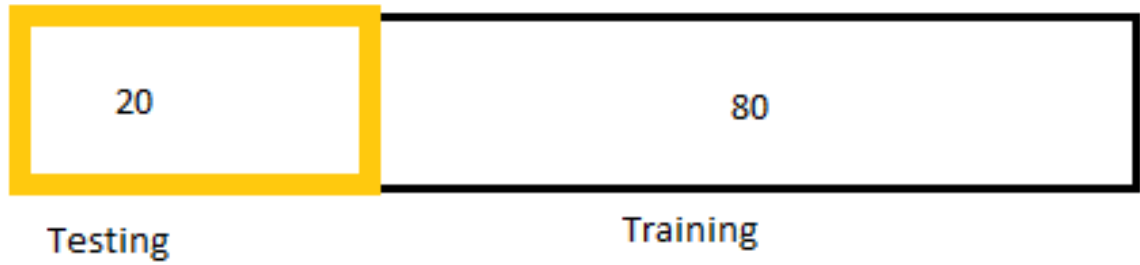


Figure 2.69: Contoh Penggunaan K Fold Cross Validation

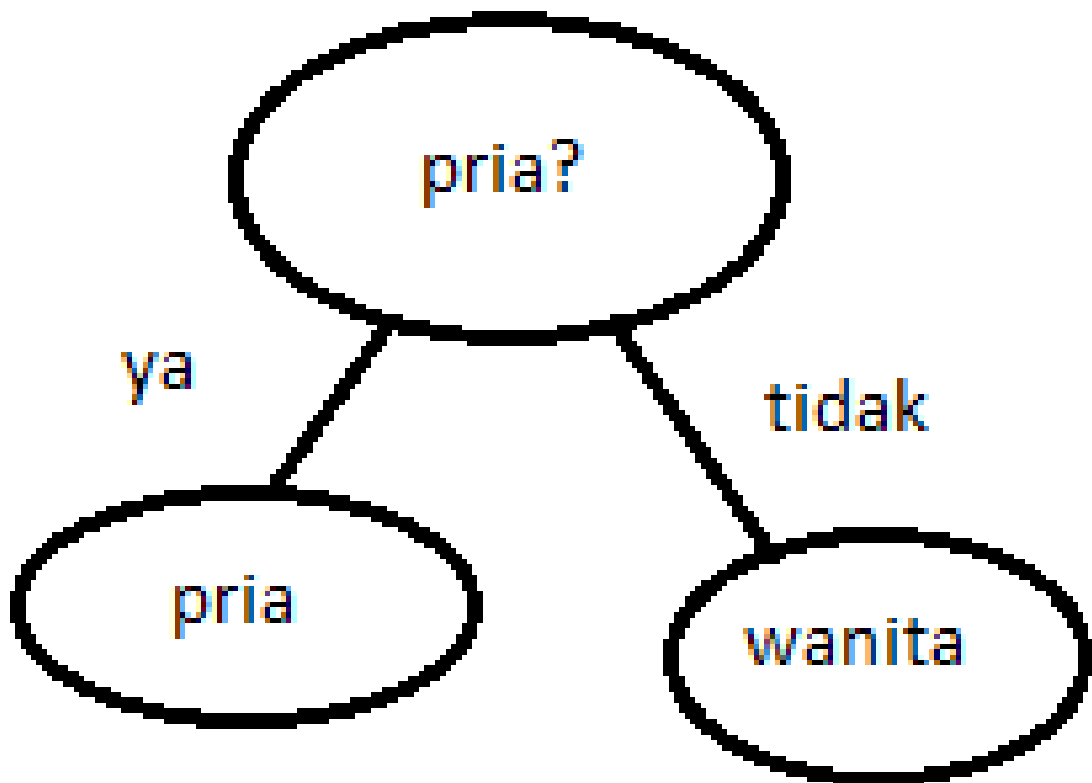


Figure 2.70: Contoh Penggunaan Decision Tree

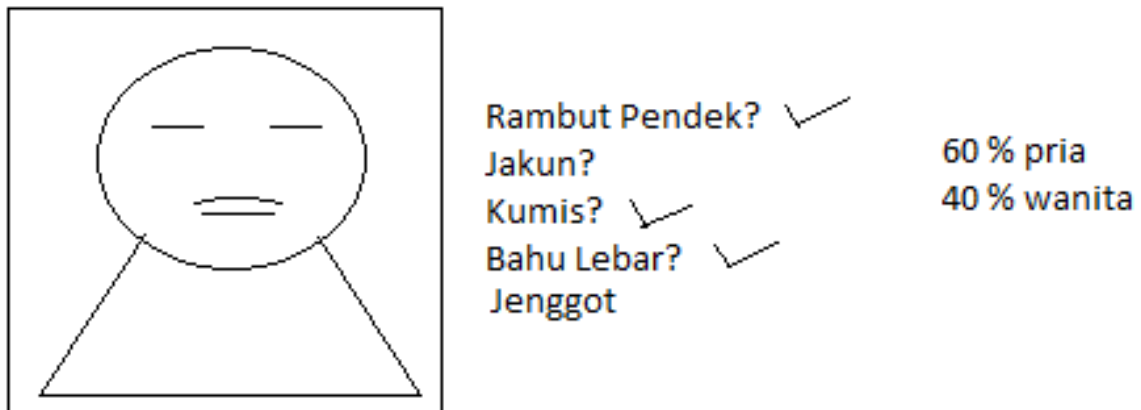


Figure 2.71: Contoh Penggunaan Information Gain

```
In [3]: import pandas as plum
...: durian = plum.read_csv('dataset/student-mat.csv', sep=';')
...: len(durian)
Out[3]: 395
```

Figure 2.72: code 1 hasil

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

[5 rows x 31 columns]
```

Figure 2.73: code 2 hasil

```
Out[5]:
...
   age  Medu  Fedu  ...  internet_yes  romantic_no  romantic_yes
0   18     4     4  ...              0             1              0
1   17     1     1  ...              1             1              0
2   15     1     1  ...              1             1              0
3   15     4     2  ...              1             0              1
4   16     3     3  ...              0             1              0

[5 rows x 57 columns]
```

Figure 2.74: code 3 hasil

```
....: import numpy as nanas
....: print("Passing: %d out of %d (%.2f%%)" % (nanas.sum(d_pass),
len(d_pass), 100*float(nanas.sum(d_pass)) / len(d_pass)))
Passing: 166 out of 395 (42.03%)
```

Figure 2.75: code 4 hasil

```
In [20]: from sklearn import tree
....: tomat = tree.DecisionTreeClassifier(criterion="entropy", max_depth=5)
....: tomat = tomat.fit(d_train_att, d_train_pass)
```

Figure 2.76: code 5 hasil

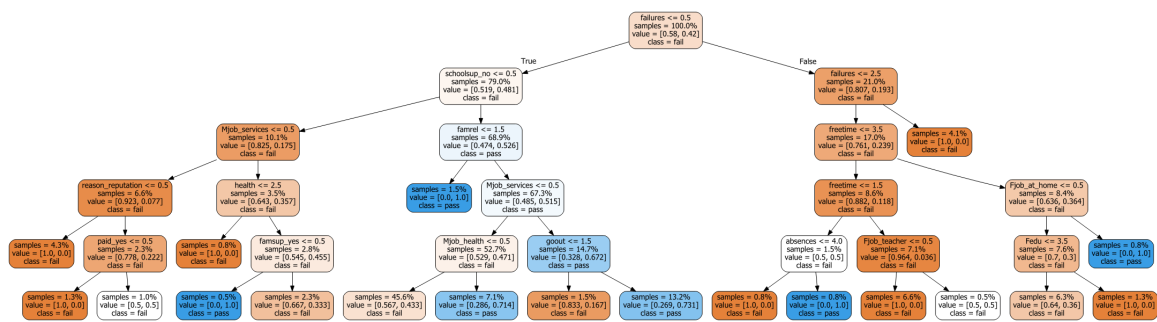


Figure 2.77: code 6 hasil

```
In [50]: tree.export_graphviz(tomat, out_file="student-performance.dot",
label="all", impurity=False, proportion=True,
....: feature_names=list(d_train_att),
class_names=["fail", "pass"],
....: filled=True, rounded=True)
```

Figure 2.78: code 7 hasil

```
Out[8]: 0.6644295302013423
```

Figure 2.79: code 8 hasil

```
Accuracy: 0.58 (+/- 0.09)
```

Figure 2.80: code 9 hasil

```
Max depth: 1, Accuracy: 0.58 (+/- 0.01)
Max depth: 2, Accuracy: 0.58 (+/- 0.03)
Max depth: 3, Accuracy: 0.57 (+/- 0.11)
Max depth: 4, Accuracy: 0.54 (+/- 0.06)
Max depth: 5, Accuracy: 0.59 (+/- 0.09)
Max depth: 6, Accuracy: 0.58 (+/- 0.10)
Max depth: 7, Accuracy: 0.57 (+/- 0.10)
Max depth: 8, Accuracy: 0.59 (+/- 0.11)
Max depth: 9, Accuracy: 0.60 (+/- 0.08)
Max depth: 10, Accuracy: 0.59 (+/- 0.03)
Max depth: 11, Accuracy: 0.58 (+/- 0.08)
Max depth: 12, Accuracy: 0.58 (+/- 0.10)
Max depth: 13, Accuracy: 0.57 (+/- 0.06)
Max depth: 14, Accuracy: 0.58 (+/- 0.10)
Max depth: 15, Accuracy: 0.58 (+/- 0.07)
Max depth: 16, Accuracy: 0.56 (+/- 0.07)
Max depth: 17, Accuracy: 0.57 (+/- 0.07)
Max depth: 18, Accuracy: 0.59 (+/- 0.06)
Max depth: 19, Accuracy: 0.58 (+/- 0.08)
```

Figure 2.81: code 10 hasil

```

array([[1.00000000e+00, 5.79751704e-01, 6.30768599e-03],
       [2.00000000e+00, 5.84847452e-01, 2.64856147e-02],
       [3.00000000e+00, 5.72122687e-01, 1.08161582e-01],
       [4.00000000e+00, 5.41806232e-01, 5.92037007e-02],
       [5.00000000e+00, 5.87280104e-01, 9.24946725e-02],
       [6.00000000e+00, 5.87247647e-01, 9.98900892e-02],
       [7.00000000e+00, 5.84684356e-01, 9.49754676e-02],
       [8.00000000e+00, 5.87248458e-01, 1.04805546e-01],
       [9.00000000e+00, 5.89782538e-01, 8.41669324e-02],
       [1.00000000e+01, 5.94973223e-01, 5.83834054e-02],
       [1.10000000e+01, 5.82154333e-01, 5.44767819e-02],
       [1.20000000e+01, 5.72090230e-01, 8.18652096e-02],
       [1.30000000e+01, 5.69623499e-01, 4.45760652e-02],
       [1.40000000e+01, 5.59432814e-01, 6.41830829e-02],
       [1.50000000e+01, 5.74750081e-01, 8.67458998e-02],
       [1.60000000e+01, 5.51932814e-01, 7.98351552e-02],
       [1.70000000e+01, 5.77250893e-01, 5.74806698e-02],
       [1.80000000e+01, 5.69623499e-01, 4.16011588e-02],
       [1.90000000e+01, 5.77345829e-01, 8.06083322e-02]])

```

Figure 2.82: code 11 hasil

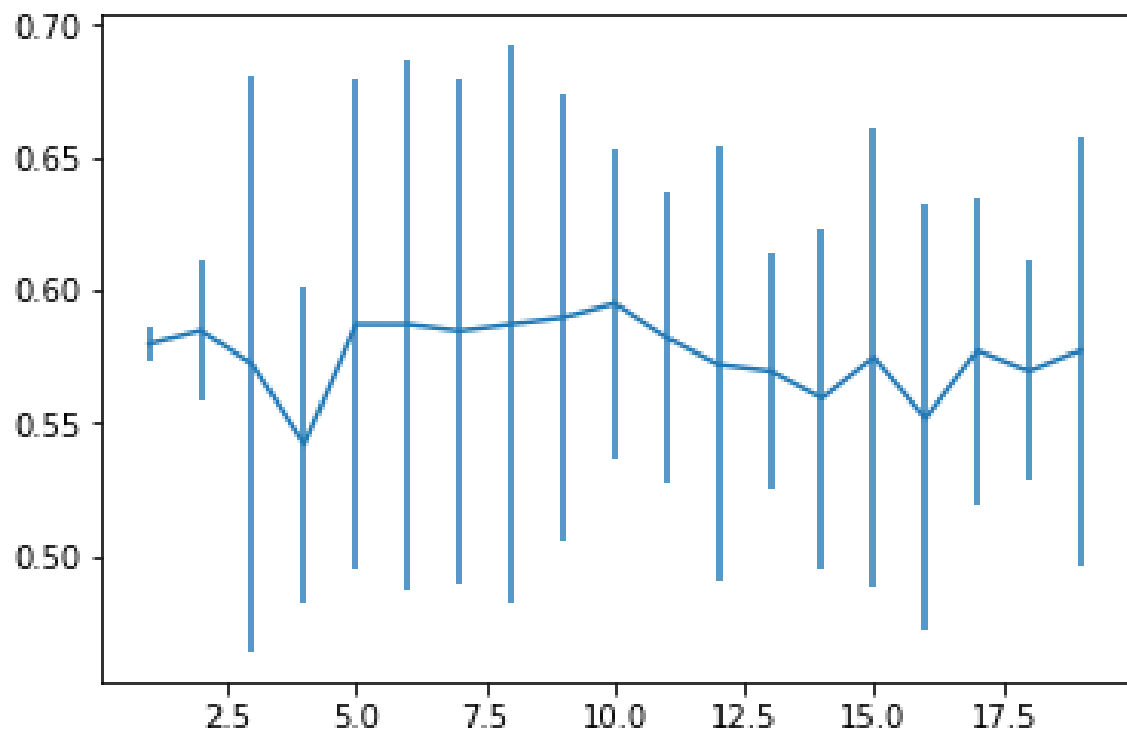


Figure 2.83: code 12 hasil

```
File "C:\Users\Fathi-PC\Anaconda3\lib\site-packages\graphviz\backend.py",
line 150, in run
    raise ExecutableNotFound(cmd)

ExecutableNotFound: failed to execute ['dot', '-Tsvg'], make sure the
graphviz executables are on your systems' PATH
```

Figure 2.84: Error Path

```
C:\Users\Fathi-PC>pip install graphviz
Collecting graphviz
  Using cached https://files.pythonhosted.org/
a/graphviz-0.10.1-py2.py3-none-any.whl
Installing collected packages: graphviz
Successfully installed graphviz-0.10.1

C:\Users\Fathi-PC>conda install graphviz
Collecting package metadata: done
Solving environment: done

## Package Plan ##
```

Figure 2.85: Fix Error

# Chapter 3

## Methods

### 3.1 Fathi Rabbani / 1164074

#### 3.1.1 Teori

##### 1. Random Forest

Random forest adalah suatu algoritma yang digunakan pada klasifikasi data dalam jumlah yang besar. Klasifikasi random forest dilakukan melalui penggabungan pohon (tree) dengan melakukan training pada sampel data yang dimiliki. 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. berikut adalah struktur dari Random Forest ada pada Gambar 3.1

##### 2. Membaca Dataset, Makna setiap file dan Menjelaskan data CUB-200-2011

###### (a) Membaca Data

- dengan membuka data yang sudah didownload yaitu data CUB-200-2011 atau data tentang perbandingan data jenis burung,
- lalu data tersebut dibuka dengan menggunakan aplikasi Spyder, dan dijalankan setiap baris Code yang ada.
- data yang ada pada folder CUB-200-2011 dibuka dengan menggunakan code dari Chapter 2 yang ada pada buku pembelajaran.

###### (b) Makna setiap File

- data yang terdapat pada file CUB-200-2011 ada data folder ATTRIBUTE, IMAGES, PARTS yang memiliki kegunaannya sendiri yang dimana



pada penggunaannya data yang dipakai adalah data `image_attribute_label` pada folder `attribute`, data `image_class_labels` dan data `classes`.

- file `image_attribute_label` berguna sebagai data awal yang digunakan untuk membaca data `attribute` yang terdapat pada masing - masing gambar burung yang ada.
- sedangkan file `image_class_label` yang ada pada folder `CUB-200-2011` berguna sebagai data yang akan membuat kolom baru pada dataset yang fungsinya adalah untuk memasukan hasil dari semua data yang dimiliki oleh `imgatt2`.
- dan file `classes` berguna sebagai dataset yang akan dipanggil oleh fungsi `code` untuk menampilkan nama dari data burung yang dimiliki.

(c) Isi Field

- file `image_attribute_label` berisi tentang data `attribute` yang ada pada data gambar file burung yang dimiliki di folder `image` pada `CUB-200-2011`
- file `image_class_label` berisi tentang data yang dimiliki oleh `attribute` dari `image_attribute_label` dimana data yang bernilai atau memiliki nilai disusun hingga menghasilkan data yang mudah dipahami.
- file `classes` berisi tentang data yang berguna untuk menampilkan data nama dari setiap data jenis burung yang dimiliki.

### 3. Cross Validation

Cross-validation adalah metode statistik yang dapat digunakan untuk mengevaluasi kinerja model atau algoritma dimana data dipisahkan menjadi dua subset yaitu data training dan data testing.

### 4. Score 44 Random Forest, 27 Decision Tree dan 29 SVM

- (a) merupakan hasil dari pengolahan tentang data jenis burung yang dimiliki setelah melalui proses pembagian data training dan testing yang menghasilkan score 44 persen sebagai pembandingan bahwa data yang diolah tersebut bernilai 44 persen tingkat kebenarannya atau keakuratannya.
- (b) lalu pada penggunaan decision tree yang menghasilkan nilai 27 persen menjelaskan bahwa data yang diolah dengan menggunakan decision tree

sebagai fungsi pembandingannya itu lebih kecil tingkat keakuratan hasilnya yang dimana kita mencari keakuratan data dari setiap jenis burung yang ada.

- (c) sedangkan dengan menggunakan SVM menghasilkan nilai sebesar 29 persen yang dimana merupakan nilai nertal menjelaskan bahwa data yang diolah masih belum akurat tingkat kesamaannya dengan data jenis burung yang dimiliki.

dari penjelasan tersebut disimpulkan bahwa penggunaan random forest dalam menentukan keakuratan data itu lebih besar scorenya dibandingkan menggunakan decision tree dan SVM.

#### 5. Membaca Confusion Matriks

```
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()
```

penggunaan code diatas adala cara untuk membaca dataset jenis burung dengan metode confusion matriks. dimana contoh hasil dari membaca data dengan menggunakan confusion matriks adalah sebagai berikut : Gambar 3.2

- 6. Voting pada Random Forest voting pada random forest berguna untuk mengambil nilai yang akan digunakan sebagai bandingan dari masing - masing tree yang ada untuk menghasilkan nilai final sebagai data yang diinginkan, contohnya adalah seperti berikut ini : Gambar 3.3

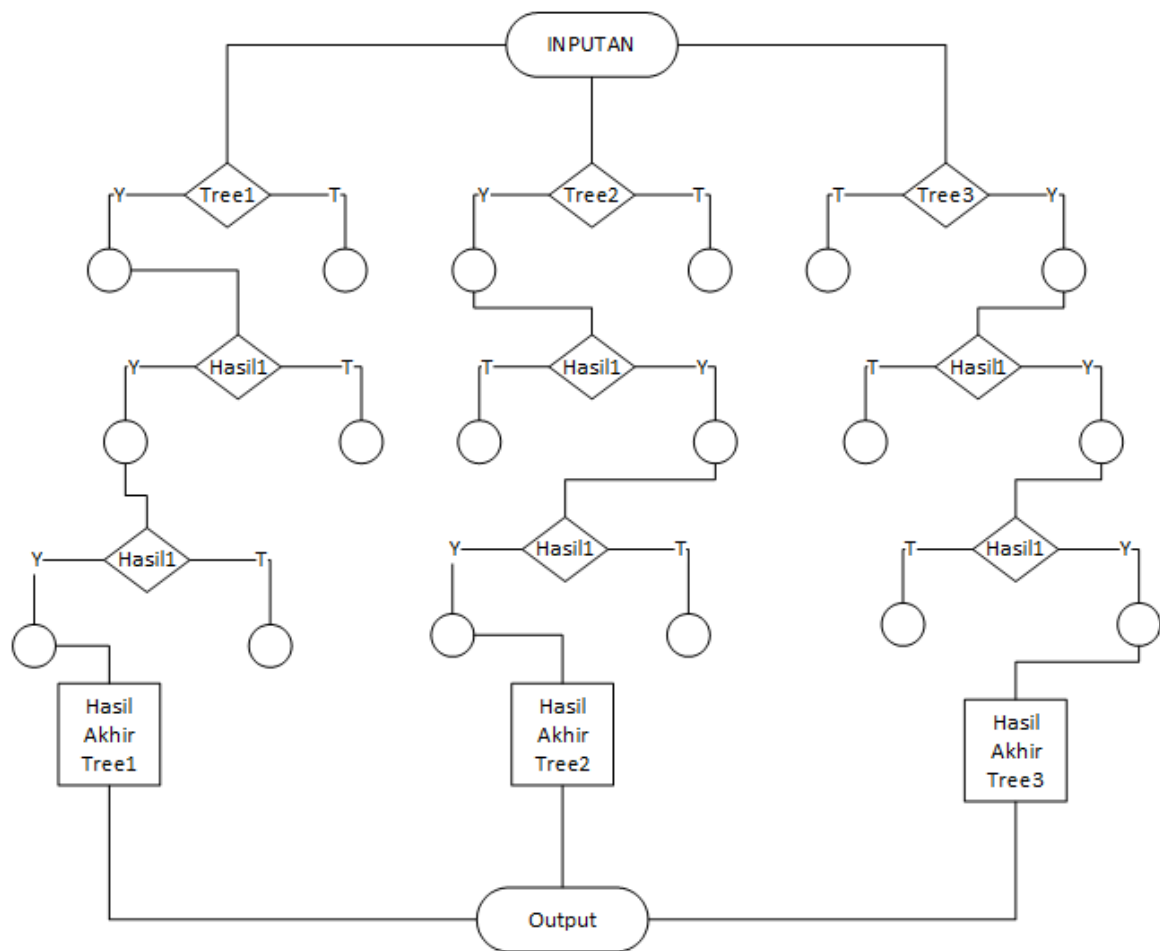


Figure 3.1: Random Forest

|                |                         | Hasil Prediksi      |                     |
|----------------|-------------------------|---------------------|---------------------|
|                |                         | Bukan Gerakan       | Gerakan             |
| Hasil Validasi | Jumlah Seluruh Grid (N) | True Negative (TN)  | False Positive (FP) |
|                | Bukan Gerakan           | False Negative (FN) | True Positive (TP)  |
|                |                         | Gerakan             |                     |

Figure 3.2: Hasil dari membaca data dengan Confusion Matriks

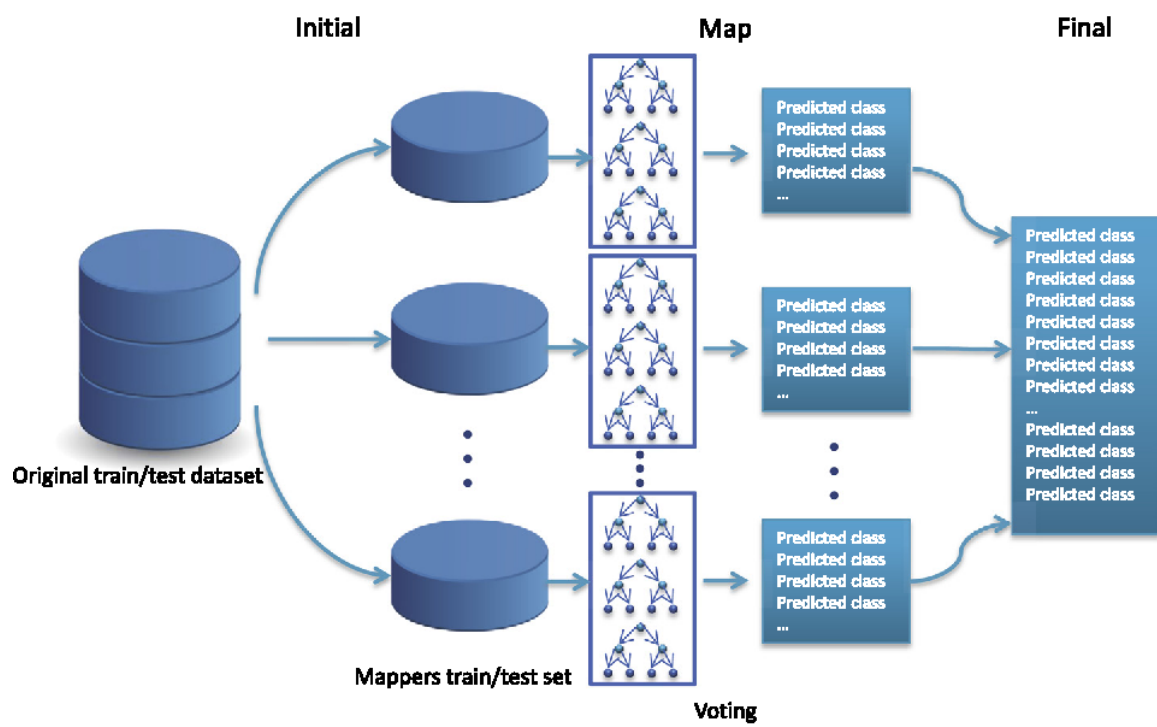


Figure 3.3: Voting Random Forest

# Chapter 4

## Experiment and Result

brief of experiment and result.

### 4.1 Experiment

Please tell how the experiment conducted from method.

### 4.2 Result

Please provide the result of experiment

# Chapter 5

## Conclusion

brief of conclusion

### **5.1 Conclusion of Problems**

Tell about solving the problem

### **5.2 Conclusion of Method**

Tell about solving using method

### **5.3 Conclusion of Experiment**

Tell about solving in the experiment

### **5.4 Conclusion of Result**

tell about result for purpose of this research.

# Chapter 6

## Discussion

Please tell more about conclusion and how to the next work of this study.



# Chapter 7

## Discussion

Please tell more about conclusion and how to the next work of this study.

# Chapter 8

## Discussion

Please tell more about conclusion and how to the next work of this study.

# Chapter 9

## Discussion

Please tell more about conclusion and how to the next work of this study.

# Chapter 10

## Discussion

Please tell more about conclusion and how to the next work of this study.

# Chapter 11

## Discussion

Please tell more about conclusion and how to the next work of this study.

# Chapter 12

## Discussion

Please tell more about conclusion and how to the next work of this study.

# Chapter 13

## Discussion

Please tell more about conclusion and how to the next work of this study.

# Chapter 14

## Discussion

Please tell more about conclusion and how to the next work of this study.



# Appendix A

## Form Penilaian Jurnal

gambar A.1 dan A.2 merupakan contoh bagaimana reviewer menilai jurnal kita.

| NO | UNSUR                                        | KETERANGAN                                                                                                                                                                        | MAKS | KETERANGAN                                                                                                                                                                                                                                              |
|----|----------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 1  | Keefektifan Judul Artikel                    | Maksimal 12 (dua belas) kata dalam Bahasa Indonesia atau 10 (sepuluh) kata dalam Bahasa Inggris                                                                                   | 2    | a. Tidak lugas dan tidak ringkas (0)<br>b. Kurang lugas dan kurang ringkas (1)<br>c. Ringkas dan lugas (2)                                                                                                                                              |
| 2  | Pencantuman Nama Penulis dan Lembaga Penulis |                                                                                                                                                                                   | 1    | a. Tidak lengkap dan tidak konsisten (0)<br>b. Lengkap tetapi tidak konsisten (0,5)<br>c. Lengkap dan konsisten (1)                                                                                                                                     |
| 3  | Abstrak                                      | Dalam Bahasa Indonesia dan Bahasa Inggris yang baik, jumlah 150-200 kata. Isi terdiri dari latar belakang, metode, hasil, dan kesimpulan. Isi tertuang dengan kalimat yang jelas. | 2    | a. Tidak dalam Bahasa Indonesia dan Bahasa Inggris (0)<br>b. Abstrak kurang jelas dan ringkas, atau hanya dalam Bahasa Inggris, atau dalam Bahasa Indonesia saja (1)<br>c. Abstrak yang jelas dan ringkas dalam Bahasa Indonesia dan Bahasa Inggris (2) |
| 4  | Kata Kunci                                   | Maksimal 5 kata kunci terpenting dalam paper                                                                                                                                      | 1    | a. Tidak ada (0)<br>b. Ada tetapi kurang mencerminkan konsep penting dalam artikel (0,5)<br>c. Ada dan mencerminkan konsep penting dalam artikel (1)                                                                                                    |
| 5  | Sistematika Pembahasan                       | Terdiri dari pendahuluan, tinjauan pustaka, metode penelitian, hasil dan pembahasan, kesimpulan dan saran, daftar pustaka                                                         | 1    | a. Tidak lengkap (0)<br>b. Lengkap tetapi tidak sesuai sistematika (0,5)<br>c. Lengkap dan bersistem (1)                                                                                                                                                |
| 6  | Pemanfaatan Instrumen Pendukung              | Pemanfaatan Instrumen Pendukung seperti gambar dan tabel                                                                                                                          | 1    | a. Tidak dimanfaatkan (0)<br>b. Kurang informatif atau komplementer (0,5)<br>c. Informatif dan komplementer (1)                                                                                                                                         |
| 7  | Cara Pengacuan dan Pengutipan                |                                                                                                                                                                                   | 1    | a. Tidak baku (0)<br>b. Kurang baku (0,5)<br>c. Baku (1)                                                                                                                                                                                                |
| 8  | Penyusunan Daftar Pustaka                    | Penyusunan Daftar Pustaka                                                                                                                                                         | 1    | a. Tidak baku (0)<br>b. Kurang baku (0,5)<br>c. Baku (1)                                                                                                                                                                                                |
| 9  | Peristilahan dan Kebahasaan                  |                                                                                                                                                                                   | 2    | a. Buruk (0)<br>b. Baik (1)<br>c. Cukup (2)                                                                                                                                                                                                             |
| 10 | Makna Sumbangan bagi Kemajuan                |                                                                                                                                                                                   | 4    | a. Tidak ada (0)<br>b. Kurang (1)<br>c. Sedang (2)<br>d. Cukup (3)<br>e. Tinggi (4)                                                                                                                                                                     |

Figure A.1: Form nilai bagian 1.

|                                           |                                                      |                                                                                                           |    |                                                                                                                                       |
|-------------------------------------------|------------------------------------------------------|-----------------------------------------------------------------------------------------------------------|----|---------------------------------------------------------------------------------------------------------------------------------------|
| 11                                        | Dampak Ilmiah                                        |                                                                                                           | 7  | a. Tidak ada (0)<br>b. Kurang (1)<br>c. Sedang (3)<br>d. Cukup (5)<br>e. Besar (7)                                                    |
| 12                                        | Nisbah Sumber Acuan Primer berbanding Sumber lainnya | Sumber acuan yang langsung merujuk pada bidang ilmiah tertentu, sesuai topik penelitian dan sudah teruji. | 3  | a. < 40% (1)<br>b. 40-80% (2)<br>c. > 80% (3)                                                                                         |
| 13                                        | Derajat Kemutakhiran Pustaka Acuan                   | Derajat Kemutakhiran Pustaka Acuan                                                                        | 3  | a. < 40% (1)<br>b. 40-80% (2)<br>c. > 80% (3)                                                                                         |
| 14                                        | Analisis dan Sintesis                                | Analisis dan Sintesis                                                                                     | 4  | a. Sedang (2)<br>b. Cukup (3)<br>c. Baik (4)                                                                                          |
| 15                                        | Penyimpulan                                          | Sangat jelas relevasinya dengan latar belakang dan pembahasan, dirumuskan dengan singkat                  | 3  | a. Kurang (1)<br>b. Cukup (2)<br>c. Baik (3)                                                                                          |
| 16                                        | Unsur Plagiat                                        |                                                                                                           | 0  | a. Tidak mengandung plagiat (0)<br>b. Terdapat bagian-bagian yang merupakan plagiat (-5)<br>c. Keseluruhannya merupakan plagiat (-20) |
| TOTAL                                     |                                                      |                                                                                                           | 36 |                                                                                                                                       |
| Catatan : Nilai minimal untuk diterima 25 |                                                      |                                                                                                           |    |                                                                                                                                       |

Figure A.2: form nilai bagian 2.

# Appendix B

## FAQ

M : Kalo Intership II atau TA harus buat aplikasi ? D : Ga harus buat aplikasi tapi harus ngoding

M : Pa saya bingung mau ngapain, saya juga bingung mau presentasi apa? D : Makanya baca de, buka jurnal topik ‘ganteng’ nah kamu baca dulu sehari 5 kali ya, 4 hari udah 20 tuh. Bingung itu tanda kurang wawasan alias kurang baca.

M : Pa saya sudah cari jurnal terindeks scopus tapi ga nemu. D : Kamu punya mata de? coba dicolok dulu. Kamu udah lakuin apa aja? tolong di list laporkan ke grup Tingkat Akhir. Tinggal buka google scholar klik dari tahun 2014, cek nama jurnalnya di scimagojr.com beres.

M : Pa saya belum dapat tempat intership, jadi ga tau mau presentasi apa? D : kamu kok ga nyambung, yang dipresentasikan itu yang kamu baca bukan yang akan kamu lakukan.

M : Pa ini jurnal harus yang terindex scopus ga bisa yang lain ? D : Index scopus menandakan artikel tersebut dalam standar semantik yang mudah dipahami dan dibaca serta bukan artikel asal jadi. Jika diluar scopus biasanya lebih sukar untuk dibaca dan dipahami karena tidak adanya proses review yang baik dan benar terhadap artikel.

M : Pa saya tidak mengerti D : Coba lihat standar alasan

M : Pa saya bingung D : Coba lihat standar alasan

M : Pa saya sibuk D : Mbahmu....

M : Pa saya ganteng D : Ndasmu....

M : Pa saya kece D : wes karepmu lah....

Biasanya anda memiliki alasan tertentu jika menghadapi kendala saat proses bimbingan, disini saya akan melakukan standar alasan agar persepsi yang diterima sama dan tidak salah kaprah. Penggunaan kata alasan tersebut antara lain :

1. Tidak Mengerti : anda boleh menggunakan alasan ini jika anda sudah melakukan tahapan membaca dan meresumekan 15 jurnal. Sudah mencoba dan mempraktekkan teorinya dengan mencari di youtube dan google minimal 6 jam sehari selama 3 hari berturut-turut.

2. Bingung : anda boleh mengatakan alasan bingung setelah maksimal dalam berusaha menyelesaikan tugas bimbingan dari dosen(sudah dilakukan semua). Anda belum bisa mengatakan alasan bingung jika anda masih belum menyelesaikan tugas bimbingan dan poin nomor 1 diatas. Setelah anda menyelesaikan tugas bimbingan secara maksimal dan tahap 1 poin diatas, tapi anda masih tetap bingung maka anda boleh memakai alasan ini.

# Bibliography

- [1] Abdillah Baraja. Kecerdasan buatan tinjauan historikal. *Speed-Sentra Penelitian Engineering dan Edukasi*, 1(1), 2008.
- [2] Joshua Eckroth. *Python Artificial Intelligence Projects for Beginners: Get up and running with Artificial Intelligence using 8 smart and exciting AI applications*. Packt Publishing Ltd, 2018.
- [3] Herny Februariyanti and Eri Zuliarso. Klasifikasi dokumen berita teks bahasa indonesia menggunakan ontologi. *Dinamik*, 17(1), 2012.
- [4] Deny Kurniawan. Regresi linier. *R-Foundation for Statistical Computing. Vienna, Austria*, 17, 2008.
- [5] Stuart J Russell and Peter Norvig. *Artificial intelligence: a modern approach*. Malaysia; Pearson Education Limited,, 2016.
- [6] Xiaojin Zhu and Andrew B Goldberg. Introduction to semi-supervised learning. *Synthesis lectures on artificial intelligence and machine learning*, 3(1):1–130, 2009.