

# 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

<b>1</b>	<b>Mengenal Kecerdasan Buatan dan Scikit-Learn</b>	<b>1</b>
1.1	Teori . . . . .	1
1.2	Instalasi . . . . .	2
1.3	Penanganan Error . . . . .	2
1.4	Andri Fajar S/1164065 . . . . .	2
1.4.1	TEORI . . . . .	2
1.4.2	Instalasi . . . . .	4
1.4.3	Mencoba Learning and predicting . . . . .	4
1.4.4	Mencoba Model Persistance . . . . .	5
1.4.5	Mencoba Conventions . . . . .	9
1.5	Penanganan Error . . . . .	12
<b>2</b>	<b>Related Works</b>	<b>15</b>
2.1	Same Topics . . . . .	15
2.1.1	Topic 1 . . . . .	15
2.1.2	Topic 2 . . . . .	15
2.2	Same Method . . . . .	15
2.2.1	Method 1 . . . . .	15
2.2.2	Method 2 . . . . .	15
2.3	Andri Fajar Sunandhar/1164065 . . . . .	16
2.3.1	binary classification dilengkapi ilustrasi gambar . . . . .	16
2.3.2	supervised learning dan unsupervised learning dan clustering dengan ilustrasi gambar . . . . .	16
2.3.3	evaluasi dan akurasi dari buku dan disertai ilustrasi contoh den- gan gambar . . . . .	18
2.3.4	bagaimana cara membuat dan membaca confusion matrix, buat confusion matrix . . . . .	19

2.3.5	bagaimana K-fold cross validation bekerja dengan gambar ilustrasi . . . . .	20
2.3.6	decision tree dengan gambar ilustrasi . . . . .	20
2.3.7	Information Gain dan entropi dengan gambar ilustrasi . . . .	21
2.4	scikit-learn . . . . .	22
2.5	Penanganan Error . . . . .	28
2.5.1	Error Graphviz . . . . .	28
<b>3</b>	<b>Methods</b>	<b>30</b>
3.1	The data . . . . .	30
3.2	Method 1 . . . . .	30
3.3	Method 2 . . . . .	30
3.4	Andri Fajar Sunandhar/1164065 . . . . .	30
3.4.1	Teori . . . . .	30
3.4.2	Praktek Program . . . . .	34
3.4.3	Penanganan Error . . . . .	42
<b>4</b>	<b>Experiment and Result</b>	<b>47</b>
4.1	Experiment . . . . .	47
4.2	Result . . . . .	47
4.3	Andri Fajar Sunandhar/1164065 . . . . .	47
4.3.1	Teori . . . . .	47
4.3.2	Praktek Program . . . . .	49
4.3.3	Penanganan Error . . . . .	54
<b>5</b>	<b>Conclusion</b>	<b>55</b>
5.1	Conclusion of Problems . . . . .	55
5.2	Conclusion of Method . . . . .	55
5.3	Conclusion of Experiment . . . . .	55
5.4	Conclusion of Result . . . . .	55
5.5	Andri Fajar Sunandhar / 1164065 . . . . .	55
5.5.1	Teori . . . . .	55
5.5.2	Praktek Program . . . . .	59
<b>6</b>	<b>Discussion</b>	<b>65</b>
<b>7</b>	<b>Discussion</b>	<b>66</b>

8 Discussion	67
9 Discussion	68
10 Discussion	69
11 Discussion	70
12 Discussion	71
13 Discussion	72
14 Discussion	73
A Form Penilaian Jurnal	74
B FAQ	77
Bibliography	79

# List of Figures

1.1	conda install scikit-learn. . . . .	4
1.2	Melihat Version. . . . .	5
1.3	Install pip. . . . .	5
1.4	Hasil Kompile. . . . .	6
1.5	Hasil Kompile. . . . .	6
1.6	Hasil Kompile. . . . .	7
1.7	Membuka Python . . . . .	7
1.8	Estimator Sklearn . . . . .	8
1.9	Mendefinisikan Classifier . . . . .	8
1.10	Memanggil Classifier . . . . .	8
1.11	Memprediksi Nilai Baru . . . . .	8
1.12	Hasil Classifier . . . . .	8
1.13	Hasil Classifier . . . . .	8
1.14	Pickle Python . . . . .	9
1.15	Classifier Pickle . . . . .	9
1.16	Joblib . . . . .	9
1.17	Deklarasi Numpy . . . . .	10
1.18	Contoh Casting . . . . .	10
1.19	FitTransform . . . . .	10
1.20	Regresi Yang Dilempar . . . . .	11
1.21	Memperbaharui Parameter . . . . .	11
1.22	MultiClass . . . . .	12
1.23	MultiClass biner 2D . . . . .	12
1.24	MultiLabel . . . . .	13
1.25	Error Import . . . . .	13
1.26	Instal Library Joblib . . . . .	14
1.27	Import Library Joblib . . . . .	14
2.1	Binary Classification . . . . .	16



2.2	Supervised Learning . . . . .	17
2.3	Unsupervised Learning . . . . .	17
2.4	Cluster . . . . .	18
2.5	Evaluasi dan Akurasi . . . . .	19
2.6	K-fold cross validation . . . . .	20
2.7	Decision Tree . . . . .	21
2.8	Information gain . . . . .	21
2.9	Loading Dataset . . . . .	22
2.10	Generate Binary Label . . . . .	22
2.11	One-hot Encoding . . . . .	23
2.12	Shuffle Rows . . . . .	24
2.13	Fit Decision Tree . . . . .	24
2.14	Fit Decision Tree . . . . .	25
2.15	Fit Decision Tree . . . . .	25
2.16	Score . . . . .	25
2.17	Cross Val Score . . . . .	26
2.18	Max Depth . . . . .	26
2.19	Depth in Range . . . . .	27
2.20	Matplotlib . . . . .	28
2.21	Error Graphviz . . . . .	28
2.22	install Graphviz . . . . .	29
2.23	Solving Environment . . . . .	29
2.24	Evaluasi Error . . . . .	29
3.1	Random Forest. . . . .	31
3.2	Kode membaca file.csv . . . . .	31
3.3	Window Console . . . . .	31
3.4	Variable Explorer . . . . .	32
3.5	Dataset Cell . . . . .	32
3.6	Pohon Keputusan . . . . .	33
3.7	Data Testing . . . . .	34
3.8	Voting. . . . .	35
3.9	Aplikasi Pandas . . . . .	35
3.10	Hasil Pandas . . . . .	35
3.11	Aplikasi Numpy . . . . .	36
3.12	Hasil Numpy . . . . .	36

3.13 Aplikasi Matplotlib . . . . .	36
3.14 Hasil Matplotlib . . . . .	37
3.15 Gambar1 . . . . .	37
3.16 Gambar2 . . . . .	38
3.17 Gambar3 . . . . .	38
3.18 Gambar 4 . . . . .	39
3.19 Gambar 5 . . . . .	39
3.20 Gambar 6 . . . . .	40
3.21 Gambar 7 . . . . .	40
3.22 Gambar 8 . . . . .	41
3.23 Gambar 9 . . . . .	41
3.24 Gambar 10 . . . . .	42
3.25 Gambar 11 . . . . .	42
3.26 Gambar 12 . . . . .	43
3.27 Gambar 13 . . . . .	43
3.28 Gambar 14 . . . . .	43
3.29 Gambar 15 . . . . .	43
3.30 Gambar 16 . . . . .	43
3.31 Gambar 17 . . . . .	43
3.32 Gambar 18 . . . . .	44
3.33 Memetakan ke confusion matrix . . . . .	44
3.34 Melihat hasil . . . . .	44
3.35 Melakukan Plot . . . . .	44
3.36 Plotting nama data . . . . .	44
3.37 Melakukan perintah plot . . . . .	45
3.38 SVM . . . . .	45
3.39 Decission Tree . . . . .	45
3.40 Pengecekan cross validation random forest . . . . .	45
3.41 Pengecekan cross validation decision tree . . . . .	45
3.42 Pengamatan Komponen . . . . .	45
3.43 Plot informasi . . . . .	46
3.44 Error . . . . .	46
4.1 Klasifikasi teks . . . . .	48
4.2 Klasifikasi bunga . . . . .	48
4.3 Teknik YouTube . . . . .	48

4.4	Bag of Word . . . . .	49
4.5	TF IDF . . . . .	50
4.6	Pandas . . . . .	50
4.7	Hasil Pandas . . . . .	50
4.8	Memecah dataframe . . . . .	51
4.9	Hasil memecah dataframe . . . . .	51
4.10	Vektorisasi dan klasifikasi . . . . .	51
4.11	Decission Tree . . . . .	52
4.12	Hasil klasifikasi SVM . . . . .	52
4.13	Decission Tree . . . . .	53
4.14	ploting confusion matrix . . . . .	53
4.15	Program cross validation . . . . .	53
4.16	Program pengamatan komponen informasi . . . . .	54
4.17	skrinsut error . . . . .	54
4.18	Solusi error . . . . .	54
5.1	Gambar Vektorisasi Kata. . . . .	56
5.2	Gambar Vektorisasi Dataset Google. . . . .	56
5.3	Gambari Vektorisasi Kata. . . . .	57
5.4	Gambar Vektorisasi Dokumen. . . . .	58
5.5	Gambar Mean. . . . .	58
5.6	Gambar Deviasi. . . . .	59
5.7	Gambar Skip-Gram. . . . .	60
5.8	Gambar Vektor Love. . . . .	60
5.9	Gambar vektor faith. . . . .	61
5.10	Gambar vektor fall. . . . .	61
5.11	Gambar vektor sick. . . . .	62
5.12	Gambar vektor clear. . . . .	62
5.13	Gambar vektor shine. . . . .	63
5.14	Gambar vektor bag. . . . .	63
5.15	Gambar Vektor car. . . . .	63
5.16	Gambar Vektor wash. . . . .	63
5.17	Gambar vektor motor. . . . .	63
5.18	Gambar vektor cycle. . . . .	63
5.19	Gambar Similariti. . . . .	64
5.20	Gambar Extract Words. . . . .	64

5.21	Gambar PermuteSentences. . . . .	64
A.1	Form nilai bagian 1. . . . .	75
A.2	form nilai bagian 2. . . . .	76

# Chapter 1

## Mengenai Kecerdasan Buatan dan Scikit-Learn

Buku umum yang digunakan adalah [2] dan untuk sebelum UTS menggunakan buku *Python Artificial Intelligence Projects for Beginners*[1]. 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)

## 1.4 Andri Fajar S/1164065

### 1.4.1 TEORI

1. Definisi, Sejarah, Dan Perkembangan Sejarah AI

Didefinisikan kecerdasan yang ditunjukkan oleh suatu entitas buatan. Umumnya dianggap komputer. Kecerdasan Buatan (Artificial Intelligence atau AI) didefinisikan sebagai kecerdasan yang ditunjukkan oleh suatu entitas buatan. Sistem seperti ini umumnya dianggap komputer. Kecerdasan dimasukkan ke

dalam mesin (komputer) agar dapat melakukan pekerjaan seperti yang dapat dilakukan manusia. Kecerdasan Buatan (Artificial Intelligence atau AI) didefinisikan sebagai kecerdasan yang ditunjukkan oleh suatu entitas buatan. Sistem seperti ini umumnya di anggap komputer. Kecerdasan diciptakan dan dimasukkan melakukan pekerjaan seperti yang dapat dilakukan manusia.

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[?].

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.

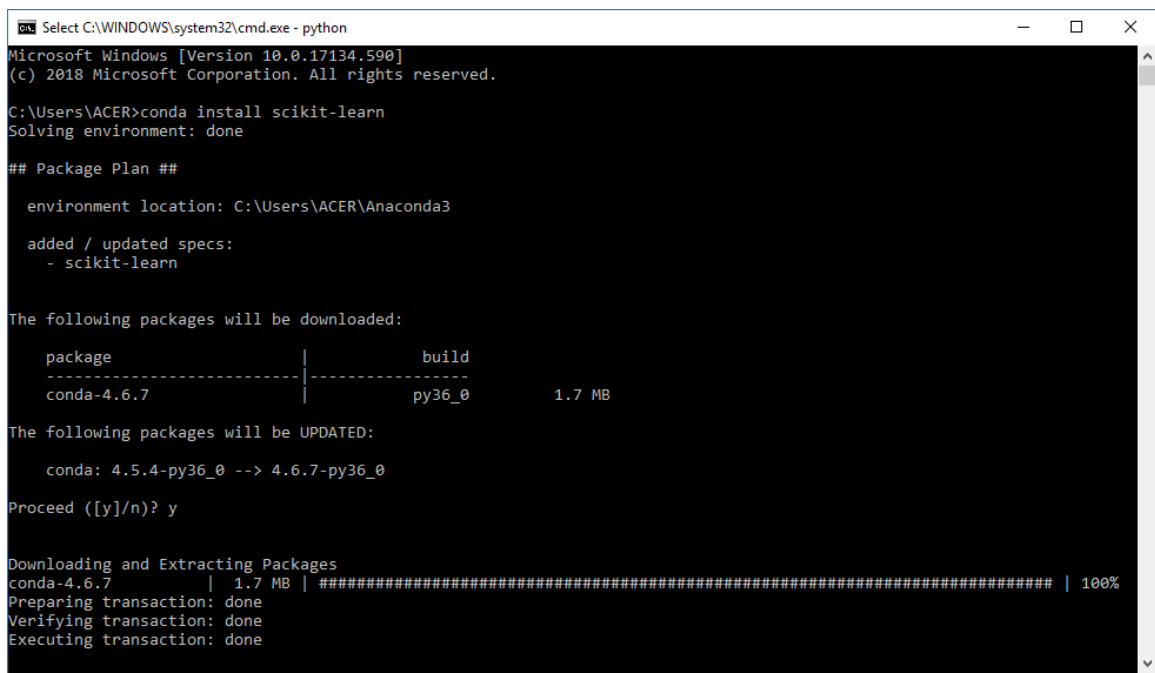
Data set adalah cabang aplikasi dari Artificial Intelligence/Kecerdasan Buatan yang fokus pada pengembangan sebuah sistem yang mampu belajar sendiri tanpa harus berulang kali di program oleh manusia.

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

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

### 1.4.2 Instalasi

- Memberikan perintah conda install scikit-learn di cmd, lihat gambar 1.1
- Melihat versinya dengan memberikan perintah conda -version dan python -version, lihat gambar 1.2
- Install pip, lihat pada gambar 1.3
- Hasil Kompile, lihat gambar 1.4
- Import dataset kemudian load iris dan data dari digits, lihat gambar 1.5
- Melihat data digits



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

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

## Package Plan ##

  environment location: C:\Users\ACER\Anaconda3

  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 1.1: conda install scikit-learn.

### 1.4.3 Mencoba Learning and predicting

1. Buka CMD lalu ketikkan perintah Python.
2. "from sklearn import svm" artinya akan memanggil dan menggunakan estimator dari kelas sklearn.svm.SVC



```
C:\Users\ACER>conda --version
conda 4.6.7

C:\Users\ACER>python --version
Python 3.6.5
```

Figure 1.2: Melihat Version.

```
C:\Users\ACER>pip install -U scikit-learn
Collecting scikit-learn
  Downloading https://files.pythonhosted.org/packages/ee/c8/c89ebdc0d7dbba6e6fd222daabd257da3c28a967dd7c352d4272b2e1cef6
/scikit_learn-0.20.2-cp36-cp36m-win32.whl (4.3MB)
    100% |#####| 4.3MB 622kB/s
Collecting numpy>=1.8.2 (from scikit-learn)
  Downloading https://files.pythonhosted.org/packages/84/10/f1f99ba67aff4c3fb033571e87876ed0403114b13bc70cc125372b0c1dcb
/numpy-1.16.1-cp36-cp36m-win32.whl (10.0MB)
    100% |#####| 10.0MB 863kB/s
Collecting scipy>=0.13.3 (from scikit-learn)
  Downloading https://files.pythonhosted.org/packages/b7/b6/eedfa8b002ffae365c3f957154a9c29cb91a8e657808749c78758f0f8110
/scipy-1.2.1-cp36-cp36m-win32.whl (26.9MB)
    100% |#####| 27.0MB 89kB/s
Installing collected packages: numpy, scipy, scikit-learn
Successfully installed numpy-1.16.1 scikit-learn-0.20.2 scipy-1.2.1
You are using pip version 18.1, however version 19.0.3 is available.
You should consider upgrading via the 'python -m pip install --upgrade pip' command.
```

Figure 1.3: Install pip.

3. disini gamma didefinisikan secara manual
4. Estimator clf (for classifier) pertama kali dipasang pada model. Ini dilakukan dengan melewati training set ke metode fit. Untuk training set, akan menggunakan semua gambar dari set data yang ada, kecuali untuk gambar terakhir, yang dicadangan untuk prediksi. Pada skrip dibawah memilih training set dengan sintaks Python [: -1], yang menghasilkan array baru yang berisi semua kecuali item terakhir dari digits.data
5. Pada penggalan skrip dibawah, ini menunjukan prediksi nilai baru menggunakan gambar terakhir dari digits.data.

#### 1.4.4 Mencoba Model Persistence

1. "from sklearn import svm" artinya akan mengimport sebuah Support Vector Machine(SVM) yang merupakan algoritma classification yang akan diambil dari Scikit-Learn.

```
C:\Users\ACER>python
Python 3.6.5 (v3.6.5:f59c0932b4, Mar 28 2018, 16:07:46) [MSC v.1900 32 bit (Intel)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> print('andri')
andri
```

Figure 1.4: Hasil Kompile.

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

Figure 1.5: Hasil Kompile.

2. "from sklearn import datasets" artinya akan mengambil package datasets dari Scikit-Learn.
3. ketikan, `clf = svm.SVC(gamma='scale')` berfungsi untuk mendeklarasikan suatu value yang bernama `clf` yang berisi `gamma`.
4. Ketikan, `X, y = iris.data, iris.target`, artinya `X` sebagai data iris, dan `y` merupakan larik target.
5. Ketikan, `clf.fit(X, y)` berfungsi untuk melakukan pengujian classifier. hasilnya seperti ini

Dari gambar diatas dapat dijelaskan bahwa akan mengimport Pickle dari Python. Pickle digunakan untuk serialisasi dan de-serialisasi struktur objek Python. Objek apa pun dengan Python dapat di-Pickle sehingga dapat disimpan di disk. kemudian menyimpan data objek ke file CLF sebelumnya dengan menggunakan function `pickle.dumps(clf)`.

7. Setelah mengetikan fungsi fungsi diatas, selanjutnya ketikan "`clf2 = pickle.loads(s)`" yang artinya `pickle.loads` digunakan untuk memuat data pickle dari string byte. "S" dalam loads mengacu pada fakta bahwa dalam Python 2, data dimuat dari string.

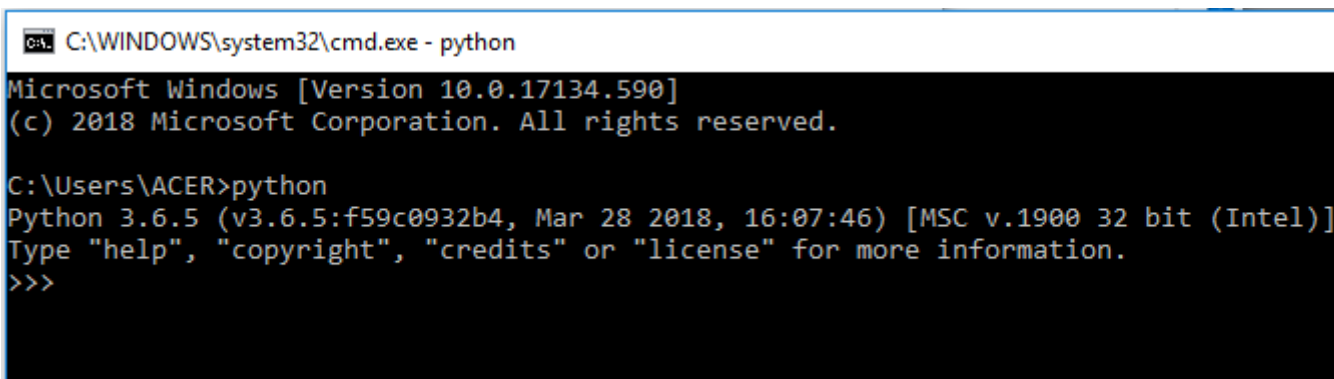
Pada gambar diatas dilakukan pengujian nilai baru dengan menggunakan "`clf2.predict(X[0:1])`" dengan target asumsinya (0,1) hasilnya berbentuk array.

```
>>> 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.]]
>>>
```

Figure 1.6: Hasil Kompile.

9. "from joblib import dump , load" yang artinya akan Merekonstruksi objek Python dari file yang sudah ada.

dump(clf, 'filename.joblib') akan merekontruksi file CLF yang tadi sudah dideklarasikan.  
 clf = load('filename.joblib') untuk mereload model yang sudah di Pickle



```
C:\WINDOWS\system32\cmd.exe - python
Microsoft Windows [Version 10.0.17134.590]
(c) 2018 Microsoft Corporation. All rights reserved.

C:\Users\ACER>python
Python 3.6.5 (v3.6.5:f59c0932b4, Mar 28 2018, 16:07:46) [MSC v.1900 32 bit (Intel)]
Type "help", "copyright", "credits" or "license" for more information.
>>>
```

Figure 1.7: Membuka Python

```
C:\WINDOWS\system32\cmd.exe - python
Microsoft Windows [Version 10.0.17134.590]
(c) 2018 Microsoft Corporation. All rights reserved.

C:\Users\ACER>python
Python 3.6.5 (v3.6.5:f59c0932b4, Mar 28 2018, 16:07:46) [MSC v.1900 32 bit (Intel)]
Type "help", "copyright", "credits" or "license" for more information.
>>> from sklearn import svm
>>>
```

Figure 1.8: Estimator Sklearn

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

Figure 1.9: Mendefinisikan Classifier

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

Figure 1.10: Memanggil Classifier

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

Figure 1.11: Memprediksi Nilai Baru

```
>>> clf.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 1.12: Hasil Classifier

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

6.

Figure 1.13: Hasil Classifier

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

Figure 1.14: Pickle Python

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

Figure 1.15: Classifier Pickle

### 1.4.5 Mencoba Conventions

1. Import numpy as np, digunakan untuk mengimport Numpy sebagai np.  
From sklearn import randomprojection artinya modul yang mengimplementasikan cara sederhana dan efisien secara komputasi untuk mengurangi dimensi data dengan memperdagangkan sejumlah akurasi yang terkendali (sebagai varian tambahan) untuk waktu pemrosesan yang lebih cepat dan ukuran model yang lebih kecil.

Pada gambar diatas dapat dijelaskan bahwa :

rng = np.random.RandomState(0), digunakan untuk menginisialisasikan random number generator.

X = rng.rand(10, 2000) artinya akan merandom value antara 10 sampai 2000.

X = np.array(X, dtype='float32') Array numpy terdiri dari buffer memori "mentah" yang diartikan sebagai array melalui "views". Anda dapat menganggap semua array numpy sebagai tampilan. Mendeklarasikan X sebagai float32.

3. Dalam contoh ini, X adalah float32, yang dilemparkan ke float64 oleh fittransform (X).
4. Target regresi dilemparkan ke float64 dan target klasifikasi dipertahankan.  
list(clf.predict(irisdata[:3])), akan memprediksi 3 data dari iris.  
clf.fit irisdata, iristargetnames[iristarget] menguji classifier dengan ada targetnya yaitu irisnya sendiri.

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

Figure 1.16: Joblib

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

Figure 1.17: Deklarasi Numpy

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

Figure 1.18: Contoh Casting

`list(clf.predict(irisdata[:3]))`, setelah diuji maka akan muncul datanya seperti dibawah ini

Di sini, prediksi pertama () mengembalikan array integer, karena `iristarget` (array integer) yang digunakan sesuai. Prediksi kedua () mengembalikan array string, karena `iristargetnames` cocok.

#### 5. Refitting dan Memperbaharui Parameter

`y = rngbinomial(1, 0.5, 100)` , random value dengan angka binomial atau suku dua untuk y

`clfsetparams(kernel='linear')fit(X, y)` mengubah kernel default menjadi linear  
`clfsetparams(kernel='rbf', gamma='scale')fit(X, y)` Di sini, kernel default rbf pertama kali diubah menjadi linear melalui

`SVCsetparams()` setelah estimator dibuat, dan diubah kembali ke rbf untuk mereparasi estimator dan membuat prediksi kedua.

#### 6. MultiClass VS MultiLabel Classifier

`from sklearn.multiclass import OneVsRestClassifier` , adalah ketika kita ingin melakukan klasifikasi multiclass atau multilabel dan baik untuk menggunakan `OneVsRestClassifier` per kelas. Untuk setiap classifier, kelas tersebut dipasang terhadap semua kelas lainnya. (Ini cukup jelas dan itu berarti bahwa masalah klasifikasi multiclass / multilabel dipecah menjadi beberapa masalah klasifikasi biner).

```
>>> transformer = random_projection.GaussianRandomProjection()
>>> X_new = transformer.fit_transform(X)
>>> X_new.dtype
dtype(<'float64'>)
```

Figure 1.19: FitTransform

```

>>> from sklearn import datasets
>>> from sklearn.svm import SVC
>>> iris = datasets.load_iris()
>>> clf = SVC(gamma='scale')
>>> 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)
>>> list(clf.predict(iris.data[:3]))
[0, 0, 0]
>>> 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)
>>> list(clf.predict(iris.data[:3]))
['setosa', 'setosa', 'setosa']

```

Figure 1.20: Regresi Yang Dilempar

```

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

```

Figure 1.21: Memperbaharui Parameter

```

>>> 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='scale',
... random_state=0))
>>> classif.fit(X, y).predict(X)
array([0, 0, 1, 1, 2])

```

Figure 1.22: MultiClass

```

>>> y = LabelBinarizer().fit_transform(y)
>>> classif.fit(X, y).predict(X)
array([[1, 0, 0],
       [1, 0, 0],
       [0, 1, 0],
       [0, 0, 0],
       [0, 0, 0]])

```

Figure 1.23: MultiClass biner 2D

from sklearn.preprocessing import LabelBinarizer ,adalah kelas utilitas untuk membantu membuat matriks indikator label dari daftar label multi-kelas Dalam gambar dibawah, classifier cocok pada array 1d label multiclass dan oleh karena itu metode predict () memberikan prediksi multiclass yang sesuai.

7. Di sini, classifier cocok () pada representasi label biner 2d dari y, menggunakan LabelBinarizer. Dalam hal ini predict () mengembalikan array 2d yang mewakili prediksi multilabel yang sesuai.
8. from sklearn.preprocessing import MultiLabelBinarizer , artinya Transformasi antara iterable dari iterables dan format multilabel. Dalam hal ini, penggolongnya sesuai pada setiap instance yang diberi beberapa label. MultiLabelBinarizer digunakan untuk membuat binarize array 2d dari multilabel agar sesuai. Hasilnya, predict () mengembalikan array 2d dengan beberapa label yang diprediksi untuk setiap instance.

## 1.5 Penanganan Error

1. Berikut ini merupakan eror yang ditemui pada saat melakukan percobaan skrip.
2. Pada gambar eror diatas, kode erornya adalah "ImportError: No Module Named" artinya mengalami masalah saat mengimpor modul yang ditentukan.



```

>>> 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)
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 1.24: MultiLabel

```

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

```

Figure 1.25: Error Import

3. Solusinya bisa dilakukan seperti berikut :  
 eror diatas terjadi dikarenakan Library Joblib belum terinstal pada PC. Maka dari itu sekarang kita harus menginstalnya dulu.
4. Buka CMD, kemudian ketikan "pip install joblib" tunggu sampai instalasi berhasil seperti gambar berikut.
5. Apabila sudah terinstall, dapat dilakukan lagi import library joblib, maka akan berhasil seperti dibawah berikut

```
C:\Users\ACER>pip install joblib
Collecting joblib
  Downloading https://files.pythonhosted.org/packages/cd/c1/50a758e8247561e58cb8730
/joblib-0.13.2-py2.py3-none-any.whl (278kB)
    100% |████████████████████████████████████████| 286kB 535kB/s
Installing collected packages: joblib
Successfully installed joblib-0.13.2
You are using pip version 18.1, however version 19.0.3 is available.
You should consider upgrading via the 'python -m pip install --upgrade pip' command
C:\Users\ACER>
```

Figure 1.26: Instal Library Joblib

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

Figure 1.27: Import Library Joblib

# Chapter 2

## Related Works

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

### 2.1 Same Topics

Cite every latest journal with same topic

#### 2.1.1 Topic 1

cite for first topic

#### 2.1.2 Topic 2

if you have two topics you can include here to

### 2.2 Same Method

write and cite latest journal with same method

#### 2.2.1 Method 1

cite and paraphrase method 1

#### 2.2.2 Method 2

cite and paraphrase method 2 if you have more method please add new subsection.

## 2.3 Andri Fajar Sunandhar/1164065

### 2.3.1 binary classification dilengkapi ilustrasi gambar

1. Binary classification yaitu berupa kelas positif dan kelas negatif. Klasifikasi biner adalah dikotomisasi yang diterapkan untuk tujuan praktis, dan dalam banyak masalah klasifikasi biner praktis, kedua kelompok tidak simetris - dari-pada akurasi keseluruhan, proporsi relatif dari berbagai jenis kesalahan yang menarik. Misalnya, dalam pengujian medis, false positive (mendeteksi penyakit ketika tidak ada) dianggap berbeda dari false negative (tidak mendeteksi penyakit ketika hadir).

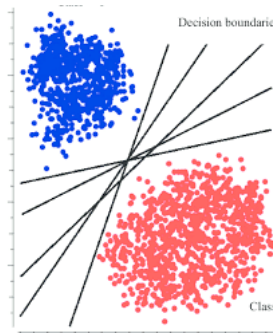


Figure 2.1: Binary Classification

### 2.3.2 supervised learning dan unsupervised learning dan clustering dengan ilustrasi gambar

1. Supervised learning adalah tugas pembelajaran mesin untuk mempelajari suatu fungsi yang memetakan input ke output berdasarkan contoh pasangan input-output. Ini menyimpulkan fungsi dari data pelatihan berlabel yang terdiri dari serangkaian contoh pelatihan. Dalam pembelajaran yang diawasi, setiap contoh adalah pasangan yang terdiri dari objek input (biasanya vektor) dan nilai output yang diinginkan (juga disebut sinyal pengawas). Algoritma pembelajaran yang diawasi menganalisis data pelatihan dan menghasilkan fungsi yang disimpulkan, yang dapat digunakan untuk memetakan contoh-contoh baru. Skenario optimal akan memungkinkan algoritma menentukan label kelas dengan benar untuk instance yang tidak terlihat. Ini membutuhkan algoritma pembelajaran untuk menggeneralisasi dari data pelatihan untuk situasi yang tidak terlihat dengan cara yang "masuk akal" (lihat bias induktif). Tugas paralel

dalam psikologi manusia dan hewan sering disebut sebagai pembelajaran konsep. Contoh dibawah yaitu Supervised Learning dengan SVC.

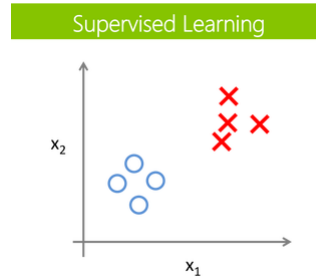


Figure 2.2: Supervised Learning

2. Unsupervised learning adalah istilah yang digunakan untuk pembelajaran bahasa Ibrani, yang terkait dengan pembelajaran tanpa guru, juga dikenal sebagai organisasi mandiri dan metode pemodelan kepadatan probabilitas input. Analisis cluster sebagai cabang pembelajaran mesin yang mengelompokkan data yang belum diberi label, diklasifikasikan atau dikategorikan. Alih-alih menangani umpan balik, analisis kluster mengidentifikasi kesamaan dalam data dan bereaksi berdasarkan ada tidaknya kesamaan di setiap potongan data baru. Berikut merupakan contoh Unsupervised Learning dengan Gaussian mixture models.

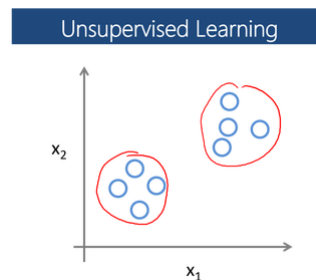


Figure 2.3: Unsupervised Learning

3. Cluster analysis or clustering adalah tugas pengelompokan sekumpulan objek sedemikian rupa sehingga objek dalam kelompok yang sama (disebut kluster) lebih mirip (dalam beberapa hal) satu sama lain daripada pada kelompok lain (kluster). Ini adalah tugas utama penambangan data eksplorasi, dan teknik umum untuk analisis data statistik, yang digunakan di banyak bidang, termasuk pembelajaran mesin, pengenalan pola, analisis gambar, pengambilan informasi,

bioinformatika, kompresi data, dan grafik komputer. Analisis Cluster sendiri bukan merupakan salah satu algoritma spesifik, tetapi tugas umum yang harus dipecahkan. Ini dapat dicapai dengan berbagai algoritma yang berbeda secara signifikan dalam pemahaman mereka tentang apa yang merupakan sebuah cluster dan bagaimana cara menemukannya secara efisien. Gagasan populer mengenai cluster termasuk kelompok dengan jarak kecil antara anggota cluster, area padat ruang data, interval atau distribusi statistik tertentu. Clustering karena itu dapat dirumuskan sebagai masalah optimasi multi-objektif. Algoritma pengelompokan dan pengaturan parameter yang sesuai (termasuk parameter seperti fungsi jarak yang akan digunakan, ambang kepadatan atau jumlah cluster yang diharapkan) tergantung pada set data individual dan penggunaan hasil yang dimaksudkan. Analisis kluster bukan merupakan tugas otomatis, tetapi proses berulang penemuan pengetahuan atau optimasi multi-objektif interaktif yang melibatkan percobaan dan kegagalan. Seringkali diperlukan untuk memodifikasi praproses data dan parameter model hingga hasilnya mencapai properti yang diinginkan.

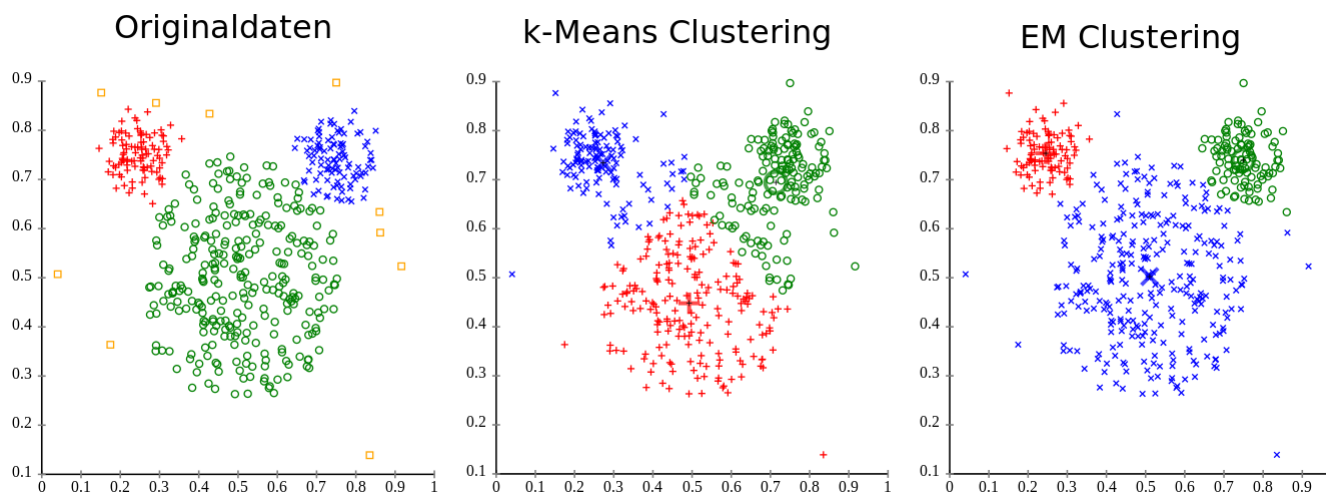
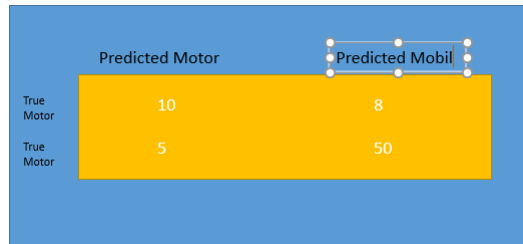


Figure 2.4: Cluster

### 2.3.3 evaluasi dan akurasi dari buku dan disertai ilustrasi contoh dengan gambar

1. Evaluasi adalah tentang bagaimana kita dapat mengevaluasi seberapa baik model bekerja dengan mengukur akurasi. Dan akurasi akan didefinisikan

sebagai persentase kasus yang diklasifikasikan dengan benar. Kita dapat menganalisis kesalahan yang dibuat oleh model, atau tingkat kebingungannya, menggunakan matriks kebingungan. Matriks kebingungan mengacu pada kebingungan dalam model, tetapi matriks kebingungan ini bisa menjadi sedikit sulit untuk dipahami ketika mereka menjadi sangat besar.



	Predicted Motor	Predicted Mobil
True Motor	10	8
True Mobil	5	50

Figure 2.5: Evaluasi dan Akurasi

### 2.3.4 bagaimana cara membuat dan membaca confusion matrix, buat confusion matrix

1. Cara membuat dan membaca confusion matrix :

- 1) Tentukan pokok permasalahan dan atributnya, misal gaji dan listik.
- 2) Buat pohon keputusan
- 3) Lalu data testingnya
- 4) Lalu mencari nilai a, b, c, dan d. Semisal a = 5, b = 1, c = 1, dan d = 3.
- 5) Selanjutnya mencari nilai recall, precision, accuracy, serta dan error rate.

2. Berikut adalah contoh dari confusion matrix :

- Recall =  $3/(1+3) = 0,75$
- Precision =  $3/(1+3) = 0,75$
- Accuracy =  $(5+3)/(5+1+1+3) = 0,8$
- Error Rate =  $(1+1)/(5+1+1+3) = 0,2$

### 2.3.5 bagaimana K-fold cross validation bekerja dengan gambar ilustrasi

1. Cara kerja K-fold cross validation :

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

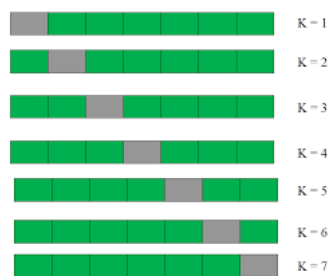


Figure 2.6: K-fold cross validation

### 2.3.6 decision tree dengan gambar ilustrasi

1. Decision Tree adalah metode pembelajaran yang diawasi non-parametrik yang digunakan untuk klasifikasi dan regresi. Tujuannya adalah untuk membuat model yang memprediksi nilai variabel target dengan mempelajari aturan keputusan sederhana yang disimpulkan dari fitur data.

Misalnya, dalam contoh di bawah ini, decision tree belajar dari data untuk memperkirakan kurva sinus dengan seperangkat aturan keputusan if-then-else. Semakin dalam pohon, semakin rumit aturan keputusan dan semakin bugar modelnya.



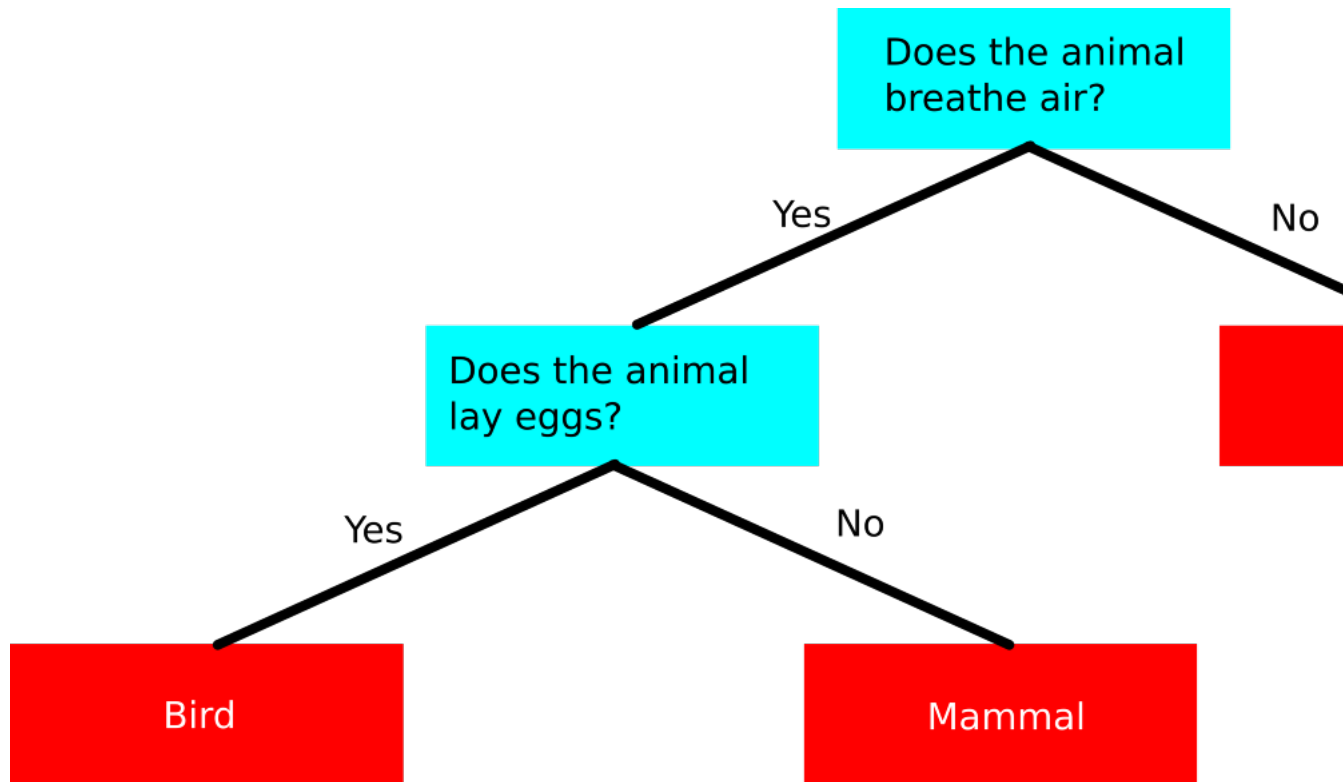


Figure 2.7: Decision Tree

### 2.3.7 Information Gain dan entropi dengan gambar ilustrasi

1. Information gain didasarkan pada penurunan entropi setelah dataset dibagi pada atribut. Membangun decision tree adalah semua tentang menemukan atribut yang mengembalikan perolehan informasi tertinggi (mis., Cabang yang paling homogen).



Figure 2.8: Information gain

2. Entropi adalah ukuran keacakan dalam informasi yang sedang diproses. Semakin tinggi entropi, semakin sulit untuk menarik kesimpulan dari informasi

itu. Membalik koin adalah contoh tindakan yang memberikan informasi yang acak. Untuk koin yang tidak memiliki afinitas untuk kepala atau ekor, hasil dari sejumlah lemparan sulit diprediksi. Mengapa? Karena tidak ada hubungan antara membalik dan hasilnya. Inilah inti dari entropi.

## 2.4 scikit-learn

HARI KEDUA ANDRI FAJAR SUNANDHAR 1164065

1. # load dataset (student Portuguese scores)

```
import pandas as pd
jeruk = pd.read_csv('E:\KAMPUS\Semester 6\Kecerdasan Buatan\modul\Python-Ar
len(jeruk)
```

Untuk mengimport atau memanggil module pandas sebagai pd. Kemudian mendefinisikan variabel "jeruk" yang akan memanggil dataset yang didapatkan dari data student-mat.csv

```
In [1]: import pandas as pd
...: d = pd.read_csv('E:\KAMPUS\Semester 6\Kecerdasan Buatan\modul
\Python-Artificial-Intelligence-Projects-for-Beginners\Chapter01\dataset
\student-mat.csv', sep=';')
...: len(d)
Out[1]: 395
```

Figure 2.9: Loading Dataset

2. # generate binary label (pass/fail) based on G1+G2+G3 (test grades, each 0-20)

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

mendeklarasikan label pass/fail nya data berdasarkan G1+G2+G3. kemudian pada variabel jeruk dideklarasikan jika baris dengan G1+G2+G3 ditambahkan, dan hasilnya sama dengan 35 maka axisnya 1.

```
In [8]: jeruk['pass'] = jeruk.apply(lambda row: 1 if (row['G1']+row['G2']+row['G3']) >= 35
else 0, axis=1)
...: jeruk = jeruk.drop(['G1', 'G2', 'G3'], axis=1)
...: jeruk.head()
Out[8]:
  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 2.10: Generate Binary Label

3. # use one-hot encoding on categorical columns

```
jeruk = apel.get_dummies(jeruk, columns=['sex', 'school', 'address', 'famsize',
                                         'reason', 'guardian', 'schoolsup', 'famsup',
                                         'nursery', 'higher', 'internet', 'romantic'])

jeruk.head()
```

One-hot encoding adalah proses di mana variabel kategorikal dikonversi menjadi bentuk yang dapat disediakan untuk algoritma .

```
In [9]: jeruk = apel.get_dummies(jeruk, columns=['sex', 'school', 'address', 'famsize',
        ...: 'Pstatus', 'Mjob', 'Fjob',
        ...: 'reason', 'guardian', 'schoolsup', 'famsup',
        ...: 'paid', 'activities',
        ...: 'nursery', 'higher', 'internet', 'romantic'])
Out[9]: jeruk.head()
age  Medu  Fedu  ...  internet_yes  romantic_no  romantic_yes
0   18     4     4  ...             0             1             0
1   17     1     1  ...             1             1             0
2   15     1     1  ...             1             1             0
3   15     4     2  ...             1             0             1
4   16     3     3  ...             0             1             0
[5 rows x 57 columns]
```

Figure 2.11: One-hot Encoding

4. # shuffle rows

```
jeruk = jeruk.sample(frac=1)
# split training and testing data
jeruk_train = jeruk[:500]
jeruk_test = jeruk[500:]
```

```
jeruk_train_att = jeruk_train.drop(['pass'], axis=1)
jeruk_train_pass = jeruk_train['pass']
```

```
jeruk_test_att = jeruk_test.drop(['pass'], axis=1)
jeruk_test_pass = jeruk_test['pass']
```

```
jeruk_att = jeruk.drop(['pass'], axis=1)
jeruk_pass = jeruk['pass']
```

# number of passing students in whole dataset:

```
import numpy as np
print("Passing: %d out of %d (%.2f%%)" % (np.sum(jeruk_pass), len(jeruk_pass)))
```

Pada bagian tersebut, terdapat train dan test yang digunakan untuk untuk membagi train, test dan kemudian membagi lagi train ke validasi dan test.

Kemudia akan mengimport module numpy sebagai np yang akan digunakan untuk mengembalikan nilai passing dari pelajar dari keseluruhan dataset dengan cara print.

```
In [10]: jeruk = jeruk.sample(frac=1)
...: # split training and testing data
...: jeruk_train = jeruk[:500]
...: jeruk_test = jeruk[500:]
...:
...: jeruk_train_att = jeruk_train.drop(['pass'], axis=1)
...: jeruk_train_pass = jeruk_train['pass']
...:
...: jeruk_test_att = jeruk_test.drop(['pass'], axis=1)
...: jeruk_test_pass = jeruk_test['pass']
...:
...: jeruk_att = jeruk.drop(['pass'], axis=1)
...: jeruk_pass = jeruk['pass']
...:
...: # number of passing students in whole dataset:
...: import numpy as np
...: print("Passing: %d out of %d (%.2f%%)" % (np.sum(jeruk_pass), len(jeruk_pass),
100*float(np.sum(jeruk_pass)) / len(jeruk_pass)))
Passing: 166 out of 395 (42.03%)
```

Figure 2.12: Shuffle Rows

#### 5. # fit a decision tree

```
from sklearn import tree
semangka = tree.DecisionTreeClassifier(criterion="entropy", max_depth=5)
semangka = semangka.fit(jeruk_train_att, jeruk_train_pass)
```

Dari librari scikitlearn import modul tree. Kemudian definisikan variabel semangka dengan menggunakan DecisionTreeClassifier. Kemudian pada variabel semangka terdapat Criterion , setelah itu agar DecisionTreeClassifier dapat dijalankan gunakan perintah fit. hasilnya seperti dibawah

```
In [11]: from sklearn import tree
...: semangka = tree.DecisionTreeClassifier(criterion="entropy", max_depth=5)
...: semangka = semangka.fit(jeruk_train_att, jeruk_train_pass)
```

Figure 2.13: Fit Decision Tree

#### 6. # visualize tree

```
import graphviz
dot_data = tree.export_graphviz(semangka, out_file=None, label="all", impurity=
                                feature_names=list(jeruk_train_att), class_names=
                                filled=True, rounded=True)

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

Mengimport Graphviz Sehingga akan muncul gambardiagram grafik bercabang.

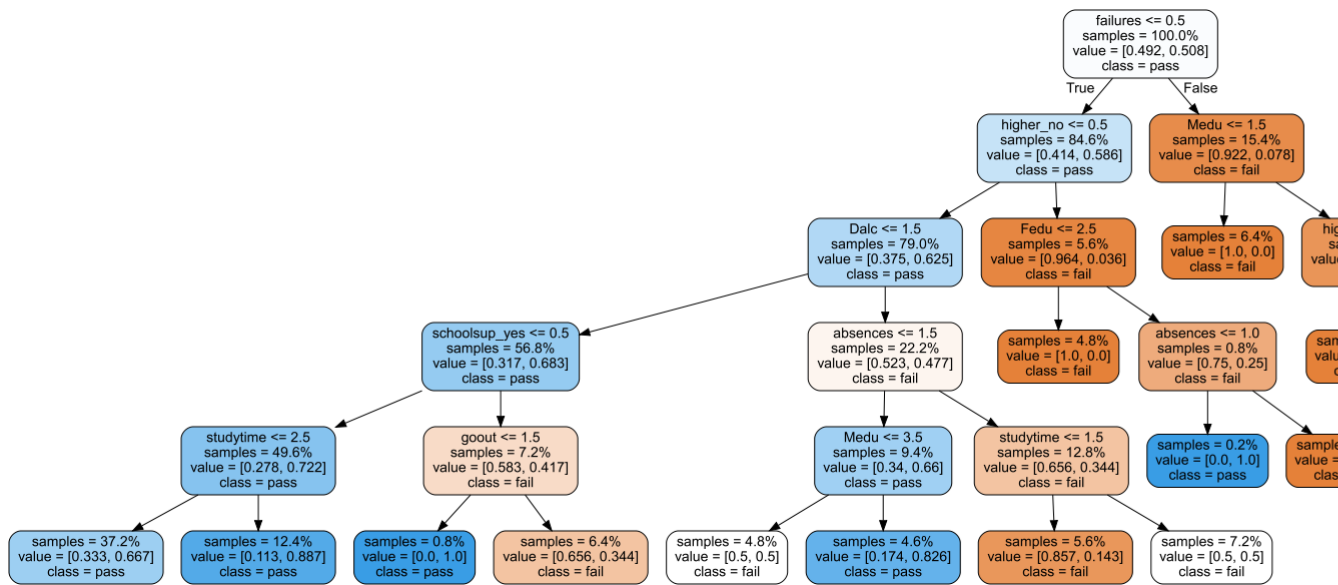


Figure 2.14: Fit Decision Tree

## 7. # save tree

```
tree.export_graphviz(semangka, out_file="student-performance.dot", label="all",
                    feature_names=list(jeruk_train_att), class_names=["fail", "pass"],
                    filled=True, rounded=True)
```

tree.exportgraphviz merupakan fungsi yang menghasilkan representasi Graphviz dari decision tree.

```
In [13]: tree.export_graphviz(semangka, out_file="student-
performance.dot", label="all", impurity=False, proportion=True,
...:                         feature_names=list(jeruk_train_att),
...:                         class_names=["fail", "pass"],
...:                         filled=True, rounded=True)
```

Figure 2.15: Fit Decision Tree

## 8. semangka.score(jeruk\_test\_att, jeruk\_test\_pass)

Score juga disebut prediksi, Nilai atau skor yang dibuat dapat mewakili prediksi nilai masa depan, tetapi mereka juga mungkin mewakili kategori atau hasil yang mungkin. disini semangka akan memprediksi jeruk.

```
In [9]: semangka.score(jeruk_test_att, jeruk_test_pass)
Out[9]: 0.6845637583892618
```

Figure 2.16: Score

```

9. from sklearn.model_selection import cross_val_score
   scores = cross_val_score(semangka, jeruk_att, jeruk_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))

```

Dari sklearn.modelselection akan mengimport crossvalscore. Kemudian akan menampilkan score rata rata dan kurang lebih dua standar deviasi yang mencakup 95 persen score.

```

In [15]: from sklearn.model_selection import cross_val_score
...: scores = cross_val_score(semangka, jeruk_att, jeruk_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: 0.58 (+/- 0.04)

```

Figure 2.17: Cross Val Score

```

10. for max_depth in range(1, 20):
    semangka = tree.DecisionTreeClassifier(criterion="entropy", max_depth=max_depth)
    scores = cross_val_score(semangka, jeruk_att, jeruk_pass, cv=5)
    print("Max depth: %d, Accuracy: %0.2f (+/- %0.2f)" % (max_depth, scores.mean(),

```

Semangka akan mendefinisikan tree.DecissionTreeClassifier nya yang kemudian variabel semangka akan mengevaluasi score dengan validasi silang.

```

In [16]: for max_depth in range(1, 20):
...:     semangka = tree.DecisionTreeClassifier(criterion="entropy", max_depth=max_depth)
...:     scores = cross_val_score(semangka, jeruk_att, jeruk_pass, cv=5)
...:     print("Max depth: %d, Accuracy: %0.2f (+/- %0.2f)" % (max_depth, scores.mean(), scores.std() * 2))
Max depth: 1, Accuracy: 0.58 (+/- 0.01)
Max depth: 2, Accuracy: 0.58 (+/- 0.05)
Max depth: 3, Accuracy: 0.58 (+/- 0.02)
Max depth: 4, Accuracy: 0.59 (+/- 0.08)
Max depth: 5, Accuracy: 0.58 (+/- 0.03)
Max depth: 6, Accuracy: 0.58 (+/- 0.08)
Max depth: 7, Accuracy: 0.57 (+/- 0.10)
Max depth: 8, Accuracy: 0.60 (+/- 0.10)
Max depth: 9, Accuracy: 0.60 (+/- 0.09)
Max depth: 10, Accuracy: 0.62 (+/- 0.05)
Max depth: 11, Accuracy: 0.61 (+/- 0.08)
Max depth: 12, Accuracy: 0.63 (+/- 0.08)
Max depth: 13, Accuracy: 0.60 (+/- 0.12)
Max depth: 14, Accuracy: 0.59 (+/- 0.07)
Max depth: 15, Accuracy: 0.62 (+/- 0.07)
Max depth: 16, Accuracy: 0.59 (+/- 0.05)
Max depth: 17, Accuracy: 0.62 (+/- 0.05)
Max depth: 18, Accuracy: 0.62 (+/- 0.09)
Max depth: 19, Accuracy: 0.62 (+/- 0.07)

```

Figure 2.18: Max Depth

```

11. depth_acc = np.empty((19,3), float)
    i = 0
    for max_depth in range(1, 20):
        semangka = tree.DecisionTreeClassifier(criterion="entropy", max_depth=max_depth)
        scores = cross_val_score(semangka, jeruk_att, jeruk_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

Dengan 19 sebagai bentuk array kosong, 3 sebagai output data-type dan float urutan kolom-utama (gaya Fortran) dalam memori. variabel semangka yang akan melakukan split score dan nangka akan mengvalidasi score secara silang.

```

In [17]: depth_acc = np.empty((19,3), float)
...: i = 0
...: for max_depth in range(1, 20):
...:     semangka = tree.DecisionTreeClassifier(criterion="entropy", max_depth=max_depth)
...:     scores = cross_val_score(semangka, jeruk_att, jeruk_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[17]:
array([[1.00000000e+00, 5.79751704e-01, 6.30768599e-03],
       [2.00000000e+00, 5.79817429e-01, 5.40865102e-02],
       [3.00000000e+00, 5.82346264e-01, 2.23551177e-02],
       [4.00000000e+00, 5.84752515e-01, 7.32490010e-02],
       [5.00000000e+00, 5.84719247e-01, 4.05630974e-02],
       [6.00000000e+00, 5.79462837e-01, 7.53158175e-02],
       [7.00000000e+00, 5.71772152e-01, 9.89792479e-02],
       [8.00000000e+00, 6.02088608e-01, 1.20171591e-01],
       [9.00000000e+00, 6.02249270e-01, 9.20901855e-02],
       [1.00000000e+01, 6.17666342e-01, 6.17102083e-02],
       [1.10000000e+01, 6.02378286e-01, 5.13911736e-02],
       [1.20000000e+01, 6.15198799e-01, 5.15762620e-02],
       [1.30000000e+01, 6.05072217e-01, 5.81263071e-02],
       [1.40000000e+01, 6.15038137e-01, 5.79996256e-02],
       [1.50000000e+01, 5.97380721e-01, 6.43910242e-02],
       [1.60000000e+01, 5.82190036e-01, 7.12462584e-02],
       [1.70000000e+01, 5.97284972e-01, 8.33497031e-02],
       [1.80000000e+01, 6.25166342e-01, 6.78027722e-02],
       [1.90000000e+01, 5.97285784e-01, 1.01208000e-01]])

```

Figure 2.19: Depth in Range

```

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

```

Mengimpor librari dari matplotlib yaitu pyplot sebagai plt fig dan ax menggunakan subplots untuk membuat gambar . ax.errorbar akan membuat error bar

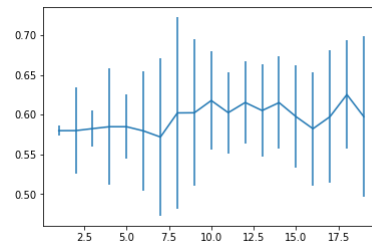


Figure 2.20: Matplotlib

## 2.5 Penanganan Error

Hari Kedua Andri fajar Sunandhar 1164065

### 2.5.1 Error Graphviz

1. error yang didapatkan saat menjalankan Graphviz

```

Variable explorer | File explorer | Help
IPython console
Console 3/A
...: 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-11-ef75b81d0d6a>", line 1, in <module>
    import graphviz
ModuleNotFoundError: No module named 'graphviz'

```

Figure 2.21: Error Graphviz

2. Kode erornya adalah ModuleNotFoundError. Error ini terjadi karena module named Graphviz nya tidak ada.
3. Solusi yang bisa dilakukan untuk mengatasi eror tersebut adalah sebagai berikut :

- buka CMD kemudian perintah pip install graphviz
- masukan perintah conda install pip, untuk solving environment
- selanjutnya masukan perintah conda install python-graphviz , untuk menambahkan package python-graphviz pada conda



```
C:\Users\ACER>pip install graphviz
Collecting graphviz
  Downloading https://files.pythonhosted.org/packages/1f/e2/ef2581b5b866256
/graphviz-0.10.1-py2.py3-none-any.whl
Installing collected packages: graphviz
Successfully installed graphviz-0.10.1
You are using pip version 18.1, however version 19.0.3 is available.
You should consider upgrading via the 'python -m pip install --upgrade pip'
```

Figure 2.22: install Graphviz

```
C:\Users\ACER>conda install pip
Collecting package metadata: done
Solving environment: done

# All requested packages already installed.
```

Figure 2.23: Solving Environment

```
C:\Users\ACER>conda install python-graphviz
Collecting package metadata: done
Solving environment: done

## Package Plan ##

  environment location: C:\Users\ACER\Anaconda3

added / updated specs:
- python-graphviz

The following packages will be downloaded:

package | build | size
-----|-----|-----
graphviz-2.38 | hfa6e2cd_3 | 37.7 MB
python-graphviz-0.8.4 | py36_1 | 28 KB
-----|-----|-----
Total: | | 37.8 MB

The following NEW packages will be INSTALLED:

graphviz | pkgs/main/win-32::graphviz-2.38-hfa6e2cd_3
python-graphviz | pkgs/main/win-32::python-graphviz-0.8.4-py36_1

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

Figure 2.24: Evaluasi Error

# Chapter 3

## Methods

### 3.1 The data

Please tell where is the data come from, a little brief of company can be put here.

### 3.2 Method 1

Definition, steps, algorithm or equation of method 1 and how to apply into your data

### 3.3 Method 2

Definition, steps, algorithm or equation of method 2 and how to apply into your data

### 3.4 Andri Fajar Sunandhar/1164065

#### 3.4.1 Teori

1. Apa itu Random Forest Serta Gambar Ilustrasinya

Random Forest adalah suatu algoritma yang digunakan pada klasifikasi data dalam jumlah yang besar. Klasifikasi random forest dilakukan melalui penggabungan pohon dengan melakukan training pada sampel data yang dimiliki. Penggunaan tree yang semakin banyak akan mempengaruhi akurasi yang akan didapatkan menjadi lebih baik. Penentuan klasifikasi dengan random forest diambil berdasarkan hasil voting dari pohon yang terbentuk. Pemenang dari pohon yang terbentuk ditentukan dengan vote terbanyak. Pembangunan pohon pada random forest sampai dengan mencapai ukuran maksimum dari pohon data. Akan tetapi, pembangunan pohon Random Forest tidak dilakukan

pemangkasan yang merupakan sebuah metode untuk mengurangi kompleksitas ruang. Contoh Ilustrasi sederhana Gambar Random Forest.

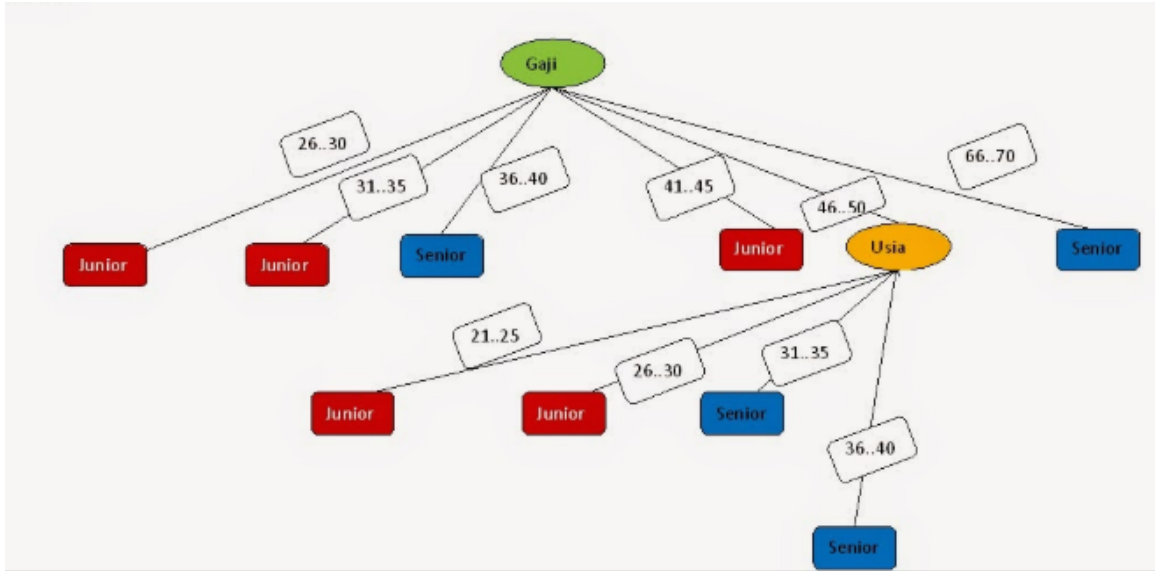


Figure 3.1: Random Forest.

## 2. Cara Membaca Dataset

- (a) Buka Anaconda Navigator.
- (b) Jalankan Spyder
- (c) Import libraries yang dibutuhkan
- (d) Masukkan kode berikut untuk membaca file Data.csv.

```
16
17 dataset = pd.read_csv('Data.csv')
18 |
```

Figure 3.2: Kode membaca file.csv

- (e) Jalankan kode tersebut, maka di windiws console akan muncul pesan :

```
In [6]: dataset = pd.read_csv('Data.csv')
In [7]:
```

Figure 3.3: Window Console

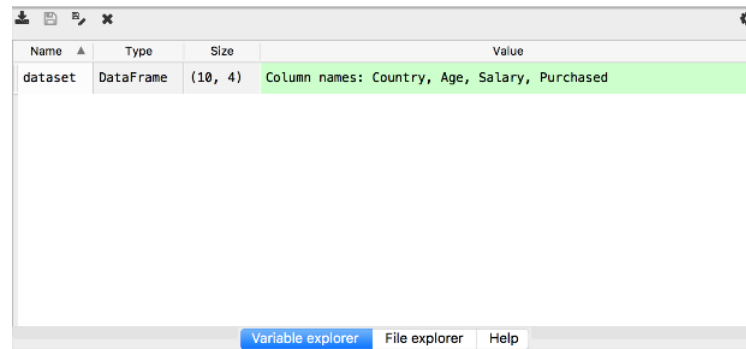


Figure 3.4: Variable Explorer

- (f) Klik variable explorer, maka akan terlihat dataset yang baru ter-import.
- (g) Kemudian double klik pada dataset cell, maka akan muncul pop-up windows seperti berikut:

Index	Country	Age	Salary	Purchased
0	France	44	72000	No
1	Spain	27	46000	Yes
2	Germany	30	54000	No
3	Spain	38	61000	No
4	Germany	40	nan	Yes
5	France	35	58000	Yes
6	Spain	nan	52000	No
7	France	48	79000	Yes
8	Germany	50	83000	No
9	France	37	67000	Yes

Figure 3.5: Dataset Cell

- (h) Seperti yang terlihat pada gambar tersebut dataset ini memiliki Kolom Country, Age, dan Salary sebagai independent variable-nya dan kolom Purchased sebagai dependent variable-nya.

### 3. Cross Validation

Cross validation adalah metode statistik yang digunakan untuk memperkirakan keterampilan model pembelajaran mesin. Ini biasanya digunakan dalam pembelajaran mesin yang diterapkan untuk membandingkan dan memilih model untuk masalah pemodelan prediktif yang diberikan karena mudah dipahami, mudah diimplementasikan, dan menghasilkan estimasi keterampilan yang umumnya memiliki bias lebih rendah daripada metode lainnya.

4. Arti Score 44% Pada Random Forest, 27% Pada Decision Tree dan 29% Dari SVM

(a) Arti Score 44%

Pada Random Forest, Score tersebut merupakan hasil dari akurasi.

(b) Arti Score 27%

Pada decision tree adalah presentasi hasil dari perhitungan dataset.

(c) Arti Score 29% Pada SVM

merupakan hasil pendekatan jaringan saraf. Jaringan saraf sendiri merupakan komponen jaringan utama dari sistem saraf. Sistem tersebut mengatur dan mengontrol fungsi tubuh dan aktivitas dan terdiri dari dua bagian: (SSP) yang terdiri dari otak dan sumsum tulang belakang, dan percabangan saraf perifer dari sistem saraf tepi (SST) yang terdapat dalam pengolahan dataset terkait.

(a) Confusion Matrix Dan Ilustrasinya

(a) Perhitungan confusion matrix adalah sebagai berikut, akan saya beri contoh sederhana yaitu pengambilan keputusan untuk mendapatkan bantuan beasiswa. Saya menggunakan dua atribut, yaitu rekening listrik dan gaji. Ini adalah pohon keputusannya:

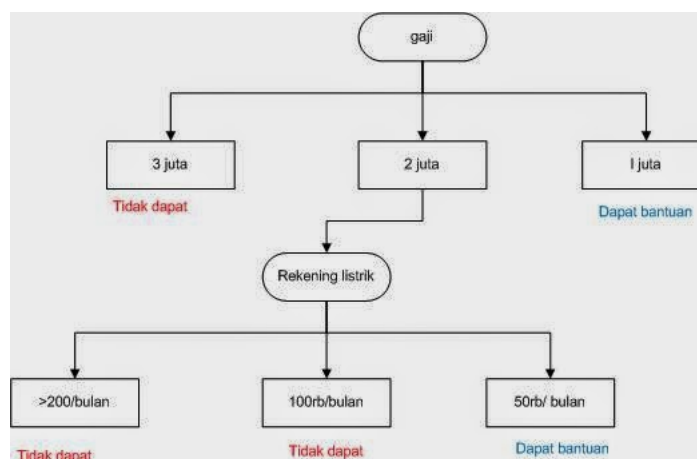


Figure 3.6: Pohon Keputusan

Kemudian data testingnya adalah

Yang pertama kita lakukan yaitu mencari 4 nilai yaitu a,b,c, dan d:

no	nama	gaji	rekening	hasil	kecocokan
1	Aji	3 juta	100rb/bulan	dapat bantu	t
2	Ali	1 juta	50rb/bulan	dapat bantu	y
3	Amar	2 juta	100rb/bulan	tidak dapat	y
4	Bastoni	1 juta	100rb/bulan	tidak dapat	y
5	Tolib	2 juta	50rb/bulan	dapat bantu	y
6	Sarip	3 juta	>200rb/bulan	tidak dapat	y
7	Tuwar	3 juta	100rb/bulan	tidak dapat	y
8	Rokip	2 juta	100rb/bulan	tidak dapat	y
9	Habib	1 juta	100rb/bulan	dapat bantu	y
10	Sohe	2 juta	50rb/bulan	tidak dapat	t

Figure 3.7: Data Testing

a= 5

b= 1

c= 1

d= 3

Kemudian kita dapat mencari nilai Recall, Precision, accuracy dan Error Rate

Recall =  $3/(1+3) = 0,75$

Precision =  $3/(1+3) = 0,75$

Accuracy =  $(5+3)/(5+1+1+3) = 0,8$

Error Rate =  $(1+1)/(5+1+1+3) = 0,2$

#### 5. Jelaskan Voting Pada Random Forest Beserta Ilustrasinya

Voting merupakan metode yang paling umum digunakan dalam random forest. Ketika classifier membuat keputusan, Anda dapat memanfaatkan yang terbaik keputusan umum dan rata-rata yang didefinisikan ke dalam bentuk "voting".

Setelah pohon terbentuk, maka akan dilakukan voting pada setiap kelas dari data sampel. Kemudian, mengkombinasikan vote dari setiap kelas kemudian diambil vote yang paling banyak. Dengan menggunakan random forest pada klasifikasi data maka, akan menghasilkan vote yang paling baik. 3.8

### 3.4.2 Praktek Program

(a) Aplikasi Sederhana Menggunakan Pandas

Penjelasan kodingan :

(a) Memanggil library.

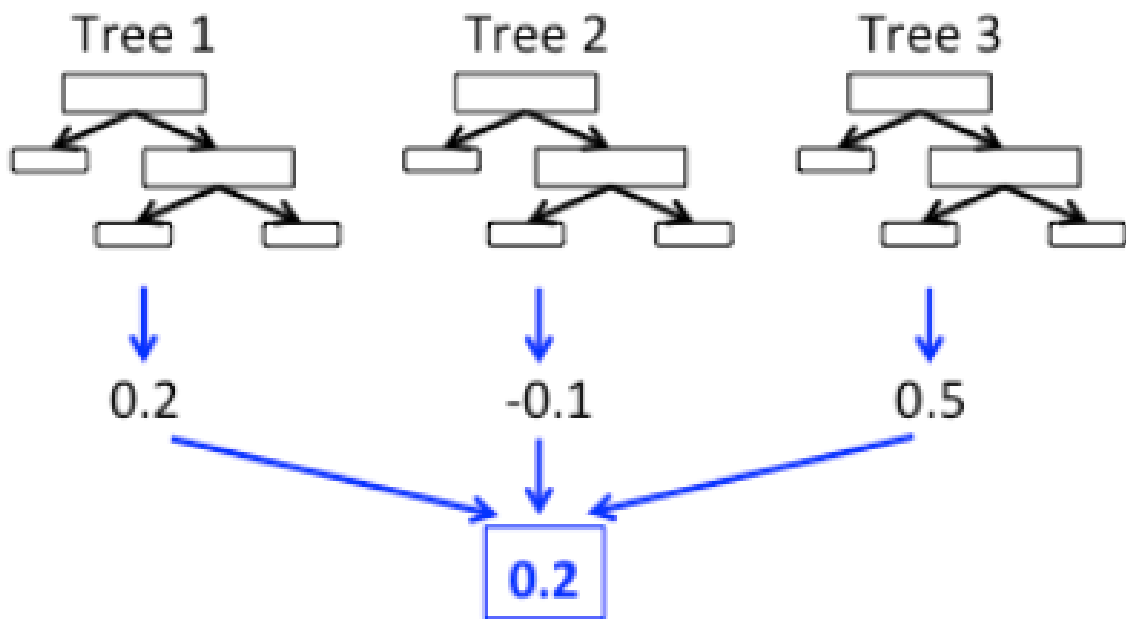


Figure 3.8: Voting.

```
import pandas as pd
df = pd.DataFrame({'X':[78,85,96,80,86], 'Y':[84,94,89,83,86], 'Z':[86,97,96,72,83]});
print(df)
```

Figure 3.9: Aplikasi Pandas

(b) Membuat variable dengan data frame.

(c) Menampilkan hasil

Sehingga menghasilkan :

```
In [44]: runfile('D:/Chapter02/aa.py', wdir='D:/Chapter02')
   X   Y   Z
0  78  84  86
1  85  94  97
2  96  89  96
3  80  83  72
4  86  86  83
```

Figure 3.10: Hasil Pandas

## 6. Aplikasi Sederhana Menggunakan Numpy

Penjelasan kodingan :

(a) Memanggil library numpy

(b) Membuat variable dengan value eye dengan size10

(c) Menampilkan hasil value

```
import numpy as Andri
matrix_one = np.eye(10)
matrix_one
```

Figure 3.11: Aplikasi Numpy

Sehingga menghasilkan :

```
Out[12]:
array([[1., 0., 0., 0., 0., 0., 0., 0., 0., 0.],
       [0., 1., 0., 0., 0., 0., 0., 0., 0., 0.],
       [0., 0., 1., 0., 0., 0., 0., 0., 0., 0.],
       [0., 0., 0., 1., 0., 0., 0., 0., 0., 0.],
       [0., 0., 0., 0., 1., 0., 0., 0., 0., 0.],
       [0., 0., 0., 0., 0., 1., 0., 0., 0., 0.],
       [0., 0., 0., 0., 0., 0., 1., 0., 0., 0.],
       [0., 0., 0., 0., 0., 0., 0., 1., 0., 0.],
       [0., 0., 0., 0., 0., 0., 0., 0., 1., 0.],
       [0., 0., 0., 0., 0., 0., 0., 0., 0., 1.]])

In [13]:
```

Figure 3.12: Hasil Numpy

## 7. Aplikasi Sederhana Menggunakan Matplotlib

```
import matplotlib.pyplot as Andri
plt.plot([10,20,30,40,50,60,70])
plt.show()
```

Figure 3.13: Aplikasi Matplotlib

Penjelasan kodingan :

- (a) Memanggil library matplotlib.pyplot
- (b) Membuat variable yang berisi 10,20,30,40,50,60,70
- (c) Membuat garis koordinat
- (d) Menampilkan hasil plt

Sehingga menghasilkan :

## 8. Program Klasifikasi Random Forest :

- Code Random Forest 1 :
- Penjelasan : Membaca dataset. Codingan di atas menghasilkan variabel baru yaitu imgatt. Terdapat 3 kolom dan 3677856 baris data.

## 9. Code Random Forest 2 :

- Penjelasan : Codingan di atas berfungsi untuk melihat sebagian data awal dari dataset. Hasilnya terdapat pada gambar di atas setelah di eksekusi.



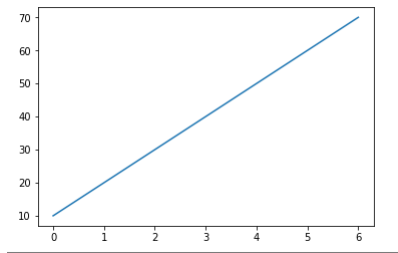


Figure 3.14: Hasil Matplotlib

```
In [30]: import pandas as pd
...:
...: # some lines have too many fields (?), so skip bad lines
...: imgatt = pd.read_csv("CUB_200_2011/attributes/image_attribute_labels.txt",
...:                      sep='\s+', header=None, error_bad_lines=False, warn_bad_lines=False,
...:                      usecols=[0,1,2], names=['imgid', 'attid', 'present'])
...:
...: # description from dataset README:
...: #
...: # The set of attribute labels as perceived by MTurkers for each image
...: # is contained in the file attributes/image_attribute_labels.txt, with
...: # each line corresponding to one image/attribute/worker triplet:
...: #
...: # <image_id> <attribute_id> <is_present> <certainty_id> <time>
...: #
...: # where <image_id>, <attribute_id>, <certainty_id> correspond to the IDs
...: # in images.txt, attributes/attributes.txt, and attributes/certainties.txt
...: # respectively. <is_present> is 0 or 1 (1 denotes that the attribute is
...: # present). <time> denotes the time spent by the MTurker in seconds.
```

Figure 3.15: Gambar1

#### 10. Code Random Forest 3 :

- Penjelasan : Codengan di atas merupakan tampilan untuk menampilkan hasil dari dataset yang telah di run atau di eksekusi. Dimana pada gambar di atas 3677856 merupakan baris dan 3 adalah kolom.

#### 11. Code Random Forest 4 :

- Penjelasan : Pada gambar di atas menmapilkan hasil dari variabel imgatt2. Dimana index nya 'imgid', kolom berisi 'attid' dan values atau nilainya berisi 'present'.

#### 12. Code Random Forest 5 :

- Penjelasan : Pada gambar di atas menmapilkan hasil dari variabel imgatt2.head. Dimana dataset nya ada 5 baris dan 312 kolom.

#### 13. Code Random Forest 6 :

```
In [31]: imgatt.head()
Out[31]:
```

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

Figure 3.16: Gambar2

```
In [32]: imgatt.shape
Out[32]: (3677856, 3)
```

Figure 3.17: Gambar3

- Penjelasan : Pada gambar di atas menampilkan jumlah dari baris dan kolom dari variabel `imgatt2`. Dimana 11788 adalah baris dan 312 adalah kolom.

#### 14. Code Random Forest 7 :

- Penjelasan : Pada gambar di atas menunjukkan load dari jawabannya yang berisi " apakah burung tersebut ( subjek pada dataset ) termasuk dalam spesies yang mana ?. Kolom yang digunakan adalah `imgid` dan label, kemudian melakukan pivot yang mana `imgid` menjadi index yang artinya unik sehubungan dengan dataset yang telah dieksekusi.

#### 15. Code Random Forest 8 :

- Penjelasan : Pada gambar di atas menunjukkan hasil dari variabel `imglabels`. Dimana menampilkan dataset dari `imgid` dan label. Dan dapat dilihat hasilnya dari gambar di atas.

#### 16. Code Random Forest 9 :

- Penjelasan : Pada gambar di atas menunjukkan jumlah baris dan kolom dari variabel `imglabels`. Dimana hasil dari kodingan tersebut dapat dilihat setelah di run.

#### 17. Code Random Forest 10 :

- Penjelasan : Pada gambar diatas dikarenakan isinya sama, maka bisa melakukan join antara dua data yang diesekusi ( yaitu ada `imgatt2` dan `imglabels` ), sehingga pada hasilnya akan didapatkan data ciri dan data

```
In [33]: imgatt2 = imgatt.pivot(index='imgid', columns='attid', values='present')
```

Figure 3.18: Gambar 4

```
In [34]: imgatt2.head()
Out[34]:
attid 1    2    3    4    5    6    7    ...  306  307  308  309  310  311  312
imgid
1      0    0    0    0    1    0    0    ...    0    0    1    0    0    0    0
2      0    0    0    0    0    0    0    ...    0    0    0    0    0    0    0
3      0    0    0    0    1    0    0    ...    0    0    1    0    0    1    0
4      0    0    0    0    1    0    0    ...    1    0    0    1    0    0    0
5      0    0    0    0    1    0    0    ...    0    0    0    0    0    0    0

[5 rows x 312 columns]
```

Figure 3.19: Gambar 5

jawaban atau labelnya sehingga bisa dikategorikan/dikelompokkan sebagai supervised learning. Jadi perintah untuk menggabungkan kedua data, kemudian dilakukan pemisahan antara data set untuk training dan test pada dataset yang dieksekusi.

18. Code Random Forest 11 :

- Penjelasan : Pada gambar di atas menghasilkan pemisahan dan pemilihan tabel ( memisahkan dan memilih tabel ).

19. Code Random Forest 12 :

- Penjelasan : Pada gambar di atas menunjukkan hasil dari variabel dtatthead. Dimana data nya dapat dilihat pada gambar diatas. Dan dataset nya terdiri dari 5 baris dan 312 kolom.

20. Code Random Forest 13 :

- Penjelasan : Pada gambar di atas menunjukkan hasil dari variabel dflabel.head. Dimana berisikan data dari imgid dan label. Dan hasilnya dapat dilihat pada gambar di atas.

21. Code Random Forest 14 :

- Penjelasan : Pada gambar di atas merupakan pembagian dari data training dan dataset

22. Code Random Forest 15 :

```
In [35]: imgatt2.shape
Out[35]: (11788, 312)
```

Figure 3.20: Gambar 6

```
In [50]: imglabels = pd.read_csv("image_class_labels.txt",
...:                             sep=' ', header=None, names=['imgid', 'label'])
...:
...:
...: imglabels = imglabels.set_index('imgid')
...:
...: # description from dataset README:
...: #
...: # The ground truth class labels (bird species labels) for each image are contained
...: # in the file image_class_labels.txt, with each line corresponding to one image:
...: #
...: # <image_id> <class_id>
...: #
...: # where <image_id> and <class_id> correspond to the IDs in images.txt and classes.txt,
...: # respectively.
```

Figure 3.21: Gambar 7

- Penjelasan : Pada gambar di atas merupakan pemanggilan kelas Random-ForestClassifier. max features yang diartikan berapa banyak kolom pada setiap tree.

#### 23. Code Random Forest 16 :

- Penjelasan : Pada gambar di atas merupakan perintah untuk melakukan fit untuk membangun random forest yang sudah ditentukan dengan maksimum fitur sebanyak 50.

#### 24. Code Random Forest 17 :

- Penjelasan : Pada gambar di atas menunjukkan hasil dari cetakan variabel dftrainatt.head.

#### 25. Code Random Forest 18 :

- Penjelasan : Pada gambar di atas merupakan hasil dari variabel dftestatt da dfsetlabel. Dimana hasilnya dapat dilihat dari pada gambar di atas

#### 26. Program Klasifikasi Confusion Matrix

- Setelah melakukan random forest kemudian dipetakan ke dalam confusion matrix.

```
In [39]: imglabels.head()
Out[39]:
```

imgid	label
0	1
1	1
2	1
3	1
4	1

Figure 3.22: Gambar 8

```
In [40]: imglabels.shape
Out[40]: (11788, 1)
```

Figure 3.23: Gambar 9

- Lalu melihat hasilnya.
- Kemudian dilakukan perintah plot.
- Selanjutnya nama data akan di set agar plot sumbunya sesuai.
- Setelah label berubah, maka dilakukan perintah plot.

27. Program Klasifikasi SVM dan Decision Tree Beserta Penjelasan Keluarannya :

- Code SVM :
- Penjelasan : Pada gambar di atas cara untuk mencoba klasi kasi dengan SVM dengan dataset yang sama.

28. Code Decision Tree :

- Penjelasan : Pada gambar di atas merupakan cara untuk mencoba klasi kasi dengan decission tree dengan dataset yang sama.

29. Program Cross Validation

- Melakukan pengecekan cross validation untuk random forest.
- Melakukan pengecekan cross validation untuk decission tree.
- Melakukan pengecekan cross validation untuk SVM.

30. Program Pengamatan Komponen Informasi

- Melakukan pengamatan komponen informasi untuk mengetahui berapa banyak tree yang dibuat, atribut yang dipakai, dan informasi lainnya.
- Melakukan plot informasi agar bisa dibaca.

```
In [41]: df = imgatt2.join(imglabels)
...: df = df.sample(frac=1)
```

Figure 3.24: Gambar 10

```
In [43]: df_att = df.iloc[:, :312]
...: df_label = df.iloc[:, 312:]
```

Figure 3.25: Gambar 11

### 3.4.3 Penanganan Error

Penyelesaian Tugas Harian ( Penanganan Error )

1. Menyelesaikan dan Membahas Penanganan Error :
  - Skreensut Error
  - Kode Error: file b'data/CUB 200 2011/attributes/image attributes labels.txt'
  - Solusi Pemecahan Error : Hapus Direktori data pada kode pastikan satu folder.

```
In [44]: df_att.head()
Out[44]:
```

	1	2	3	4	5	6	7	...	306	307	308	309	310	311	312
imgid								...							
10779	0	0	0	0	0	0	1	...	1	0	0	0	0	0	1
9334	0	0	0	0	0	0	1	...	1	0	0	0	0	1	0
10372	0	0	0	0	0	0	1	...	0	0	0	1	0	0	0
1554	1	0	0	0	0	0	0	...	1	0	0	0	1	0	0
378	0	0	0	0	0	0	1	...	0	0	0	1	0	0	0

[5 rows x 312 columns]

Figure 3.26: Gambar 12

```
In [45]: df_label.head()
Out[45]:
```

	label
imgid	
10779	183
9334	159
10372	177
1554	28
378	8

Figure 3.27: Gambar 13

```
In [46]: df_train_att = df_att[:8000]
...: df_train_label = df_label[:8000]
...: df_test_att = df_att[8000:]
...: df_test_label = df_label[8000:]
...:
...: df_train_label = df_train_label['label']
...: df_test_label = df_test_label['label']
```

Figure 3.28: Gambar 14

```
In [47]: from sklearn.ensemble import RandomForestClassifier
...: clf = RandomForestClassifier(max_features=50, random_state=0, n_estimators=100)
```

Figure 3.29: Gambar 15

```
In [48]: clf.fit(df_train_att, df_train_label)

Out[48]:
RandomForestClassifier(bootstrap=True, class_weight=None, criterion='gini',
    max_depth=None, max_features=50, max_leaf_nodes=None,
    min_impurity_decrease=0.0, min_impurity_split=None,
    min_samples_leaf=1, min_samples_split=2,
    min_weight_fraction_leaf=0.0, n_estimators=100, n_jobs=None,
    oob_score=False, random_state=0, verbose=0, warm_start=False)
Out[49]:
RandomForestClassifier(bootstrap=True, class_weight=None, criterion='gini',
    max_depth=None, max_features=50, max_leaf_nodes=None,
    min_impurity_decrease=0.0, min_impurity_split=None,
    min_samples_leaf=1, min_samples_split=2,
    min_weight_fraction_leaf=0.0, n_estimators=100, n_jobs=None,
    oob_score=False, random_state=0, verbose=0, warm_start=False)
```

Figure 3.30: Gambar 16

```
In [50]: print(clf.predict(df_train_att.head()))
[183 159 177 28 8]
```

Figure 3.31: Gambar 17

```
In [51]: clf.score(df_test_att, df_test_label)
Out[51]: 0.44667370644139387
```

Figure 3.32: Gambar 18

```
In [30]: from sklearn.metrics import confusion_matrix
...: pred_labels = clf.predict(df_test_att)
...: cm = confusion_matrix(df_test_label, pred_labels)
```

Figure 3.33: Memetakan ke confusion matrix

```
In [31]: cm
Out[31]:
array([[ 7,  0,  4, ...,  0,  1,  0],
       [ 0, 12,  0, ...,  0,  0,  0],
       [ 1,  0, 11, ...,  0,  0,  0],
       ...,
       [ 0,  0,  1, ...,  4,  0,  0],
       [ 0,  0,  0, ...,  0,  8,  0],
       [ 0,  0,  0, ...,  0,  0, 15]], dtype=int64)
```

Figure 3.34: Melihat hasil

```
In [32]: import matplotlib.pyplot as plt
...: import itertools
...: def plot_confusion_matrix(cm, classes,
...:                          normalize=False,
...:                          title='Confusion matrix',
...:                          cmap=plt.cm.Blues):
...:     """
...:     This function prints and plots the confusion matrix.
...:     Normalization can be applied by setting `normalize=True`.
...:     """
...:     if normalize:
...:         cm = cm.astype('float') / cm.sum(axis=1)[:, np.newaxis]
...:         print("Normalized confusion matrix")
...:     else:
...:         print('Confusion matrix, without normalization')
...:
...:     print(cm)
...:
...:     plt.imshow(cm, interpolation='nearest', cmap=cmap)
...:     plt.title(title)
...:     #plt.colorbar()
...:     tick_marks = np.arange(len(classes))
...:     plt.xticks(tick_marks, classes, rotation=90)
...:     plt.yticks(tick_marks, classes)
...:
...:     fmt = '.2f' if normalize else 'd'
...:     thresh = cm.max() / 2.
...:     #for i, j in itertools.product(range(cm.shape[0]), range(cm.shape[1])):
```

Figure 3.35: Melakukan Plot

```
In [33]: birds = pd.read_csv("CUB_200_2011/classes.txt",
...:                        sep='\s+', header=None, usecols=[1],
...:                        names=['birdname'])
...: birds = birds['birdname']
...: birds

Out[33]:
0      001.Black_footed_Albatross
1      002.Laysan_Albatross
2      003.Sooty_Albatross
3      004.Groove_billed_Ani
4      005.Crested_Auklet
5      006.Least_Auklet
6      007.Parakeet_Auklet
7      008.Rhinoceros_Auklet
8      009.Brewer_Blackbird
9      010.Red_winged_Blackbird
10     011.Rusty_Blackbird
11     012.Yellow_headed_Blackbird
12     013.Bobolink
13     014.Indigo_Bunting
14     015.Lazuli_Bunting
15     016.Painted_Bunting
16     017.Cardinal
17     018.Spotted_Catbird
18     019.Gray_Catbird
```

Figure 3.36: Plotting nama data



```
In [34]: 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()
Normalized confusion matrix
[[0.35 0.    0.2  ... 0.    0.05 0. ]
 [0.    0.63 0.    ... 0.    0.    0. ]
 [0.04 0.    0.46 ... 0.    0.    0. ]
 ...
 [0.    0.    0.05 ... 0.21 0.    0. ]
 [0.    0.    0.    ... 0.    0.38 0. ]
 [0.    0.    0.    ... 0.    0.    0.88]]
```

Figure 3.37: Melakukan perintah plot

```
In [45]: from sklearn import svm
...: clfsvm = svm.SVC()
...: clfsvm.fit(df_train_att, df_train_label)
...: clfsvm.score(df_test_att, df_test_label)
C:\ProgramData\Anaconda3\lib\site-packages\sklearn\svm\base.py:196: FutureWarning:
The default value of gamma will change from 'auto' to 'scale' in version 0.22 to
account better for unscaled features. Set gamma explicitly to 'auto' or 'scale' to
avoid this warning.
  "avoid this warning.", FutureWarning)
Out[45]: 0.2682154171066526
```

Figure 3.38: SVM

```
In [44]: from sklearn import tree
...: clftree = tree.DecisionTreeClassifier()
...: clftree.fit(df_train_att, df_train_label)
...: clftree.score(df_test_att, df_test_label)
Out[44]: 0.2626715945089757
```

Figure 3.39: Decission Tree

```
In [39]: from sklearn.model_selection import cross_val_score
...: scores = cross_val_score(clf, df_train_att, df_train_label, cv=5)
...: # show average score and +/- two standard deviations away (covering 95%
of scores)
...: print("Accuracy: %0.2f (+/- %0.2f)" % (scores.mean(), scores.std() * 2))
Accuracy: 0.44 (+/- 0.03)
```

Figure 3.40: Pengecekan cross validation random forest

```
In [40]: scoretree = cross_val_score(clftree, df_train_att, df_train_label,
cv=5)
...: print("Accuracy: %0.2f (+/- %0.2f)" % (scoretree.mean(),
scoretree.std() * 2))
Accuracy: 0.27 (+/- 0.02)
```

Figure 3.41: Pengecekan cross validation decision tree

```
In [42]: max_features_opts = range(5, 50, 5)
...: n_estimators_opts = range(10, 200, 20)
...: rf_params = np.empty((len(max_features_opts)*len(n_estimators_opts),4),
float)
...: i = 0
...: for max_features in max_features_opts:
...:     for n_estimators in n_estimators_opts:
...:         clf = RandomForestClassifier(max_features=max_features,
n_estimators=n_estimators)
...:         scores = cross_val_score(clf, df_train_att, df_train_label,
cv=5)
...:         rf_params[i,0] = max_features
...:         rf_params[i,1] = n_estimators
...:         rf_params[i,2] = scores.mean()
...:         rf_params[i,3] = scores.std() * 2
...:         i += 1
...:     print("Max features: %d, num estimators: %d, accuracy: %0.2f
(+/- %0.2f)" %
(max_features, n_estimators, scores.mean(),
scores.std() * 2))
Max features: 5, num estimators: 10, accuracy: 0.25 (+/- 0.02)
Max features: 5, num estimators: 30, accuracy: 0.36 (+/- 0.02)
Max features: 5, num estimators: 50, accuracy: 0.39 (+/- 0.01)
Max features: 5, num estimators: 70, accuracy: 0.41 (+/- 0.01)
Max features: 5, num estimators: 90, accuracy: 0.42 (+/- 0.01)
Max features: 5, num estimators: 110, accuracy: 0.43 (+/- 0.02)
Max features: 5, num estimators: 130, accuracy: 0.44 (+/- 0.01)
Max features: 5, num estimators: 150, accuracy: 0.44 (+/- 0.02)
Max features: 5, num estimators: 170, accuracy: 0.44 (+/- 0.02)
```

Figure 3.42: Pengamatan Komponen

```
In [43]: import matplotlib.pyplot as plt
...: from mpl_toolkits.mplot3d import Axes3D
...: from matplotlib import cm
...: fig = plt.figure()
...: fig.clf()
...: ax = fig.gca(projection='3d')
...: x = rf_params[:,0]
...: y = rf_params[:,1]
...: z = rf_params[:,2]
...: ax.scatter(x, y, z)
...: ax.set_xlim(0.2, 0.5)
...: ax.set_xlabel('Max features')
...: ax.set_ylabel('Num estimators')
...: ax.set_zlabel('Avg accuracy')
...: plt.show()
```

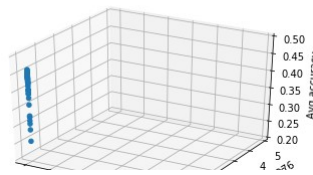


Figure 3.43: Plot informasi

```
parser_f
    return _read(filepath_or_buffer, kwds)

File "C:\ProgramData\Anaconda3\lib\site-packages\pandas\io\parsers.py", line 440, in
_read
    parser = TextFileReader(filepath_or_buffer, **kwds)

File "C:\ProgramData\Anaconda3\lib\site-packages\pandas\io\parsers.py", line 787, in
__init__
    self._make_engine(self.engine)

File "C:\ProgramData\Anaconda3\lib\site-packages\pandas\io\parsers.py", line 1014, in
_make_engine
    self._engine = CParserWrapper(self.f, **self.options)

File "C:\ProgramData\Anaconda3\lib\site-packages\pandas\io\parsers.py", line 1708, in
__init__
    self._reader = parsers.TextReader(src, **kwds)

File "pandas\_libs\parsers.pyx", line 384, in
pandas._libs.parsers.TextReader._cinit_

File "pandas\_libs\parsers.pyx", line 695, in
pandas._libs.parsers.TextReader._setup_parser_source

FileNotFoundError: File b'data\CUB_200_2011\attributes\image_attribute_labels.txt' does
not exist
```

Figure 3.44: Error

# 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

### 4.3 Andri Fajar Sunandhar/1164065

#### 4.3.1 Teori

1. Klasifikasi teks

Klasifikasi Teks adalah salah satu tugas penting dan tipikal dalam supervised machine learning (ML). Teks dapat menjadi sumber informasi yang sangat kaya, tetapi mengekstraksi wawasan darinya bisa sulit dan memakan waktu karena sifatnya yang tidak terstruktur.

2. Mengapa Klasifikasi Bunga tidak dapat menggunakan machine learning

Dikarenakan tidak semua bunga memiliki ciri - ciri yang sama. Atau dalam kata lain terdapat data noise dalam klasifikasi bunga sehingga tidak bisa menggunakan machine learning.

3. Teknik pembelajaran mesin pada teks YouTube



Figure 4.1: Klasifikasi teks



Figure 4.2: Klasifikasi bunga

Kita ambil sebuah kasus yang semua orang telah ketahui dan juga pahami. Kasus tersebut yaitu perekomendasi video dari pencarian menggunakan "text / kata" di Youtube. Pada saat menggunakan Youtube terdapat Mchine Learning yang bekerja dan memproses perintah ataupun aktivitas tersebut, dimana akan memfilter secara otomatis video yang disesuaikan dengan "keyword" yang kita masukkan sehingga memberikan keluaran video dengan keyword yang benar. Adapula pada saat kita sedang menonton video di YouTube, pada bagian sebelah kanan ( tampilan Youtube ) terdapat 'Up Next' yang menampilkan beberapa video serupa yang sedang ditonton. Dan ketika mengklik salah satu video dari baris tersebut, maka YouTube akan mengingatnya dan menggunakan kata yang tertera sebagai referensi kembali sehingga akan memberikn kemudahan pada pencarian yang lainnya, Dan disitulah mesin belajar sendiri dan menyimpan data secara berkala sehingga berkembang.



Figure 4.3: Teknik YouTube

#### 4. Vectorisasi Data

- Pembagian dan pemecahan data, dan kemudian dilakukan perhitungan datanya. Vektorisasi juga dapat dimaksudkan dengan setiap data yang mungkin dipetakan ke integer tertentu. Yang mana data tersebut dalam bentuk data vektor diperoleh dalam bentuk koordinat titik yang menampilkan, menempatkan dan menyimpan data spasial dengan menggunakan titik, garis atau area (poligon).

#### 5. Bag of word

Bag-of-words ialah sebuah gambaran sederhana digunakan dalam pengolahan bahasa alami dan pencarian informasi. Dikenal sebagai model ruang vektor. Pada model ini, tiap kalimat dalam dokumen digambarkan sebagai token, mengabaikan tata bahasa dan bahkan urutan kata namun menghitung frekuensi kejadian atau kemunculan kata dari dokumen.

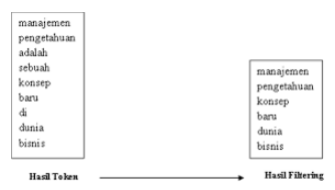


Figure 4.4: Bag of Word

#### 6. TF-IDF

TF-IDF atau TFIDF, adalah kependekan dari istilah frekuensi dokumen terbalik, dimana merupakan statistik numerik yang dimaksudkan untuk mencerminkan betapa pentingnya sebuah kata untuk sebuah dokumen dalam kumpulan atau kumpulan. Nilai tf-idf meningkat secara proporsional dengan berapa kali sebuah kata muncul dalam dokumen dan diimbangi dengan jumlah dokumen dalam korpus yang mengandung kata, yang membantu menyesuaikan fakta bahwa beberapa kata muncul lebih sering secara umum.

### 4.3.2 Praktek Program

#### 1. Aplikasi sederhana menggunakan pandas

Berikut adalah contoh aplikasi sederhana yang dibuat menggunakan pandas :

- (a) 1 = memanggil library pandas sebagai pd

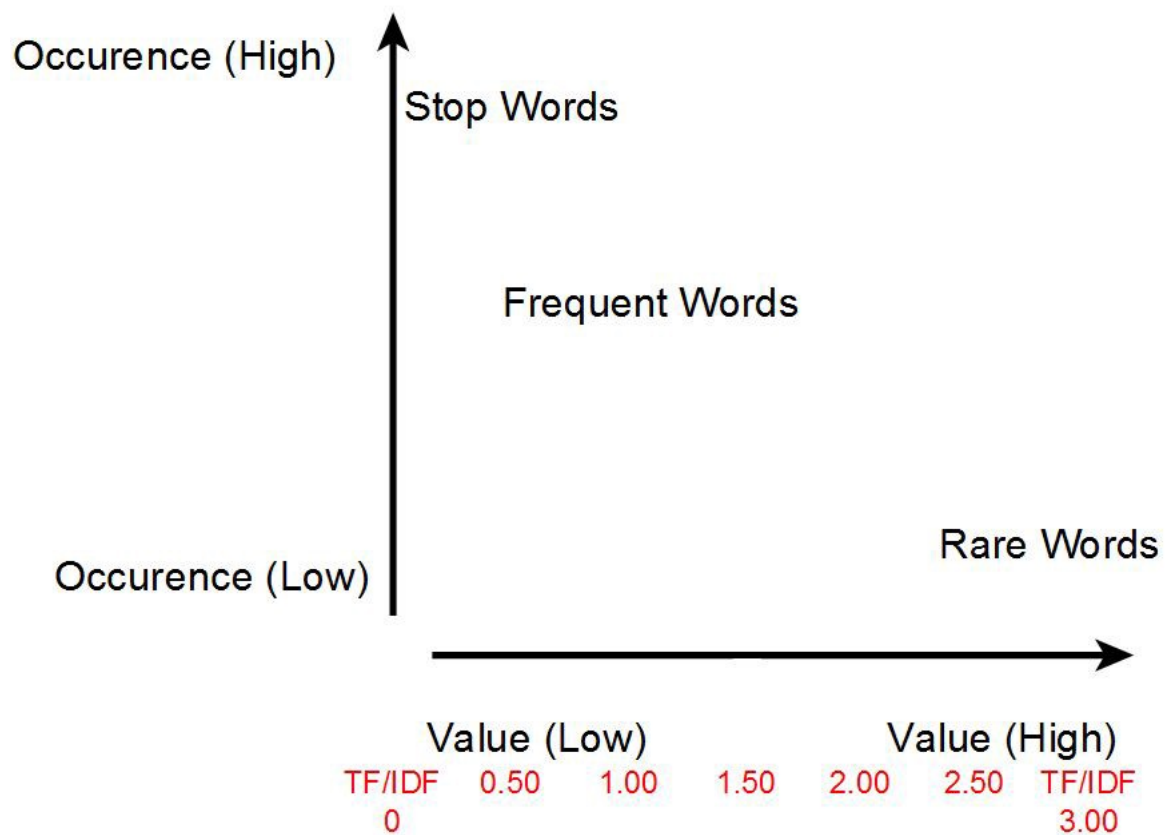


Figure 4.5: TF IDF

```
# In[2] nomor 1
import pandas as pd
afs1=pd.read_csv("E:/KAMPUS/Semester 6/Kecerdasan Buatan/bank.csv")
```

Figure 4.6: Pandas

(b) 2 = membuat variable afs1 untuk membaca file .csv (bank.csv)

Hasil dari pandas menampilkan data dari bank.csv :

Name	Type	Size	Value
afs1	DataFrame	(501, 1)	Column names: age;job;marital;education;defau...

Figure 4.7: Hasil Pandas

## 2. Memecah dataframe menjadi 2 dataframe

Memecah dataframe :

(a) 1 = Melakukan split data training sebanyak 450

```

1 # In[2] nomor2
2 as_train = afs1[:450]
3 as_test = afs1[450:]

```

Figure 4.8: Memecah dataframe

(b) 2 = dan sisanya sebagai data testing

Berikut hasil memecah dataframe menjadi 2 :

as_test	DataFrame	(51, 1)	Column names: age;job;marital;education;d...
as_train	DataFrame	(450, 1)	Column names: age;job;marital;education;d...

Figure 4.9: Hasil memecah dataframe

### 3. Vektorisasi dan klasifikasi Decision Tree dari data Youtube03-LMFAO.csv

Berikut adalah vektorisasi dan klasifikasi dari data Youtube03-LMFAO.csv

```

1 # In[1]: melakukan import pandas dan membaca file csv
2 import pandas as pd
3 afs=pd.read_csv("E:/KAMPUS/Semester 6/Kecerdasan Buatan/Youtube03-LMFAO.csv")
4
5
6 # In[2]: mengelompokkan kata spam dan bukan spam
7 spam=afs.query('CLASS == 1')
8 nospam=afs.query('CLASS == 0')
9
10
11 # In[3]: memanggil lib vektorisasi
12 #melakukan fungsi bag of word (menghitung kata yang muncul per kalimat)
13 from sklearn.feature_extraction.text import CountVectorizer
14 vectorizer = CountVectorizer()
15
16
17 # In[3]: memilih kolom CONTENT untuk dilakukan vektorisasi
18 #melakukan bag of word pada dataframe yang ada pada kolom CONTENT
19 dvec = vectorizer.fit_transform(afs['CONTENT'])
20
21
22 # In[4]: melihat isi vektorisasi di dvec
23 dvec
24
25
26 # In[5]: melihat isi data
27 print(afs['CONTENT'][349])
28
29
30 # In[6]: melihat daftar kata yang di vektorisasi
31 #feature_names merupakan digunakan untuk mengambil nama kolomnya ada apa saja
32 dk=vectorizer.get_feature_names()
33
34
35 # In[7]: akan melakukan pengocokan pada database nya supaya sempurna saat melakukan klasifikasi
36 andri = afs.sample(frac=1)

```

Figure 4.10: Vektorisasi dan klasifikasi

- (a) 1 = Melakukan import pandas dan membaca file Youtube03-LMFAO.csv
- (b) 2 = Mengelompokkan data spam bukan spam
- (c) 3 = Memanggil library vektorisasi dan menghitung kata yang muncul per kalimat
- (d) 4 = Memilih kolom CONTENT untuk melakukan vektorisasi
- (e) 5 = Melihat isi vektorisasi

- (f) 6 = Melihat isi data pada kolom CONTENT namun pada bagian kolom ke 349
- (g) 7 = Melihat daftar kata yang di vektorisasi
- (h) 8 = Akan melakukan pengocokan pada data nya supaya hasilnya sempurna ketika melakukan klasifikasi

Berikut adalah Decission Tree Youtube03-LMFAO.csv

```
In [72]: from sklearn import tree
...: clftree = tree.DecisionTreeClassifier()
...: clftree.fit(afs_train_att, afs_train_label)
...: clftree.score(afs_test_att, afs_test_label)
Out[72]: 0.9492753623188406
```

Figure 4.11: Decission Tree

Dalam gambar Decission Tree dijelaskan bahwa library tree dari sklearn. Dan mendefinisikan variable untuk memanggil Decission Tree Classifisier yang kemudian dilakukan fit atau pengujian terhadap data training.dan untuk menghitung score dari data testing. Yang menghasilkan outputan sebanyak 0.9492753623188406

#### 4. Klasifikasikan dari data vektorisasi dengan klasifikasi SVM

Berikut adalah klasifikasikan dari data vektorisasi dengan klasifikasi SVM

```
In [73]: from sklearn import svm
...: clfsvm = svm.SVC()
...: clfsvm.fit(afs_train_att, afs_train_label)
...: clfsvm.score(afs_test_att, afs_test_label)
Out[73]: 0.5652173913043478
```

Figure 4.12: Hasil klasifikasi SVM

Dalam gambar SVM dijelaskan bahwa svm mengimport library dari sklearn kemudian membuat variable clfsvm, fit tersebut membaca data training atribut dan data training label, score membaca data testing attribute dan data testing label sehingga menghasilkan outputan 0.5652173913043478

#### 5. Klasifikasikan dari data vektorisasi dengan klasifikasi Decission Tree

Maksud dari gambar vektorisasi adalah hasil dari impor dataset

Berikut adalah Decission Tree

Dalam gambar Decission Tree dijelaskan bahwa library tree dari sklearn. Dan mendefinisikan variable untuk memanggil Decission Tree Classifisier yang kemudian dilakukan fit atau pengujian terhadap data training.dan untuk menghitung



```
In [74]: from sklearn import tree
...: clftree = tree.DecisionTreeClassifier()
...: clftree.fit(afs_train_att, afs_train_label)
...: clftree.score(afs_test_att, afs_test_label)
Out[74]: 0.9492753623188406
```

Figure 4.13: Decission Tree

score dari data testing. Yang menghasilkan outputan sebanyak 0.9492753623188406

.

## 6. Plot confusion matrix menggunakan matplotlib

hasil dari plotting confusion matrix :

```
In [76]: import numpy as np
...: np.set_printoptions(precision=2)
...: plot_confusion_matrix(cm, classes=afs, normalize=True)
...: plt.show()
Normalized confusion matrix
[[0.95 0.05]
 [0.11 0.89]]
```

Figure 4.14: plotting confusion matrix

Dari gambar dijelaskan mengimport library numpy sebagai np, kemudian menampilkan precision 2 dan melakukan plot confusion matrix dari classes afs dan kemudian akan melakukan normalisasi. sehingga hasil normalisasi seperti pada gambar tersebut.

## 7. Program cross validation

Berikut adalah hasil dari program cross validation

```
In [77]: from sklearn.model_selection import cross_val_score
...: scores = cross_val_score(clf, afs_train_att, afs_train_label,
cv=5)
...: # show average score and +/- two standard deviations away
(covering 95% of scores)
...: print("Accuracy: %0.2f (+/- %0.2f)" % (scores.mean(),
scores.std() * 2))
Accuracy: 0.94 (+/- 0.03)
```

Figure 4.15: Program cross validation

Daru gambar tersebut dijelaskan cara menghitung scores dari cross validation data training attribute dan data training label kemudian dikali 2 sehingga menghasilkan akurasi 0.94.

## 8. Program pengamatan komponen informasi

Hasil dari program pengamatan komponen informasi

Gambar tersebut adalah diagram informasi dari dataset yang digunakan.

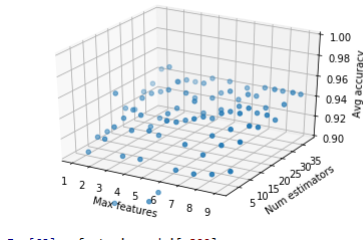


Figure 4.16: Program pengamatan komponen informasi

### 4.3.3 Penanganan Error

#### 1. skrinshot error

```
return self._engine.get_loc(self._maybe_cast_indexer(key))

File "pandas\libs\index.pyx", line 140, in
pandas._libs.index.IndexEngine.get_loc

File "pandas\libs\index.pyx", line 162, in
pandas._libs.index.IndexEngine.get_loc

File "pandas\libs\hashtable_class_helper.pxi", line 1492, in
pandas._libs.hashtable.PyObjectHashTable.get_item

File "pandas\libs\hashtable_class_helper.pxi", line 1500, in
pandas._libs.hashtable.PyObjectHashTable.get_item

KeyError: 'NAME'
```

Figure 4.17: skrinshot error

#### 2. Tuliskan kode error dan jenis errornya

- Kode error = `KeyError: 'NAME'`
- Jenis error = `KeyError`

#### 3. Solusi pemecahan masalah error

Solusinya adalah mengganti nama field `NAME` dengan `CONTENT`, dikarenakan didalam data tersebut tidak ada field `NAME`. Kemudian akan menampilkan data `CONTENT` hanya pada baris ke 349.

```
In [86]: print(afs['CONTENT'][349])
want a sub? tell me about your channel and i will subscribe (with a
couple exceptions)
```

Figure 4.18: Solusi error

# 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.

### 5.5 Andri Fajar Sunandhar / 1164065

#### 5.5.1 Teori

1. Jelaskan kenapa kata-kata harus dilakukan vektorisasi. Dilengkapi dengan ilustrasi atau Gambar

Karena mesin hanya mampu membaca data dengan bentuk angka. Berdasarkan hal tersebut maka tentunya diperlukan vektorisasi kata atau bisa disebut dengan mengubah kata menjadi bentuk vektor agar mesin seolah-olah paham apa

yang kita maksudkan dan dapat memproses aktifitas/perintah dengan benar. Kata juga harus di vektorisas iuntuk mengetahui presentase kata yang sering muncul dalam setiap kalimatnya, yang berguna untuk menentukan kata kunci. Ilustrasinya bisa dilihat pada gambar berikut 5.1.

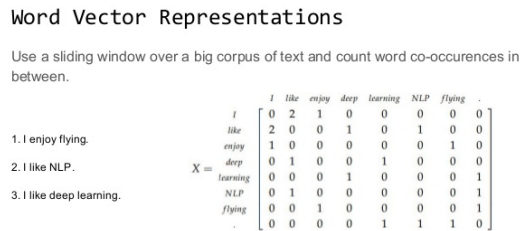


Figure 5.1: Gambar Vektorisasi Kata.

2. Jelaskan mengapa dimensi dari vektor dataset google bisa sampai 300. Dilengkapi dengan ilustrasi atau Gambar

Masing-masing nilai dalam vektor 300 dimensi yang terkait dalam sebuah kata ”dioptimalkan” dalam beberapa hal untuk menangkap aspek yang berbeda dari makna dan penggunaan kata itu. Dengan kata lain masing-masing dari 300 nilai sesuai dengan beberapa fitur abstrak kata. Menghapus kombinasi nilai-nilai ini secara acak akan menghasilkan vektor yang mungkin kurang informasi penting tentang kata tersebut dan mungkin tidak lagi berfungsi sebagai representasi yang baik dari kata itu. Atau singkat cerita mungkin ada lebih dari 3 miliar kata-kata dan kalimat atau data yang tidak mungkin disimpan dalam 1 dimensi vektor maka disimpan menjadi 300 dimensi vektor untuk mengurangi kegagalan memori. Ilustrasinya bisa dilihat pada gambar berikut 5.2.

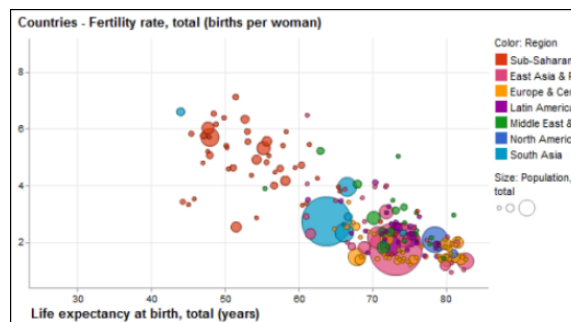


Figure 5.2: Gambar Vektorisasi Dataset Google.

3. Jelaskan konsep vektorisasi untuk kata. Dilengkapi dengan ilustrasi atau Gambar

Konsep untuk vektorisasi kata sebenarnya sama dengan ketika dilakukan input suatu kata pada mesin pencarian. Kemudian untuk hasilnya akan mengeluarkan ( berupa ) referensi mengenai kata tersebut. Jadi data kata tersebut didapatkan dari hasil pengolahan pada kalimat-kalimat sebelumnya yang telah diolah. Contoh sederhananya pada kalimat berikut ( Please click the alarm icon for more notifications about my channel ), pada kalimat tersebut terdapat konteks yakni channel, kata tersebut akan dijadikan data latih untuk mesin yang akan dipelajari dan diproses. Jadi ketika kita inputkan kta channel, maka mesin akan menampilkan keterkaitannya dengan kata tersebut sehingga akan lebih efisien dan lebih mudah. Ilustrasinya bisa dilihat pada gambar berikut 5.3.

			TEKS									
			b	a	c	a	b	b	b	b	a	c
Posisi			1	2	3	4	5	6	7	8	9	10
		0	0	0	0	0	0	0	0	0	0	0
P	b	1	0	1	1	1	0	0	0	0	1	1
O	b	2	1	1	2	2	1	0	0	0	1	2
L	b	3	2	2	2	3	2	1	0	0	1	2
A	a	4	3	2	3	2	3	2	1	1	0	1

Figure 5.3: Gambari Vektorisasi Kata.

4. Jelaskan konsep vektorisasi untuk dokumen. Dilengkapi dengan ilustrasi atau Gambar

Vektorisasi Dokumen sebenarnya terbilang sama dengan konsep vektorisasi kata, hanya yang membedakan pada proses awalnya ( pada eksekusi awal ). Untuk vektorisasi dokumen ini, mesin akan membaca semua kalimat yang terdapat pada dokumen tersebut, kemudian kalimat yang terdapat pada dokumen tersebut akan di pecah menjadi kata-kata. Ilustrasinya bisa dilihat pada gambar berikut 5.4.

5. Jelaskan apa mean dan standar deviasi. Dilengkapi dengan ilustrasi atau Gambar

Mean adalah teknik penjelasan kelompok yang didasarkan atas nilai rata-rata dari kelompok tersebut. Rata-Rata (mean) ini didapat dengan menjumlahkan

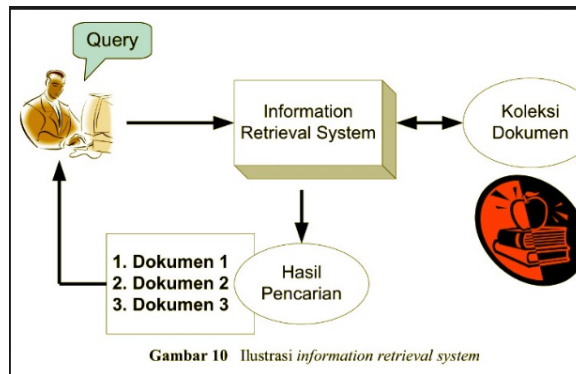


Figure 5.4: Gambar Vektorisasi Dokumen.

data seluruh individu dalam kelompok itu, kemudian dibagi dengan jumlah individu yang ada pada kelompok tersebut. 5.5



Figure 5.5: Gambar Mean.

Untuk standar deviasi sendiri merupakan sebuah teknik statistik yang digunakan dalam menjelaskan homogenitas kelompok ataupun dapat diartikan dengan nilai statistik dimana dimanfaatkan untuk menentukan bagaimana sebaran data dalam sampel, serta seberapa dekat titik data individu ke mean atau rata-rata nilai sampel yang ada. Ilustrasinya bisa dilihat pada gambar berikut 5.6

6. Jelaskan apa itu skip-gram. Dilengkapi dengan ilustrasi atau Gambar

Skip-Gram mencoba memprediksi vektor kata-kata yang ada di konteks diberikan vektor kata tertentu. Skip-Gram membuat sepasang kata target dan konteks sebagai sebuah instance sehingga Skip-Gram cenderung lebih baik ketika ukuran corpus sangat besar. Ilustrasinya bisa dilihat pada gambar berikut 5.7.

$$s = \sqrt{s^2}$$

$$s = \sqrt{\frac{1}{N-1} \sum_{i=1}^N (x_i - \bar{x})^2}$$

Keterangan:

$s^2$  = ragam atau varian sampel

$s$  = standar deviasi

$N$  = Jumlah data

$i$  = nomor data ( $i = 1, 2, 3 \dots N$ )

$x_i$  = data ke- $i$  ( $i = 1, 2, 3 \dots N$ )

$\bar{x}$  = rata-rata sampel

Figure 5.6: Gambar Deviasi.

### 5.5.2 Praktek Program

1. Mencoba dataset google dan menjelaskan vektor dari kata love, faith, fall, sick, clear, shine, bag, car, wash, motor dan cycle.
  - Pada gambar 5.8 dapat dilihat bahwa vektor memiliki array sebanyak 300 dimensi. Untuk identitas sektor satu adalah 0.10.
  - Pada gambar 5.9 untuk vektor faith dapat dilihat memiliki nilai 0.26 , untuk similaritasnya cukup mendekati vektor love dimana faith dapat dikategorikan dalam satu kategori dengan love.
  - Pada gambar 5.10 Vektor fall hanya memiliki nilai minus yaitu -0.04 , dimana mesin memahami bahwa fall tidak terdapat dalam satu kategori yang sama dengan love dan faith
  - Pada gambar 5.11 Vektor sick memiliki nilai identitas 1.82 dimana tidak mendekati love, faith maupun fall.
  - Pada gambar 5.12 Vektor clear memiliki nilai identitas -2,44 dan tidak mendekati nilai dari vektor fall sehingga tidak dapat dijadikan dalam satu kategori

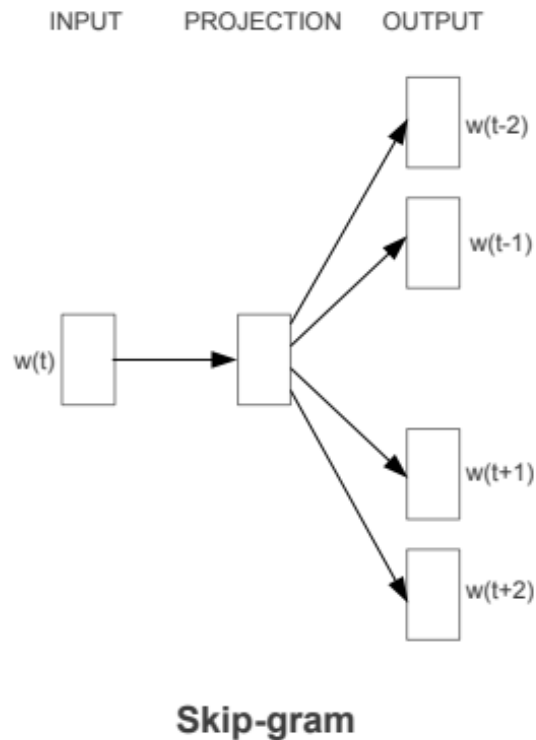


Figure 5.7: Gambar Skip-Gram.

```
In [14]: gmodel ['love']
Out[14]:
array([ 0.10302734, -0.15234375,  0.02587891,  0.16503906,
        -0.16503906,
         0.06689453,  0.29296875, -0.26367188, -0.140625 ,
         0.0117100])
```

Figure 5.8: Gambar Vektor Love.

- Pada gambar 5.13 Untuk vektor shine -0.12 tidak mendekati vektor manapun.
- Pada gambar 5.14 Vektor bag memiliki i=nilai identitas -0.03 yang mendekati dengan vektor fall. Sehingga mesin memahami bahwa mungkin saja kedua vektor tersebut berada dalam satu kategori.
- Pada gambar 5.15 Vektor car nilainya 0.13 mendekati vektor love dan faith sehingga mungkin dapat dikategorikan dalam satu kategori.
- Pada gambar 5.16 Vektor wash memiliki nilai 9.46 jauh dari vektor vektor lainnya.
- Pada gambar 5.17 Vektor motor memiliki nilai identitas 5.73 yang bisa mendekati vektor wash. Dapat dikatakan bahwa motor dapat dicuci jika diarti dalam satu kategori yang sama.



```
In [15]: gmodel['faith']
Out[15]:
array([ 0.26367188, -0.04150391,  0.1953125 ,  0.13476562,
        -0.14648438,
```

Figure 5.9: Gambar vektor faith.

```
In [16]: gmodel['fall']
Out[16]:
array([-0.04272461,  0.10742188, -0.09277344,  0.16894531, -0.1328125
,
```

Figure 5.10: Gambar vektor fall.

- Pada gambar 5.18 Vektor cycle memiliki nilai identitas 5.73 yang bisa mendekati vektor wash. Dapat dikatakan bahwa motor dapat dicuci jika diarti dalam satu kategori yang sama.
2. Mencoba untuk melakukan perbandingan similirati dari masing-masing kata tersebut. Lihat gambar 5.19 yang merupakan hasil prediksi similariti Dapat disimpulkan bahwa:

- Untuk fall dan love hasilnya adalah 11%
- Untuk fall dan faith hasilnya adalah 5%
- Untuk fall dan sick hasilnya adalah 8%
- Untuk fall dan clear hasilnya adalah 8%
- Untuk fall dan shine hasilnya adalah 27%
- Untuk fall dan bag hasilnya adalah 7%
- Untuk fall dan car hasilnya adalah 11%
- Untuk fall dan wash hasilnya adalah 14%
- Untuk fall dan cycle hasilnya adalah 19%

### 3. Extract Words dan PermuteSentences

- Extract Words : merupakan function untuk menambahkan, menghilangkan atau menghapuskan, hal hal yang tidak penting atau tidakperlu di dalam teks. Dalam contoh gambar 5.18 ini. menggunakan function extract words untuk menghapus komen dengan python style , mencari data yang diinginkan, dan memberikan spasi pada teks.
- PermuteSentences : merupakan class yang digunakan unut melakukan pengocokan secara acak pada data yang ada. Digunakan cara ini agar tidak

```
In [17]: gmodel['sick']
Out[17]:
array([ 1.82617188e-01,  1.49414062e-01, -4.05273438e-02,
        1.64062500e-01,
        -2.59765625e-01,  3.22265625e-01,  1.73828125e-01,
```

Figure 5.11: Gambar vektor sick.

```
In [18]: gmodel['clear']
Out[18]:
array([-2.44140625e-04, -1.02050781e-01, -1.49414062e-01,
        -4.24804688e-02,
```

Figure 5.12: Gambar vektor clear.

terjadi kelebihan memori pada saat dijalankan. Contoh pada gambar 5.21 yaitu fungsi akan memanggil dede. Yang kemudian mendefinisikan variabel shuffled untuk dede dan melakukan random shuffle yaitu pengocokan acak untuk kata dede.

```

In [19]: gmodel['shine']
Out[19]:
array([-0.12402344,  0.25976562, -0.15917969, -0.27734375,
        0.30273438,
        0.09960938,  0.39257812, -0.22949219, -0.18359375,  0.3671875
        ,

```

Figure 5.13: Gambar vektor shine.

```

In [20]: gmodel['bag']
Out[20]:
array([-0.03515625,  0.15234375, -0.12402344,  0.13378906,
        -0.11328125,
        -0.0133667 , -0.16113281,  0.14648438, -0.06835938,  0.140625
        ,
        -0.06005859, -0.3046875 ,  0.20996094, -0.04345703, -0.2109375

```

Figure 5.14: Gambar vektor bag.

```

In [21]: gmodel['car']
Out[21]:
array([ 0.13085938,  0.00842285,  0.03344727, -0.05883789,
        0.04003906,
        -0.14257812,  0.04931641, -0.16894531,  0.20898438,
        0.11962891,

```

Figure 5.15: Gambar Vektor car.

```

In [22]: gmodel['wash']
Out[22]:
array([ 9.46044922e-03,  1.41601562e-01, -5.46875000e-02,
        1.34765625e-01,
        -2.38281250e-01,  3.24218750e-01, -8.44726562e-02,
        -1.29882812e-01,

```

Figure 5.16: Gambar Vektor wash.

```

In [23]: gmodel['motor']
Out[23]:
array([ 5.73730469e-02,  1.50390625e-01, -4.61425781e-02,
        -1.32812500e-01,
        -2.59765625e-01, -1.77734375e-01,  3.68652344e-02,

```

Figure 5.17: Gambar vektor motor.

```

In [24]: gmodel['cycle']
Out[24]:
array([ 0.04541016,  0.21679688, -0.02709961,  0.12353516,
        -0.20703125,
        -0.1328125 ,  0.26367188, -0.12890625, -0.125
        ,

```

Figure 5.18: Gambar vektor cycle.

```

In [21]: gmodel.similarity('fall','love')
Out[21]: 0.11425873111714527

In [22]: gmodel.similarity('fall','faith')
Out[22]: 0.056926477919440686

In [23]: gmodel.similarity('fall','sick')
Out[23]: 0.08965754281727235

In [24]: gmodel.similarity('fall','clear')
Out[24]: 0.08062217411272342

In [25]: gmodel.similarity('fall','shine')
Out[25]: 0.27789493775772145

In [26]: gmodel.similarity('fall','bag')
Out[26]: 0.07147240769241402

In [27]: gmodel.similarity('fall','car')
Out[27]: 0.11321347547615472

In [28]: gmodel.similarity('fall','wash')
Out[28]: 0.1444007383236386

In [29]: gmodel.similarity('fall','cycle')
Out[29]: 0.19036458769342857

```

Figure 5.19: Gambar Similariti.

```

In [31]: import re
...: def extract_words(dede):
...:     dede = dede.lower()
...:     dede = re.sub(r'<[^>]+>', '', dede) #hapus tag html
...:     dede = re.sub(r'(\w)\s+(\w)', '\1\2', dede) #hapus petik satu
...:     dede = re.sub(r'\w', ' ', dede) #hapus tanda baca
...:     dede = re.sub(r'\s+', ' ', dede) #hapus spasi yang
berurutan
...:     return dede.split()
...:

```

Figure 5.20: Gambar Extract Words.

```

...:
...: import random
...: class PermuteSentences(object):
...:     def __init__(self, dede):
...:         self.dede = dede
...:
...:     def __iter__(self):
...:         shuffled = list(self.dede)
...:         random.shuffle(shuffled)
...:         for dede in shuffled:
...:             yield dede

```

Figure 5.21: Gambar PermuteSentences.

# 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) b. Kurang lugas dan kurang ringkas (1) c. Ringkas dan lugas (2)
2	Pencantuman Nama Penulis dan Lembaga Penulis		1	a. Tidak lengkap dan tidak konsisten (0) b. Lengkap tetapi tidak konsisten (0,5) 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) b. Abstrak kurang jelas dan ringkas, atau hanya dalam Bahasa Inggris, atau dalam Bahasa Indonesia saja (1) 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) b. Ada tetapi kurang mencerminkan konsep penting dalam artikel (0,5) 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) b. Lengkap tetapi tidak sesuai sistematika (0,5) c. Lengkap dan bersistem (1)
6	Pemanfaatan Instrumen Pendukung	Pemanfaatan Instrumen Pendukung seperti gambar dan tabel	1	a. Tidak dimanfaatkan (0) b. Kurang informatif atau komplementer (0,5) c. Informatif dan komplementer (1)
7	Cara Pengacuan dan Pengutipan		1	a. Tidak baku (0) b. Kurang baku (0,5) c. Baku (1)
8	Penyusunan Daftar Pustaka	Penyusunan Daftar Pustaka	1	a. Tidak baku (0) b. Kurang baku (0,5) c. Baku (1)
9	Peristilahan dan Kebahasaan		2	a. Buruk (0) b. Baik (1) c. Cukup (2)
10	Makna Sumbangan bagi Kemajuan		4	a. Tidak ada (0) b. Kurang (1) c. Sedang (2) d. Cukup (3) e. Tinggi (4)

Figure A.1: Form nilai bagian 1.

11	Dampak Ilmiah		7	a. Tidak ada (0) b. Kurang (1) c. Sedang (3) d. Cukup (5) 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) b. 40-80% (2) c. > 80% (3)
13	Derajat Kemutakhiran Pustaka Acuan	Derajat Kemutakhiran Pustaka Acuan	3	a. < 40% (1) b. 40-80% (2) c. > 80% (3)
14	Analisis dan Sintesis	Analisis dan Sintesis	4	a. Sedang (2) b. Cukup (3) c. Baik (4)
15	Penyimpulan	Sangat jelas relevasinya dengan latar belakang dan pembahasan, dirumuskan dengan singkat	3	a. Kurang (1) b. Cukup (2) c. Baik (3)
16	Unsur Plagiat		0	a. Tidak mengandung plagiat (0) b. Terdapat bagian-bagian yang merupakan plagiat (-5) 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] 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.
- [2] Stuart J Russell and Peter Norvig. *Artificial intelligence: a modern approach*. Malaysia; Pearson Education Limited,, 2016.