

# 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 penggerjaan 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 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 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 dengan 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 Eror . . . . .	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
5.5.3	Penanganan Error . . . . .	64

<b>6 Discussion</b>	<b>68</b>
6.1 Andri Fajar Sunandhar / 1164065 . . . . .	68
6.1.1 Teori . . . . .	68
6.1.2 Praktek . . . . .	72
6.1.3 Penanganan Error . . . . .	81
<b>7 Discussion</b>	<b>82</b>
7.1 Andri Fajar Sunandhar / 1164065 . . . . .	82
7.1.1 Teori . . . . .	82
7.1.2 Praktek . . . . .	87
7.1.3 Penanganan Error . . . . .	107
<b>8 Discussion</b>	<b>108</b>
<b>9 Discussion</b>	<b>109</b>
<b>10 Discussion</b>	<b>110</b>
<b>11 Discussion</b>	<b>111</b>
<b>12 Discussion</b>	<b>112</b>
<b>13 Discussion</b>	<b>113</b>
<b>14 Discussion</b>	<b>114</b>
<b>A Form Penilaian Jurnal</b>	<b>115</b>
<b>B FAQ</b>	<b>118</b>
<b>Bibliography</b>	<b>120</b>

# 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	Memperbarui Parameter . . . . .	11
1.22	MultiClass . . . . .	12
1.23	MultiClass biner 2D . . . . .	12
1.24	MultiLabel . . . . .	13
1.25	Eror 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 Eror . . . . .	29
3.1	Random Forest. . . . .	31
3.2	Kode membaca file.csv . . . . .	31
3.3	Window Console . . . . .	32
3.4	Variable Explorer . . . . .	32
3.5	Dataset Cell . . . . .	33
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. . . . .	64
5.16	Gambar Vektor wash. . . . .	64
5.17	Gambar vektor motor. . . . .	65
5.18	Gambar vektor cycle. . . . .	65
5.19	Gambar Similariti. . . . .	65
5.20	Gambar Extract Words. . . . .	65

5.21	Gambar PermuteSentences . . . . .	66
5.22	Gambar librari Gensim . . . . .	66
5.23	Gambar Data Training . . . . .	66
5.24	Gambar Pengocokan Data. . . . .	66
5.25	Gambar Pembersihan Data. . . . .	67
5.26	Gambar Temporary Training . . . . .	67
5.27	Gambar Infer Code . . . . .	67
5.28	Gambar Cosine. . . . .	67
5.29	Gambar Score dari cross Validation. . . . .	67
5.30	skrinsut error . . . . .	67
5.31	Solusi error . . . . .	67
6.1	Gambar MFCC. . . . .	68
6.2	Gambar Neural Network. . . . .	69
6.3	Gambar Pembobotan neural network. . . . .	70
6.4	Gambar Aktivitas neural network. . . . .	71
6.5	Gambar Plot dari MFCC. . . . .	71
6.6	Gambar One-hot Encoding. . . . .	72
6.7	Gambar np.unique. . . . .	72
6.8	Gambar to.categorical. . . . .	73
6.9	Gambar Sequential. . . . .	73
6.10	Gambar Hasil Load. . . . .	74
6.11	Gambar Display MFCC. . . . .	74
6.12	Gambar hasil display. . . . .	75
6.13	Gambar extract feature. . . . .	75
6.14	Gambar Generate feture and label. . . . .	76
6.15	Gambar Generate feture and label. . . . .	76
6.16	Gambar hasil pemisahan. . . . .	77
6.17	Gambar Sequential. . . . .	77
6.18	Gambar hasil compilel. . . . .	78
6.19	Gambar fungsi fit. . . . .	79
6.20	Gambar fungsi evaluate. . . . .	80
6.21	Gambar fungsi predict. . . . .	80
6.22	Gambar Sequential. . . . .	81
7.1	Tokenizer . . . . .	82
7.2	StratifiedKFold . . . . .	83

7.3	Ilustrasi Kode Program No.4 . . . . .	83
7.4	fungsi tokenizer . . . . .	84
7.5	fungsi dtrain inputs . . . . .	84
7.6	dtrain iinputs . . . . .	85
7.7	fungsi dtrain outputs . . . . .	85
7.8	fungsi di listing 7.2 . . . . .	86
7.9	fungsi di listing 7.3 . . . . .	86
7.10	Algoritma Konvolusi Dengan Matrix (3x3). . . . .	87
7.11	Hasil kode program pada blok 1 . . . . .	88
7.12	Hasil kode program pada blok 2 . . . . .	89
7.13	Hasil kode program pada blok 3 . . . . .	90
7.14	Hasil kode program pada blok 4 . . . . .	91
7.15	Hasil kode program pada blok 5 . . . . .	92
7.16	Hasil kode program pada blok 6 . . . . .	93
7.17	Hasil kode program pada blok 7 . . . . .	93
7.18	Hasil kode program pada blok 8 . . . . .	94
7.19	Hasil kode program pada blok 9 . . . . .	95
7.20	Hasil kode program pada blok 10 . . . . .	97
7.21	Hasil kode program pada blok 11 . . . . .	97
7.22	Hasil kode program pada blok 12 . . . . .	99
7.23	Hasil kode program pada blok 13 . . . . .	101
7.24	Hasil kode program pada blok 14 . . . . .	102
7.25	Hasil kode program pada blok 15 . . . . .	103
7.26	Hasil kode program pada blok 16 . . . . .	104
7.27	Hasil kode program pada blok 17 . . . . .	104
7.28	Hasil kode program pada blok 18 . . . . .	105
7.29	Hasil kode program pada blok 19 . . . . .	106
7.30	Hasil kode program pada blok 20 . . . . .	107
7.31	Screenshot Error . . . . .	107
A.1	Form nilai bagian 1. . . . .	116
A.2	form nilai bagian 2. . . . .	117

# Chapter 1

## Mengenal 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 plagiatis[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 mengelompokan suatu data ke data yang sudah ada. Sedangkan unsupervised learning tidak memiliki data latih, sehingga dari data yang ada, kita mengelompokan 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.

Regressi 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.

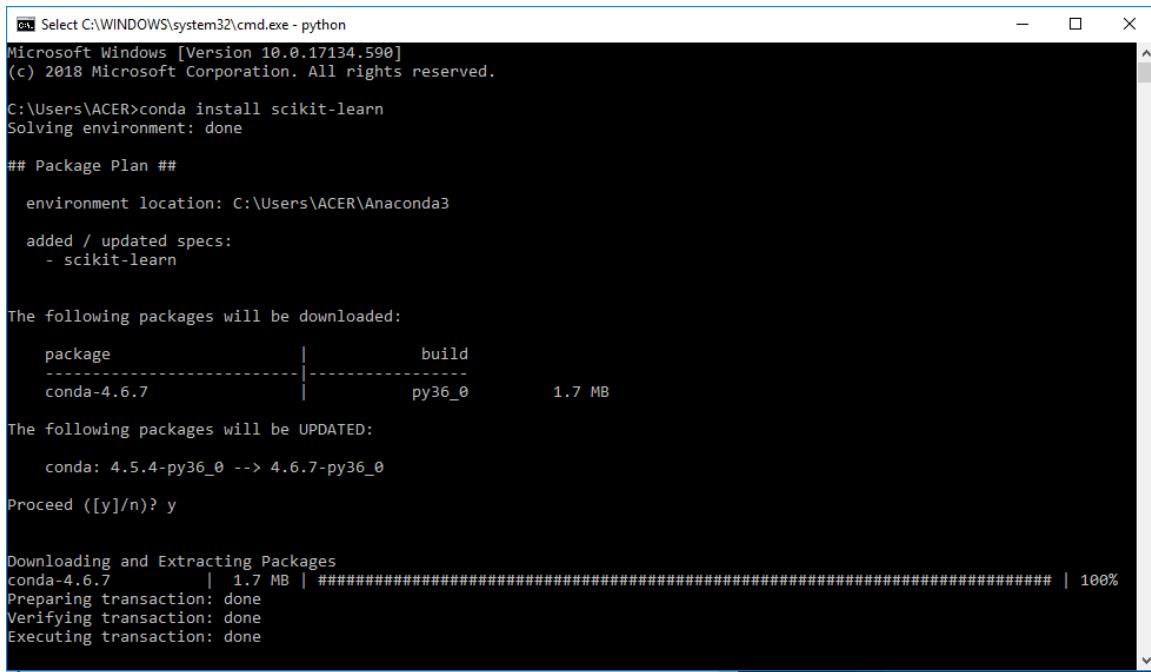
Data set adalah cabang aplikasi dari Artificial Intelligence/Kecerdasan Buatan yang fokus pada pengembangan sebuah sistem yang mampu belajar sendiri tanpa harus berulang kali di program oleh manusia.

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



```
Conda: Select C:\WINDOWS\system32\cmd.exe - python
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
  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 ketikan 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/f1f99ba67aff4c3fb033571e87876ed0403114b13bc70cc125372b0c1dc6  
/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 Persistance

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:f59c0032b4, 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

```
6. >>> import pickle
      tol=0.001, verbose=False)
      >>> s = pickle.dumps(clf)
```

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 varians 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 unutk 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

```

array([0, 1, 1, 2])
>>> 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, 1],
       [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: Eror 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).

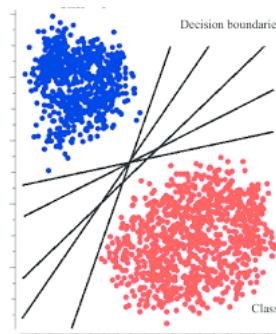


Figure 2.1: Binary Classification

### 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.

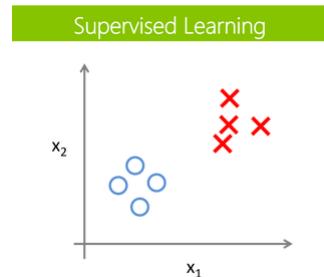


Figure 2.2: Supervised Learning

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 menanggapi umpan balik, analisis klaster mengidentifikasi kesamaan dalam data dan bereaksi berdasarkan ada tidaknya kesamaan di setiap potongan data baru. Berikut merupakan contoh Unsupervised Learning dengan Gaussian mixture models.

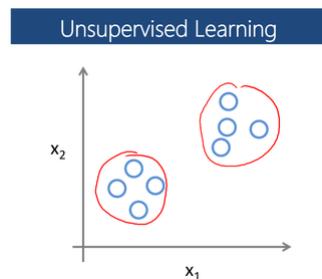


Figure 2.3: Unsupervised Learning

3. Cluster analysis or clustering adalah tugas pengelompokan sekumpulan objek sedemikian rupa sehingga objek dalam kelompok yang sama (disebut klaster) 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.

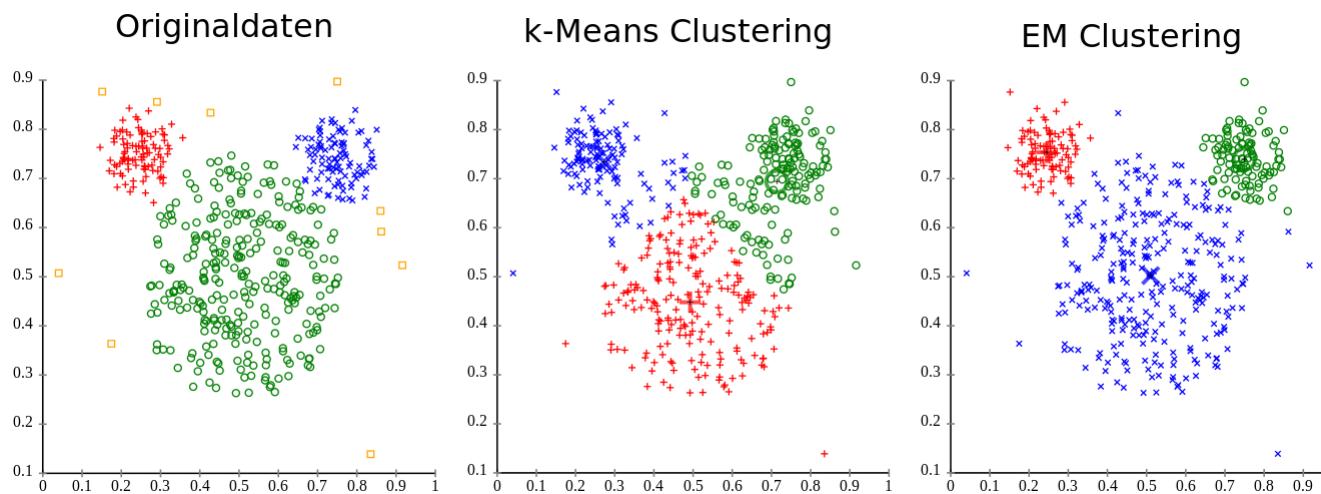


Figure 2.4: Cluster

### 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 akurasinya. Dan akurasi akan didefinisikan

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

		Predicted Motor		Predicted Mobil
		True Motor	True Mobil	
True Motor	Predicted Motor	10	8	
	Predicted 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 atributanya, 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.

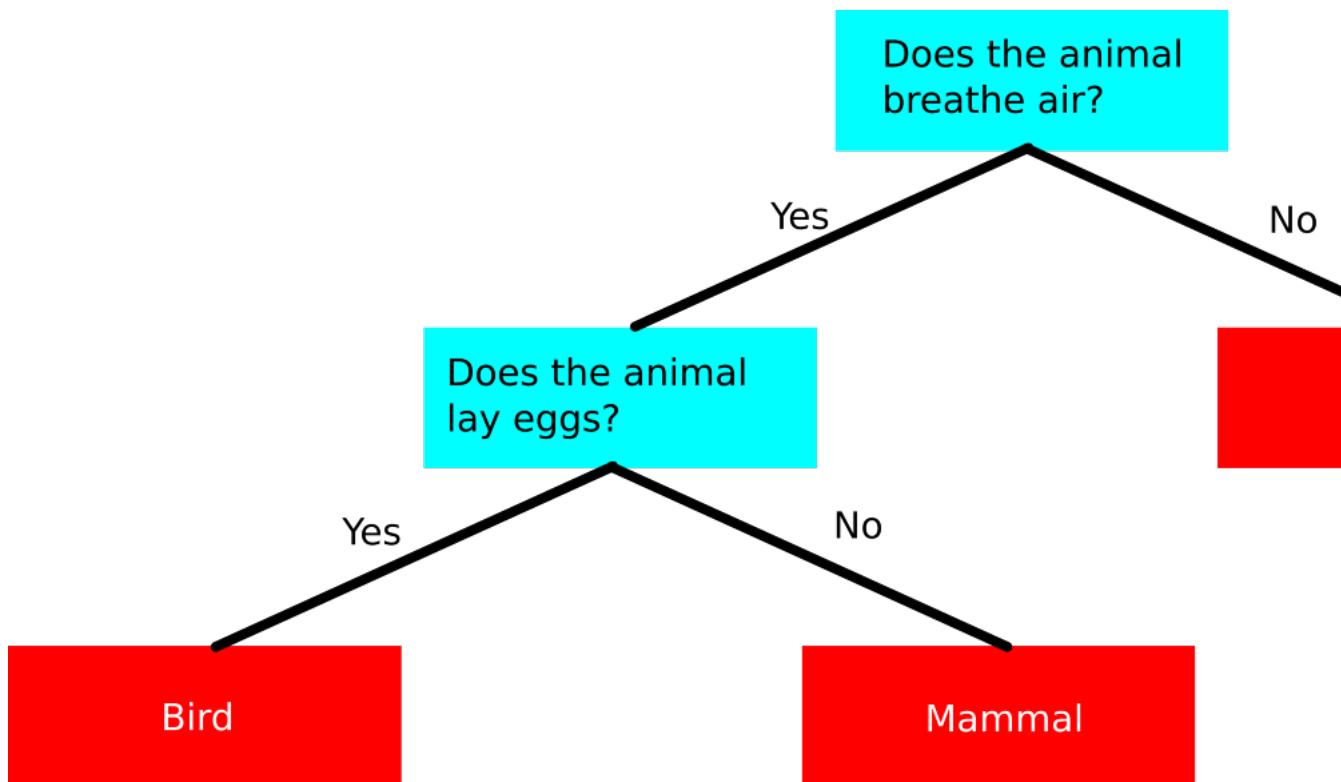


Figure 2.7: Decision Tree

### 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 apel
jeruk = apel.read_csv('E:\KAMPUS\Semester 6\Kecerdasan Buatan\modul\Python-Artificial-Intelligence-Projects-for-Beginners\Chapter01\dataset\student-mat.csv', sep=';')
len(jeruk)
```

Untuk mengimport atau memanggil module pandas sebagai apel. 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
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', 'schoolsups', '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',
                                                 ...,
                                                 'paid', 'activities',
                                                 ...,
                                                 'nursery', 'higher', 'internet', 'romantic'])
Out[9]:
   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.

Kemudian 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",
                                feature_names=list(jeruk_train_att), class_na
                                filled=True, rounded=True)

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

Mengimport Graphviz Sehingga akan muncul gambardiagram grafik bercabang.

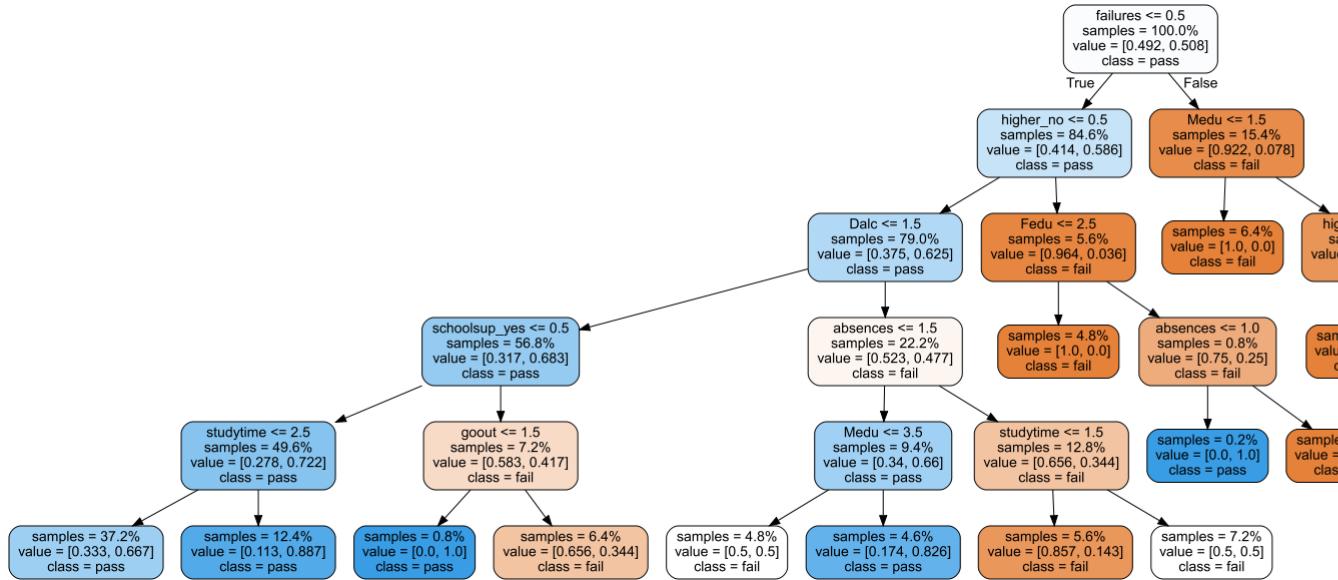


Figure 2.14: Fit Decision Tree

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(),
    scores.std() * 2))

```

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, 6.38768599e-03],
       [2.00000000e+00, 5.79817429e-01, 5.40865102e-02],
       [3.00000000e+00, 5.82348264e-01, 2.23551177e-02],
       [4.00000000e+00, 5.84752515e-01, 7.32490810e-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, 9.28981855e-02],
       [1.00000000e+01, 6.17666342e-01, 6.17102083e-02],
       [1.10000000e+01, 6.02378282e-01, 5.13911736e-02],
       [1.20000000e+01, 6.15198799e-01, 5.15762620e-02],
       [1.30000000e+01, 6.05872217e-01, 5.81263071e-02],
       [1.40000000e+01, 6.15838137e-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, 1.01208800e-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

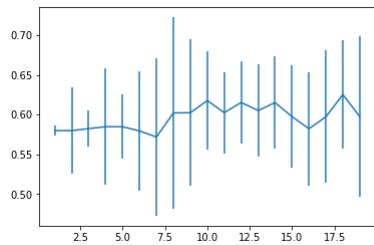


Figure 2.20: Matplotlib

## 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
In [1]: dot_data = tree.export_graphviz(t, out_file=None, label="all",
...:                         impurity=False, proportion=True,
...:                         feature_names=list(d_train_att),
...:                         class_names=["fall", "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. Eror 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       |         |
|-----------------------|-------------|---------|
| graphviz-2.38         | hfaf6e2cd_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 Eror

# **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, algoritm or equation of method 1 and how to apply into your data

### **3.3 Method 2**

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

### **3.4 Andri Fajar Sunandhar/1164065**

a

#### **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.

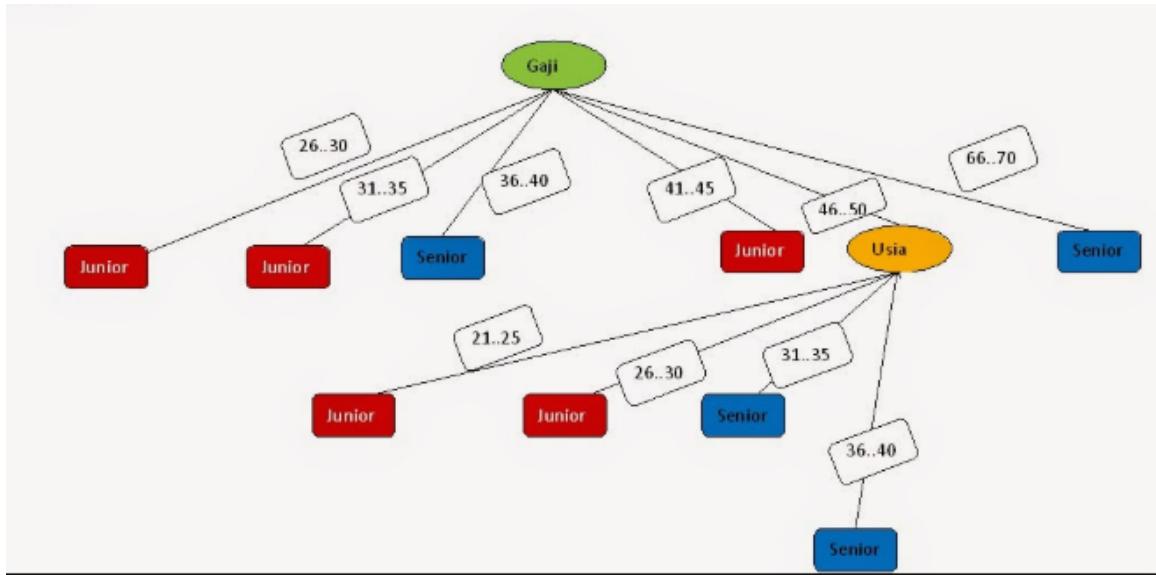


Figure 3.1: Random Forest.

## 2. Cara Membaca Dataset

- Buka Anaconda Navigator.
- Jalankan Spyder
- Import libraries yang dibutuhkan
- Masukan kode berikut untuk membaca file Data.csv.

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

Figure 3.2: Kode membaca file.csv

- Jalankan kode tersebut, maka di windows console akan muncul pesan :
- Klik variable explorer, maka akan terlihat dataset yang baru ter-import.
- Kemudian double klik pada dataset cell, maka akan muncul pop-up windows seperti berikut:

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

Figure 3.3: Window Console

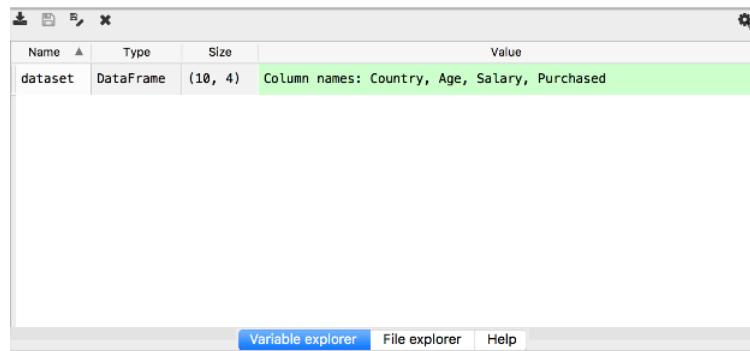


Figure 3.4: Variable Explorer

- (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

Index	Country	Age	Salary	Purchased
0	France	44	72000	No
1	Spain	27	48000	Yes
2	Germany	38	54000	No
3	Spain	38	61000	No
4	Germany	48	nan	Yes
5	France	35	58000	Yes
6	Spain	nan	52000	No
7	France	48	79000	Yes
8	Germany	58	83000	No
9	France	37	67000	Yes

Figure 3.5: Dataset Cell

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:

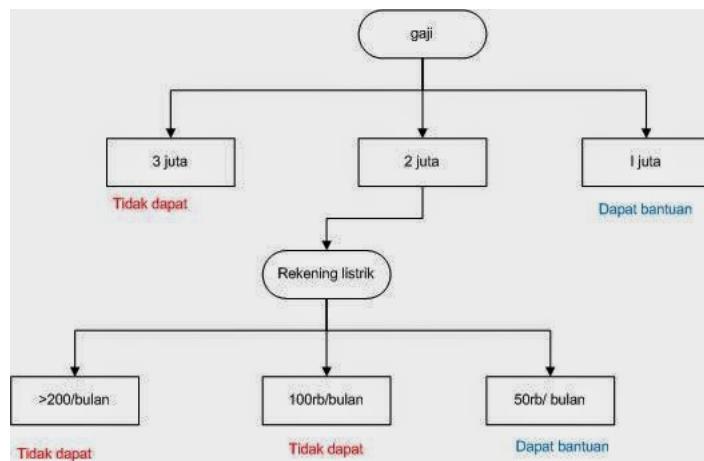


Figure 3.6: Pohon Keputusan

no	nama	gaji	rekening	hasil	kecocokan
1	Aji	3 juta	100rb/bulan	dapat bantuan	t
2	Ali	1 juta	50rb/bulan	dapat bantuan	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 bantuan	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 bantuan	y
10	Sohe	2 juta	50rb/bulan	tidak dapat	t

Figure 3.7: Data Testing

Kemudian data testingnya adalah

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

$$a = 5$$

$$b = 1$$

$$c = 1$$

$$d = 3$$

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

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

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

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

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

## 5. 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

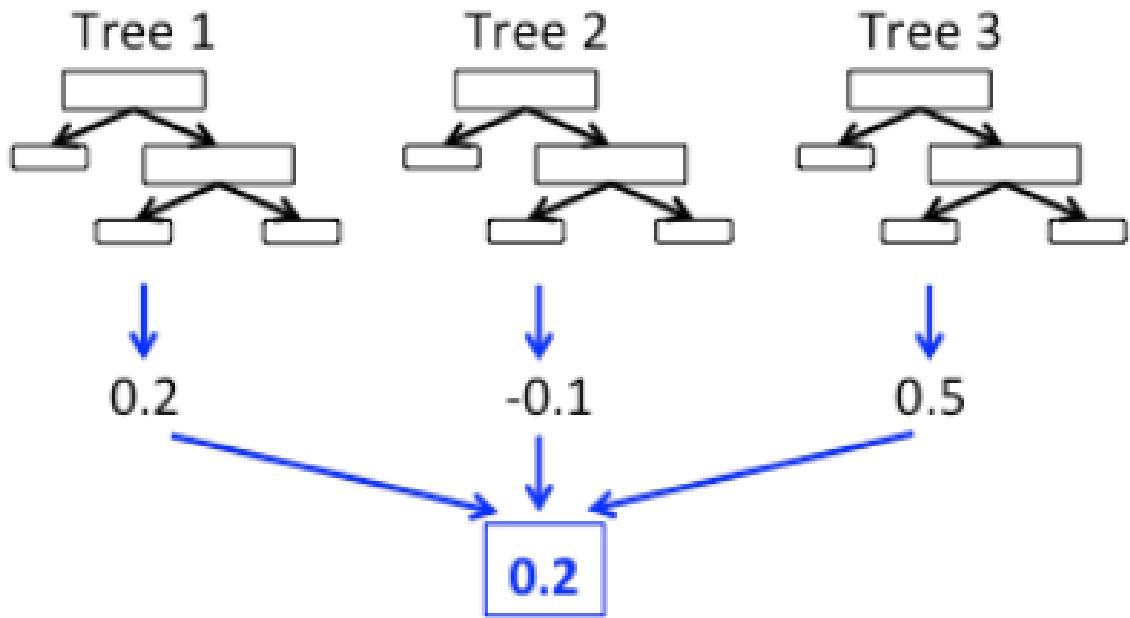


Figure 3.8: Voting.

```
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

Penjelasan kodingan :

- (a) Memanggil library.
- (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

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

Figure 3.11: Aplikasi Numpy

- (b) Membuat variable dengan value eye dengan size10
- (c) Menampilkan hasil value

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 :

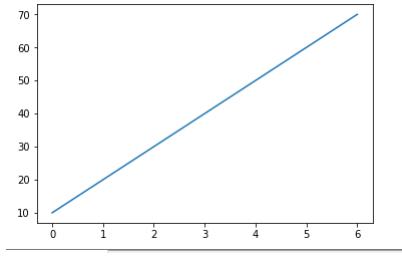


Figure 3.14: Hasil Matplotlib

```
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='\t', 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

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

#### 10. Code Random Forest 3 :

- Penjelasan : Codingan 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 menampilkan 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 menampilkan hasil dari variabel imgatt2.head. Dimana dataset nya ada 5 baris dan 312 kolom.

```
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

### 13. Code Random Forest 6 :

- 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 :

```
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    ...
2      0    0    0    0    0    0    0    ...
3      0    0    0    0    1    0    0    ...
4      0    0    0    0    1    0    0    ...
5      0    0    0    0    1    0    0    ...

[5 rows x 312 columns]
```

Figure 3.19: Gambar 5

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

```
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 pembagian dari data training dan dataset

22. Code Random Forest 15 :

- Penjelasan : Pada gambar di atas merupakan pemanggilan kelas RandomForestClassifier. 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 dftsetlabel. Dimana hasilnya dapat dilihat dari pada gambar di atas

```
In [29]: imglabels.shape  
Out[29]: (11788, 1)
```

Figure 3.22: Gambar 8

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

Figure 3.23: Gambar 9

## 26. Program Klasifikasi Confusion Matrix

- Setelah melakukan random forest kemudian dipetakan ke dalam confusion matrix.
- 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 klasifikasi dengan SVM dengan dataset yang sama.

## 28. Code Decision Tree :

- Penjelasan : Pada gambar di atas merupakan cara untuk mencoba klasifikasi dengan decision tree dengan dataset yang sama.

## 29. Program Cross Validation

- Melakukan pengecekan cross validation untuk random forest.
- Melakukan pengecekan cross validation untuk decision 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 Eror

Penyelesaian Tugas Harian ( Penanganan Error )

1. Menyelesaikan dan Membahas Penanganan Error :

- Skrinsut 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
      ...
10779  0    0    0    0    0    0    1 ...    1    0    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

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]: scorestree = cross_val_score(clftree, df_train_att, df_train_label,
cv=5)
...: print("Accuracy: %0.2f (%+/- %0.2f)" % (scorestree.mean(),
scorestree.std() * 2))
Accuracy: 0.27 (%+/- 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_zlim(0.2, 0.5)
.... ax.set_xlabel('Max features')
.... ax.set_ylabel('Num estimators')
.... ax.set_zlabel('Avg accuracy')
.... plt.show()
```

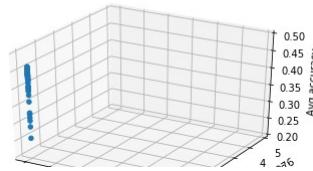


Figure 3.43: Plot informasi

```
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

FileNotFoundException: 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**

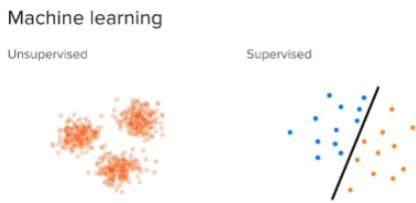


Figure 4.1: Klasifikasi teks



Figure 4.2: Klasifikasi bunga

Kita ambil sebuah kasus yang semua orang telah ketahui dan juga pahami. Kasus tersebut yaitu perekendasian video dari pencarian menggunakan "text / kata" di Youtube. Pada saat menggunakan Youtube terdapat Machine 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 memberikan kemudahan pada pencarian yang lannya, 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.

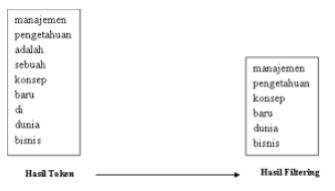


Figure 4.4: Bag of Word

#### 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

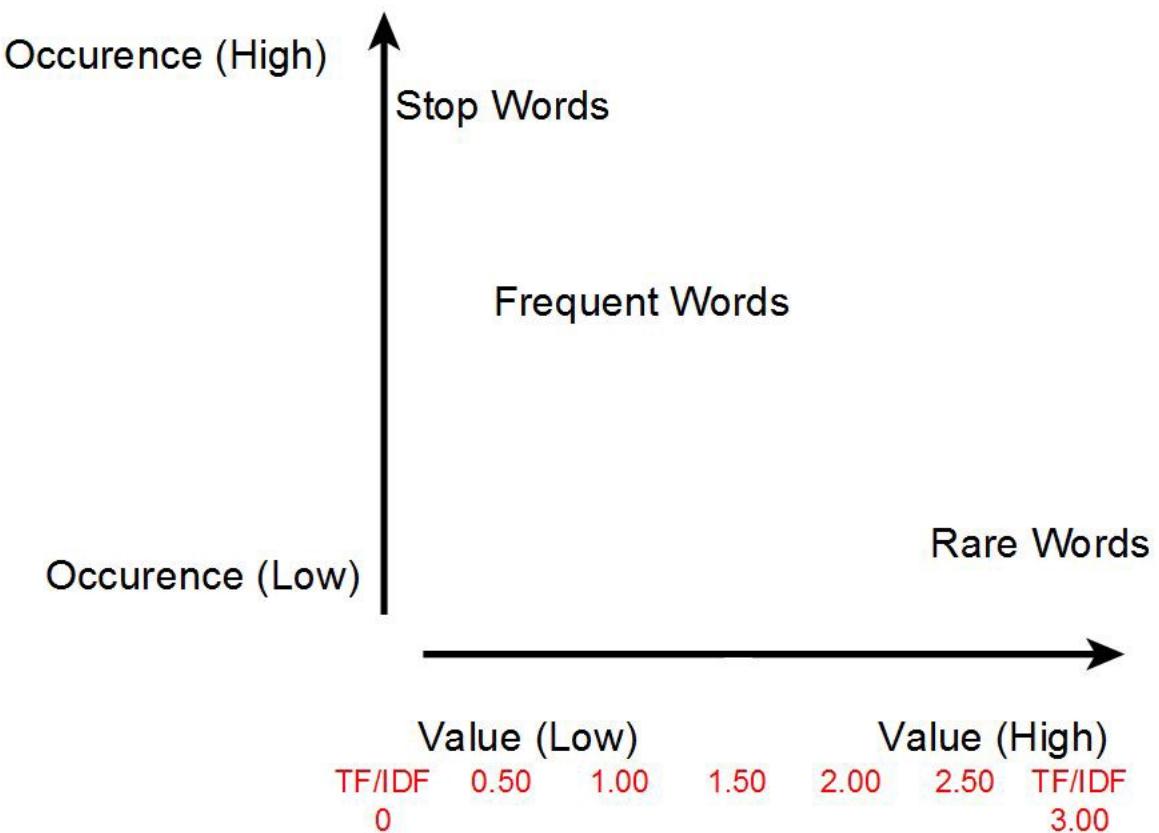


Figure 4.5: TF IDF

```
# 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 variabel 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 Decission 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 colom CONTENT untuk dilakukan vektorisasi
18 #melakukan bag of word pada dataframe yang ada pada colom 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 colomnya 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 attribute 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

Dari 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.

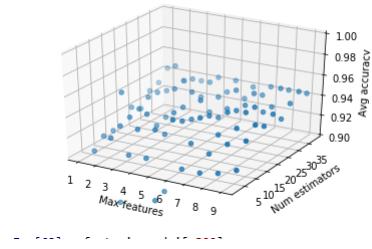


Figure 4.16: Program pengamatan komponen informasi

### 4.3.3 Penanganan Error

#### 1. skrinsut error

```

    return self._engine.get_loc(self._maybe_lower(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: skrinsut error

#### 2. Tuliskan kode eror 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 menetukan kata kunci. Ilustrasinya bisa dilihat pada gambar berikut 7.1.

#### Word Vector Representations

Use a sliding window over a big corpus of text and count word co-occurrences in between.

$$X = \begin{bmatrix} 1 & like & enjoy & deep & learning & NLP & flying & . \\ like & \begin{bmatrix} 0 & 2 & 1 & 0 & 0 & 0 & 0 & 0 \\ 2 & 0 & 0 & 1 & 0 & 1 & 0 & 0 \\ enjoy & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ deep & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 \\ learning & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 \\ NLP & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 \\ flying & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 \\ . & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 \end{bmatrix} \end{bmatrix}$$

1. I enjoy flying.  
2. I like NLP.  
3. I like deep learning.

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 seuba 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 diemensi vektor makan disimpan menjadi 300 dimensi vektor untuk mengurangi kegagalan memori. Ilustrasinya bisa dilihat pada gambar berikut 7.2.

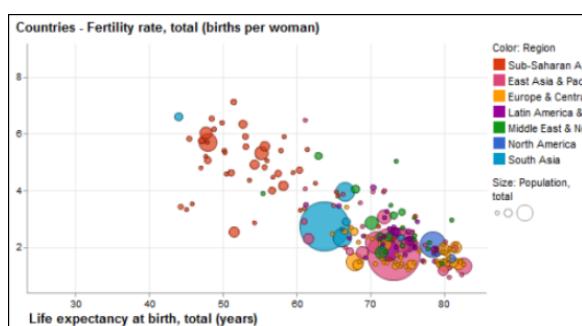


Figure 5.2: Gambar Vektorisasi Dataset Google.

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 6.3.

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

Figure 5.3: Gambar Vektorisasi Kata.

4. Jelaskan konesep vektorisasai 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 7.3.

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

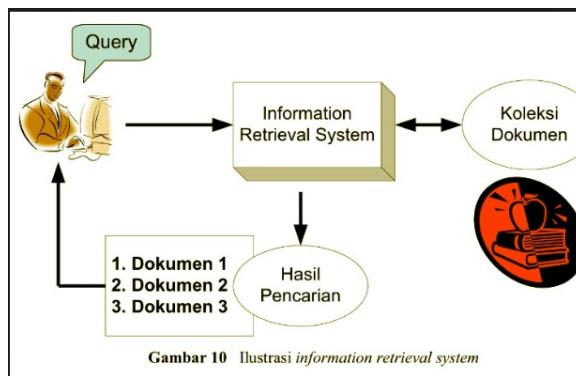


Figure 5.4: Gambar Vektorisasi Dokumen.

data seluruh individu dalam kelompok itu, kemudian dibagi dengan jumlah individu yang ada pada kelompok tersebut. 6.5



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 7.5.

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

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

**Keterangan:**

$s^2$  = ragam atau varian sampel

$s$  = standar deviasi

$N$  = Jumlah data

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

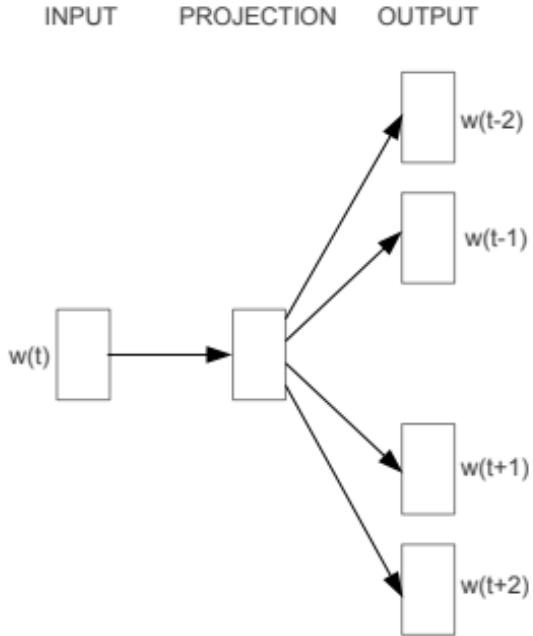
$x_i$  = data ke- $i$  ( $i = 1, 2, 3 \dots N$ )

$\bar{x}$  = 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



## Skip-gram

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 similitas 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 tidak perlu 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 untuk 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.

4. Fungsi dari librari gensim TaggedDocument dan Doc2Vec disertai praktek pemakaiannya. Tunjukkan keluarannya dari komputer sendiri dan artikan maksud setiap luaran yang didapatkan.
  - Fungsi dari Librari Gensim TaggedDocument dan Doc2Vec :
  - Penjelasan pada gambar 5.22 : yaitu mengimport modul Tagged Document dari model gensim. Sedangkan pada baris kedua import Doc2vec dari model gensim.

## 5. Cara Menambahkan Data Training

- Penskalaan yaitu dataset ditempatkan sambil mempertimbangkan kumpulan data linier seperti data bank.
- Disintegrasi dan Komposisi: Langkah ini melibatkan pemecahan fitur tertentu untuk membangun data pelatihan yang lebih baik untuk model yang Anda pahami lebih komprehensif.
- Komposisi: Ini adalah proses terakhir yang melibatkan penggabungan berbagai fitur menjadi satu fitur untuk mendapatkan data yang lebih akurat atau bermakna.
- Penjelasan pada gambar 5.23 : Membaca direktori name dari data yang ada di dalam kurung. Pada kodingan di atas ada 3 data. Data pertama yaitu: train/pos,train/neg, train/unsuo, test/pos dan test/neg. Data

```
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.

kedua yaitu:txtsentoken/pos dan txt/sentokenneg. Sedangkan data ke 3 yaitu: standfordSentimentTreeBank/Originalrtsnippes.txt

## 6. Kenapa Harus Dilakukan Pengocokan dan Pembersih Data

Pengocokan data dilakukan untuk merandom data-data yang ada. Sedangkan pembersihan data dilakukan untuk pengecekan data untuk penetapan dan pemulihan data yang hilang, pengecekan penetapan meliputi pemerikasaan data yang out of range (di luar cakupan) tidak konsisten secara logika, ada nilai-nilai ekstrim, data dengan nilai-nilai tdk terdefinisi, sedangkan pemulihan data yang hilang adalah nilai dari suatu variabel yang tidak diketahui dikarenakan jawaban responden yang membingungkan.

- Penjelasan pada gambar 5.24 : yaitu melakukan pengacakan model terhadap data unsupervised learning. Dan kemudian baru membuat modelnya setelah dilakukan pengacakan terhadap yang pertama tadi.
- Penjelasan pada gambar 5.25 : Mengimport Library Re. Kemudian membuat fungsi untuk menghapus tag html dan perkocokan. Dimana di dalam variabel ini ada kodingan untuk menghapus tag html yaitu petik satu, tanda baca dan spasi yang berurutan.

## 7. Model harus di Save dan Kenapa Temporari Training Harus Dihapus

Karena berfungsi untuk mencegah ram agar tidak lemot dan untuk mengosongkan memori agar sedikit lega atau tidak lemot.

- Penjelasan pada gambar 5.27 : Menghapus Temporary Training Data

## 8. InferCode

```
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.

Infercode digunakan untuk menyimpulkan vektor yang berhubungan dengan vektor dokumen baru.

- Penjelasan pada gambar 5.27 : Pada hasil gambar tersebut dapat disimpulkan bahwa kalimat "Belajar Jarkom" menghasilkan outputan berupa array seperti angka-angka di bawah.

#### 9. Cosine Similarity.

- Penjelasan pada gambar 5.28 : yaitu mengimport cosine similarity dari sklearn metrics pairwise. Dimana cosine similarity berfungsi untuk mengecek kemiripan dari kalimat Services Sucks. Dan akan muncul hasilnya yaitu array.

#### 10. Score Dari Cross Validation

- Penjelasan pada gambar 5.29 : Menghitung model clrf, sentvecs, setiments serta juga meghitung keakuratannya.

### 5.5.3 Penanganan Error

1. skrinsut error

2. Tuliskan kode eror dan jenis errornya

- Kode error = ModuleNotFoundError: No module named 'gensim'
- Jenis error = Module Not found Error

3. Solusi pemecahan masalah error

Solusinya adalah dengan mrnginstall modul gensim dengan perintah ...

```
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(debe):
...:     debe = debe.lower()
...:     debe = re.sub(r'<[^>]+>', ' ', debe) #hapus tag html
...:     debe = re.sub(r'(\w)\.(\w)', ' ', debe) #hapus petik satu
...:     debe = re.sub(r'\W', ' ', debe) #hapus tanda baca
...:     debe = re.sub(r'\s+', ' ', debe) #hapus spasi yang berurutan
...:     return debe.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.

```

In [21]: from gensim.models.doc2vec import TaggedDocument
...: from gensim.models import Doc2Vec

```

Figure 5.22: Gambar librari Gensim

```

In [5]: for dirname in ["train/pos", "train/neg", "train/unsup", "test/
pos", "test/neg"]:
...:     for fname in sorted(os.listdir("aclImdb/" + dirname)):
...:         if fname[-4:] == '.txt':
...:             with open("aclImdb/" + dirname + "/" + fname,
encoding='UTF-8') as f:
...:                 sent = f.read()
...:                 words = extract_words(sent)
...:                 unsup_sentences.append(TaggedDocument(words,
[dirname + "/" + fname]))
In [6]: for dirname in ["txt_sentoken/pos", "txt_sentoken/neg"]:
...:     for fname in sorted(os.listdir(dirname)):
...:         if fname[-4:] == '.txt':
...:             with open(dirname + "/" + fname, encoding='UTF-8')
as f:
...:                 for i, sent in enumerate(f):
...:                     words = extract_words(sent)
...:                     unsup_sentences.append(TaggedDocument(words,
["%s/%s-%d" % (dirname, fname, i)]))
In [7]: with open("stanfordSentimentTreebank/original_rt_snippets.txt",
encoding='UTF-8') as f:
...:     for i, line in enumerate(f):
...:         words = extract_words(line)
...:         unsup_sentences.append(TaggedDocument(words, ["rt-%d" %
i]))

```

Figure 5.23: Gambar Data Training.

```

In [28]: mute=PermuteSentences(unsup_sentences)
In [29]: model = Doc2Vec(mute, dm=0, hs=1, vector_size=52)

```

Figure 5.24: Gambar Pengocokan Data.

```

In [22]: import re
...:     def extract_words(sent):
...:         sent = sent.lower()
...:         sent = re.sub(r'<[^>]+>', ' ', sent) #hapus tag html
...:         sent = re.sub(r'(\w)\\'(\w)', ' ', sent) #hapus petik satu
...:         sent = re.sub(r'\W', ' ', sent) #hapus tanda baca
...:         sent = re.sub(r'\s+', ' ', sent) #hapus spasi yang berurutan
...:     return sent.split()
...:
...:
...:     import random
...:     class PermuteSentences(object):
...:         def __init__(self, sents):
...:             self.sents=sents
...:
...:         def __iter__(self):
...:             shuffled = list(self.sents)
...:             random.shuffle(shuffled)
...:             for sent in shuffled:
...:                 yield sent

```

Figure 5.25: Gambar Pembersihan Data.

```
In [70]: model.delete_temporary_training_data(keep_inference=True)
```

Figure 5.26: Gambar Temporary Training

```

In [74]: model.save('rozroza.d2v')
2019-03-29 22:26:14,715 : INFO : saving Doc2Vec object under rozroza.d2v, separately None
2019-03-29 22:26:18,533 : INFO : saved rozroza.d2v

```

Figure 5.27: Gambar Infer Code

```

In [77]: from sklearn.metrics.pairwise import cosine_similarity
...: cosine_similarity(
...:     [model.infer_vector(extract_words("Services sucks2"))],
...:     [model.infer_vector(extract_words("Services sucks2."))])
Out[77]: array([[0.8722154]], dtype=float32)

```

Figure 5.28: Gambar Cosine.

```

In [80]: scores = cross_val_score(lclf, sentvecs, sentiments, cv=5)
...:     : np.mean(scores), np.std(scores)
C:\Users\lsvapr\Anaconda3\lib\site-packages\sklearn\ensemble\forest.py:246: FutureWarning: The default value of n_estimators will change from 10 in version 0.20 to 100 in 0.22.
"10 in version 0.20 to 100 in 0.22.", FutureWarning)
C:\Users\lsvapr\Anaconda3\lib\site-packages\sklearn\ensemble\forest.py:246: FutureWarning: The default value of n_estimators will change from 10 in version 0.20 to 100 in 0.22.
"10 in version 0.20 to 100 in 0.22.", FutureWarning)
C:\Users\lsvapr\Anaconda3\lib\site-packages\sklearn\ensemble\forest.py:246: FutureWarning: The default value of n_estimators will change from 10 in version 0.20 to 100 in 0.22.
"10 in version 0.20 to 100 in 0.22.", FutureWarning)
C:\Users\lsvapr\Anaconda3\lib\site-packages\sklearn\ensemble\forest.py:246: FutureWarning: The default value of n_estimators will change from 10 in version 0.20 to 100 in 0.22.
"10 in version 0.20 to 100 in 0.22.", FutureWarning)
C:\Users\lsvapr\Anaconda3\lib\site-packages\sklearn\ensemble\forest.py:246: FutureWarning: The default value of n_estimators will change from 10 in version 0.20 to 100 in 0.22.
"10 in version 0.20 to 100 in 0.22.", FutureWarning)
Out[80]: (0.7036666666666667, 0.017776388834631156)

```

Figure 5.29: Gambar Score dari cross Validation.

```

In [1]: import gensim, logging
Traceback (most recent call last):
  File "<ipython-input-1-9f9b7323a65c>", line 1, in <module>
    import gensim, logging
ModuleNotFoundError: No module named 'gensim'

```

Figure 5.30: skrinsut error

```
(base) C:\Users\lsvapr>conda install gensim
```

Figure 5.31: Solusi error

# Chapter 6

## Discussion

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

### 6.1 Andri Fajar Sunandhar / 1164065

#### 6.1.1 Teori

1. Mengapa file suara harus dilakukan MFCC, dilengkapi dengan ilustrasi atau gambar.

Mel Frequency Cepstral Coefficients (MFCC) merupakan koefisien yang merepresentasikan audio. Sehingga diharuskannya melakukan MFCC kepada objek suara atau audio agar suara dapat berubah atau diubah ke dalam bentuk data matrix dimana telah dilakukan ekstraksi oleh MFCC kemudian direalisasikan sebagai data matrix. Ilustrasinya bisa dilihat pada gambar berikut 7.1.

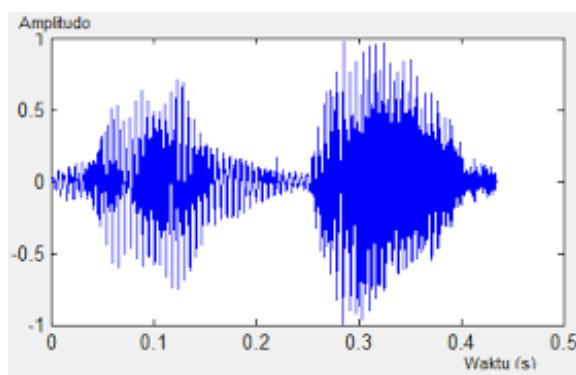


Figure 6.1: Gambar MFCC.

2. Konsep dasar Neural Network, dilengkapi dengan ilustrasi atau gambar.

Neural Network merupakan kategori ilmu Soft Computing. Neural Network sebenarnya mengadopsi dari kemampuan otak manusia yang mampu memberikan stimulasi/rangsangan, melakukan proses, dan memberikan output. Output diperoleh dari variasi stimulasi dan proses yang terjadi di dalam otak manusia. Kemampuan manusia dalam memproses informasi merupakan hasil kompleksitas proses di dalam otak. Misalnya, yang terjadi pada anak-anak, mereka mampu belajar untuk melakukan pengenalan meskipun mereka tidak mengetahui algoritma apa yang digunakan. Ilustrasinya bisa dilihat pada gambar berikut 7.2.

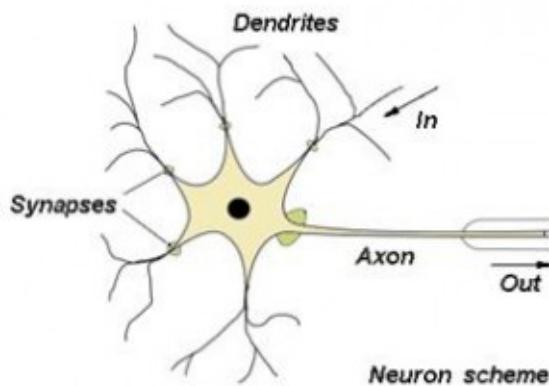


Figure 6.2: Gambar Neural Network.

3. Konsep pembobotan Neural Network, dilengkapi dengan ilustrasi atau gambar.

Sebuah Neural Network dikonfigurasi untuk aplikasi tertentu, seperti pengecualian pola atau klasifikasi data. Terjadi penglibatan dalam penyesuaian koneksi sinaptik yang ada antara neuron ketika melakukan penyempurnaan dengan proses pembelajaran. Penyesuaian nilai bobot yang ada pada tiap koneksi sinaptik antar neuron itu sendiri. Ilustrasinya bisa dilihat pada gambar berikut 6.3.

4. Konsep fungsi aktifasi dalam Neural Network, dilengkapi dengan ilustrasi atau gambar

Operasi matematik yang dikenakan pada sinyal output  $y$ . Sehingga fungsi ini akan digunakan untuk pengaktifan dan juga penonaktifan neuron. Ilustrasinya bisa dilihat pada gambar berikut 7.3.

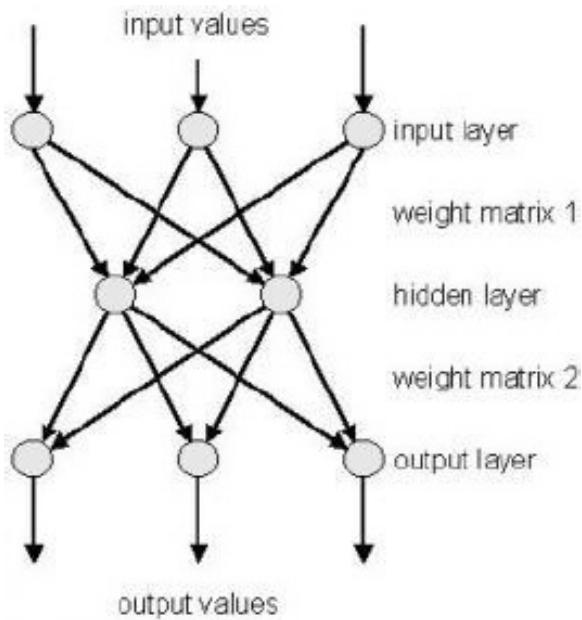


Figure 6.3: Gambari Pembobotan neural network.

5. Cara membaca hasil plot dari MFCC, dilengkapi dengan ilustrasi atau gambar.

Nanti akan ada outputan berbentuk grafik. Ada 3 dimensi atau sumbu. Dimana untuk sumbu x merupakan waktu, sedangkan sumbu y merupakan frekuensi dari suara yang dihasilkan dalam bentuk Hz. Sedangkan pada bagian tengah atau sumbu z merupakan power atau kekuatan dari lagu atau suara atau desibel yang dihasilkan. Untuk melihat penjelasan dari warna pada gambar yang di tengah, maka kita harus mendownload gambar nya terlebih dahulu. Untuk warna biru itu merupakan suara rendah, yang merah merupakan tinggi dan daya frekuensi nya berada pada nilai yang rendah karena bass bekerja pada suara yang rendah. Tidak selalu tergantung pada warna untuk menentukan nilai atau frekuensinya. Tetapi pada jenis lagu yang digunakan. Ilustrasinya bisa dilihat pada gambar berikut 6.5

6. Apa itu One-Hot Encoding, dilengkapi dengan ilustrasi atau gambar.

One-Hot Encoding adalah sekelompok bit yang kombinasinya hukumnya hanya terdiri dari bit dengan bit tinggi (1) dan bit lainnya rendah (0). Implementasi serupa di mana semua bit '1' kecuali satu '0' kadang-kadang disebut one-hot. Dalam statistik, variabel dummy mewakili teknik serupa untuk mewakili data kategorikal. Ilustrasinya bisa dilihat pada gambar berikut 7.5.

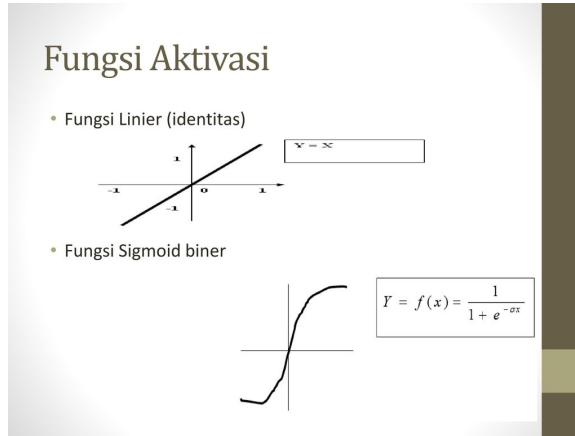


Figure 6.4: Gambar Aktivitas neural network.

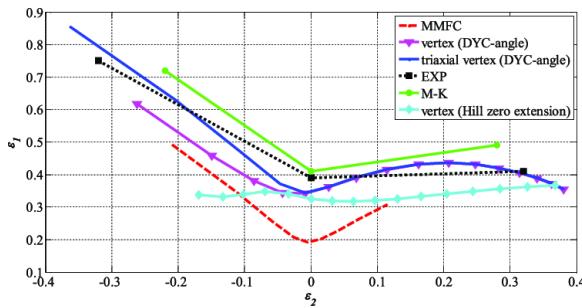


Figure 6.5: Gambar Plot dari MFCC.

7. Fungsi dari np.unique dan to.categorical, dilengkapi dengan ilustrasi atau gambar.

Np.unique Berfungsi untuk menemukan elemen unik array. Ilustrasinya bisa dilihat pada gambar berikut 6.7. Ada tiga output opsional selain elemen unik:

- Indeks array input yang memberikan nilai unik
- Indeks array unik yang merekonstruksi array input
- Berapa kali setiap nilai unik muncul dalam array input

To.categorical Berfungsi untuk menemukan elemen unik array. Ilustrasinya bisa dilihat pada gambar berikut 6.8.

8. Fungsi dari Sequential, dilengkapi dengan ilustrasi atau gambar.

Sebuah jenis model yang digunakan dalam perhitungan ataupun code program yang direalisasikan. Neural Networks Sequential membangun fitur tingkat tinggi melalui lapisannya yang berurutan. Sequential juga merupakan proses dimana

Color	Red	Yellow	Green
Red	1	0	0
Red	1	0	0
Yellow	0	1	0
Green	0	0	1
Yellow	0	0	1

Figure 6.6: Gambar One-hot Encoding.

```
def naive_bayes(training, outcome, new_sample):
    classes = np.unique(outcome)
    rows, cols = np.shape(training)
    likelihoods = {}
    for cls in classes:
        likelihoods[cls] = defaultdict(list)
    class_probabilities = occurrences(outcome)
```

Figure 6.7: Gambar np.unique.

membandingkan setiap elemen larik satu per satu secara beruntun, mulai dari elemen pertama, sampai dengan elemen terakhir atau elemen yang dicari sudah ditemukan. Ilustrasinya bisa dilihat pada gambar berikut 7.7.

### 6.1.2 Praktek

1. Data GTZAN Genre Collection dan data dari freesound.

Isi Data GTZAN Genre Collection adalah file musik yang difolderkan berdasarkan genre atau jenis lagu dan data freesound berisikan suara alam contohnya adalah suara lebah, suara air, dan sebagainya.

Berikut adalah kode program untuk meload data tersebut :

Berikut penjelasan dari kode program :

- membuat variabel untuk mendefinisikan direktori file yang akan digunakan.
- membuat variabel untuk meload atau memanggil variabel filename\_rock.

Apabila kode program dijalankan maka akan menghasilkan seperti gambar 7.8 :

2. Fungsi dari display\_mfcc() .

Berikut adalah kode program yang digunakan :

Setelah kode program dijalankan, maka akan muncul seperti gambar ?? :

Penjelasan Gambar :

```
keras.utils.to_categorical(y, num_classes=None, dtype='float32')
```

Figure 6.8: Gambar to.categorical.

11	35	42	77	89	102
----	----	----	----	----	-----

Figure 6.9: Gambar Sequential.

- Baris 1 : membuat fungsi display mfcc untuk menampilkan vektorisasi dari sebuah suara
- Baris 2 : membuat variabel y untuk meload atau membaca variable song dari perintah librosa load song
- Baris 3 : membuat variabel mfcc untuk memenggil variabel y dan mengubah suara menjadi vektor
- Baris 4 : memploting gambar dengan ukuran 10x4
- Baris 5 : menampilkan spektrogram/chromagram
- Baris 6 : menambahkan colorbar pada plot
- Baris 7 : menetapkan atau memberikan judul untuk suara
- Baris 8 : untuk memberikan label pada sumbu di grafik
- Baris 9 : fungsi untuk menampilkan hasil plot

Untuk menampilkan hasil plotting dilakukan perintah berikut :

Apabila program tersebut dijalankan, maka akan muncul hasil seperti gambar 7.9 :

Penjelasan kode : display MFCC digunakan untuk menampilkan hasil dari gambar 6.11. Dimana gambar tersebut menjelaskan kekuatan atau daya dari sebuah suara.

### 3. Fungsi dari extract\_features\_song().

Untuk menggunakan fungsi dari extract\_features\_song() diimplementasikan pada kode berikut :

Jika kode di atas dijalankan maka didapat hasil seperti gambar 6.13 :

Berikut adalah penjelasan dari gambar 6.13:

```
In [19]: filename_rock = 'genres/rock/rock.00000.au'
...: x_rock, sr_rock = librosa.load(filename_rock, duration=10)
```

Figure 6.10: Gambar Hasil Load.

```
In [8]: def display_mfcc(song):
...:     y, _ = librosa.load(song)
...:     mfcc = librosa.feature.mfcc(y)
...:
...:     plt.figure(figsize=(10, 4))
...:     librosa.display.specshow(mfcc, x_axis='time', y_axis='mel')
...:     plt.colorbar()
...:     plt.title(song)
...:     plt.tight_layout()
...:     plt.show()
```

Figure 6.11: Gambar Display MFCC.

- Baris 1 : membuat fungsi extract feature song dengan inputan f
- Baris 2 : membuat variabel y untuk meload atau membaca inputan f dari perintah librosa load song
- Baris 3 : membuat variabel mfcc untuk membuat feature dari variabel y
- Baris 4 : membuat normalisasi nilai antara -1 sampai 1
- Baris 5 : mengambil 25000 data pertama berdasarkan durasi suara atau musik lalu dikembalikan salinan arraynya dan dikecilkan menjadi satu.

Mengapa yang diambil 25.000 baris pertama? Karena menyesuaikan dengan kapasitas penyimpanan supaya tidak lemot.

#### 4. Penjelasan kode program generate\_features\_and\_labels()

Disini akan mengilustrasikan fungsi dari generate\_feature\_and\_labels(). Kode program yang digunakan adalah seperti berikut :

Apabila kode program tersebut dijalankan akan menghasilkan seperti pada gambar 6.14.

Berikut adalah penjelasan dari hasil gambar ilustrasi 6.14:

- Baris 1 : membuat perintah fungsi generate features and labels
- Baris 2 : membuat variabel all features dengan array kosong
- Baris 3 : membuat variabel all labels dengan array kosong
- Baris 4 : Membuat variable yang berisi nama folder
- Baris 5 : membuat perintah fungsi looping

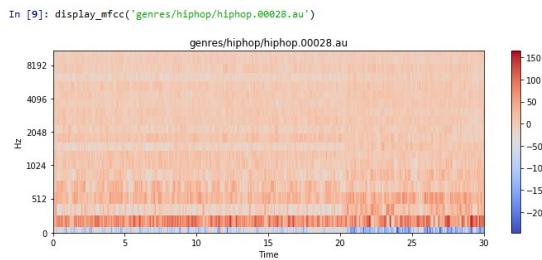


Figure 6.12: Gambar hasil display.

```
In [10]: def extract_features_song(f):
....:     y, _ = librosa.load(f)
....:
....:     # get Mel-frequency cepstral coefficients
....:     mfcc = librosa.feature.mfcc(y)
....:     # normalize values between -1,1 (divide by max)
....:     mfcc /= np.max(np.absolute(mfcc))
....:
....:     return np.ndarray.flatten(mfcc)[:25000]
```

Figure 6.13: Gambar extract feature.

- Baris 6 : membuat atribut sound files yang berisi perintah looping perfolder dari folder genres dan mengambil semua file berekstensi au.
- Baris 7 : memunculkan berapa song.
- Baris 8 : membuat perintah fungsi
- Baris 9 : membuat variabel features untuk memanggil fungsi extract features song (f) sebagai inputan. Setiap satu file array sound files dilakukan ekstrak fitur.
- Baris 10 : memasukkan semua features kedalam all features
- Baris 11 : memasukkan semua genres kedalam all labels
- Baris 12 : medefinisikan label uniq ids dan row ids untuk mengeksekusi perintah unix yang memiliki parameter atribut all labels dan return inverse.
- Baris 13 : Membuat variabel label\_row\_ids untuk menentukan type data 32 byte dari variabel tersebut.
- Baris 14 : Membuat variabel onehot\_labels yang akan mengeksekusi to\_categorical dengan variabel parameter low\_row\_ids dan len(label\_uniq\_ids)
- Baris 15 : Mengembalikan (return) dan menampilkan hasil eksekusi dari variabel parameter all\_features dan onehot\_labels perintah dari np.stack.

## 5. penggunaan fungsi generate\_features\_and\_labels()

```

In [11]: def generate_features_and_labels():
...:     all_features = []
...:     all_labels = []
...:     genres = ['blues', 'classical', 'country', 'disco', 'hiphop', 'jazz', 'metal', 'pop',
...:              'reggae', 'rock']
...:     for genre in genres:
...:         sound_files = glob.glob('genres/*genre/*.au')
...:         print(f'Processing {len(sound_files)} songs in {genre}...')
...:         for f in sound_files:
...:             features = extract_features_song(f)
...:             all_features.append(features)
...:             all_labels.append(genre)
...:     # convert labels to one-hot encoding
...:     label_uniq_ids = np.unique(all_labels, return_inverse=True).ravel()
...:     label_row_ids = label_row_ids.astype(np.int32, copy=False)
...:     onehot_labels = to_categorical(label_row_ids, len(label_uniq_ids))
...:     return np.stack(all_features), onehot_labels

```

Figure 6.14: Gambar Generate fature and label.

Mengapa fungsi generate\_features\_and\_labels() sangat lama ketika meload dataset genre? Itu karena data yang diload sangat banyak sehingga membutuhkan waktu yang banyak

Berikut kode program yang digunakan :

Dari kode program tersebut dapat dijelaskan bahwa kode tersebut digunakan untuk passing parameter dari fitur ekstraksi menggunakan mfcc. Sehingga ketika dijalankan menghasilkan seperti gambar 7.10.

```

Processing 100 songs in blues genre...
Processing 100 songs in classical genre...
Processing 100 songs in country genre...
Processing 100 songs in disco genre...
Processing 100 songs in hiphop genre...
Processing 100 songs in jazz genre...
Processing 100 songs in metal genre...
Processing 100 songs in pop genre...
Processing 100 songs in reggae genre...
Processing 100 songs in rock genre...

```

Figure 6.15: Gambar Generate fture and label.

Gambar 7.10 menjelaskan setelah kode dieksekusi, program memproses 100 song setiap genre musik. Sehingga membutuhkan waktu yang lama.

6. Jelaskan kenapa harus dilakukan pemisahan data training dan data set sebesar 80 persen?

Mengapa harus dilakukan pemisahan data training dan data testing dari dataset sebesar 80 persen, itu karena dataset yang digunakan akan dibuat klasifikasinya.

Berikut adalah kode program yang digunakan :

Penjelasan kode program :

- Membuat klasifikasi dengan plit data sebesar 80 persen data training dan 80 persen data testing.

Setelah kode program tersebut dijalankan, maka akan menghasilkan seperti gambar 6.16:

```
In [13]: training_split = 0.8
```

Figure 6.16: Gambar hasil pemisahan.

## 7. Fungsi Sequential()

Kode program yang digunakan adalah sebagai berikut :

Penjelasan parameter :

- sequential = merupakan sebuah model
- dense 100 = inputan awal neutron adalah 100.
- np.shape = mendapatkan bentuk array.
- relu = menjelaskan untuk inputan yang maksimum yang akan dipilih.
- dense 10 = outputan neutron adalah 10
- softmax = sebuah fungsi yang mengambil input vektor dari bilangan real K, dan menormalkannya menjadi distribusi probabilitas yang terdiri dari probabilitas K.

Setelah program dijalankan, maka akan menghasilkan seperti pada gambar 6.17.

```
In [18]: model = Sequential([
    ...:     Dense(100, input_dim=np.shape(train_input)[1]),
    ...:     Activation('relu'),
    ...:     Dense(10),
    ...:     Activation('softmax'),
    ...: ])
WARNING:tensorflow:From /anaconda3/lib/python3.7/site-packages/
tensorflow/python/framework/op_def_library.py:263: colocate_with
(from tensorflow.python.framework.ops) is deprecated and will be
removed in a future version.
Instructions for updating:
Colocations handled automatically by placer.
```

Figure 6.17: Gambar Sequential.

Penjelasan gambar 6.17:

- (a) Membuat model sequential
- (b) Inputan awal neutron adaldah 100 yang diambil dari data train.
- (c) Melakukan aktivasi dengan parameter fungsi relu.
- (d) Dense(10) merupakan outputan dari neural network yang mengkategorikan 10 kategori jenis genre.

- (e) Melakukan aktivasi dengan menggunakan parameter fungsi softmax.
8. Fungsi compile()

Kode program yang digunakan adalah sebagai berikut :

Penjelasan parameter :

- adam = algoritma optimisasi yang digunakan sebagai ganti dari prosedur penurunan gradien stokastik klasik untuk memperbarui bobot jaringan yang berulang berdasarkan data train.
- categorical\_crossentropy = parameter yang digunakan untuk menyusun sebuah model yang memiliki target dalam format kategorikal.
- accuracy = fungsi metrik yang digunakan untuk menilai kinerja model.

Setelah program dijalankan, maka akan menghasilkan seperti pada gambar 6.18.

```
In [19]: model.compile(optimizer='adam',
...:                 loss='categorical_crossentropy',
...:                 metrics=['accuracy'])
...: print(model.summary())

Layer (type)          Output Shape         Param #
=====
dense_1 (Dense)      (None, 100)          2500100
activation_1 (Activation) (None, 100)          0
dense_2 (Dense)      (None, 10)           1010
activation_2 (Activation) (None, 10)           0
=====
Total params: 2,501,110
Trainable params: 2,501,110
```

Figure 6.18: Gambar hasil compile.

Berikut adalah penjelasan gambar 6.18 :

- Mengcompile model yang sudah dibuat. Dengan menggunakan algoritma optimisasi, fungsi loss, dan fungsi metrik.
- Lalu perintah print digunakan untuk menampilkan hasil yang berupa seperti tabel yang berisi tipe layer, dense, aktivasi, total params dan trainable param.
- Dense\_1 dan activation\_1 menunjukkan neuron inputan sebanyak 100.
- Dense\_2 dan activation\_2 menunjukkan neuron outputan sebanyak 10.

## 9. Fungsi fit()

Kode program yang digunakan adalah sebagai berikut :

Penjelasan parameter pada kode program :

- epochs = fungsi yang digunakan untuk menentukan berapa kali melakukan iterasi.
- batch\_size = fungsi yang digunakan untuk menentukan berapa file dalam satu epochs.
- validation\_split = melakukan pengecekan cross validasi.

Apabila kode program tersebut dijalankan, maka akan menghasilkan seperti pada gambar 6.19.

```
In [20]: model.fit(train_input, train_labels, epochs=10,
batch_size=32,
...:           validation_split=0.2)
WARNING:tensorflow:From /anaconda3/lib/python3.7/site-packages/
tensorflow/python/ops/math_ops.py:3066: to_int32 (from
tensorflow.python.ops.math_ops) is deprecated and will be
removed in a future version.
Instructions for updating:
Use tf.cast instead.
Train on 640 samples, validate on 160 samples
Epoch 1/10
640/640 [=====] - 2s 3ms/step - loss:
2.3108 - acc: 0.2562 - val_loss: 1.9249 - val_acc: 0.3250
Epoch 2/10
640/640 [=====] - 1s 2ms/step - loss:
1.6474 - acc: 0.3891 - val_loss: 1.6401 - val_acc: 0.4250
Epoch 3/10
640/640 [=====] - 2s 3ms/step - loss:
1.3256 - acc: 0.5922 - val_loss: 1.5341 - val_acc: 0.4688
Epoch 4/10
128/640 [=====>.....] - ETA: 1s - loss:
1.1121 - acc: 0.6484
```

Figure 6.19: Gambar fungsi fit.

Penjelasan hasil, perhatikan gambar 6.19 :

- Melakukan pelatihan model dengan menggunakan perintah fit.
- Data yang digunakan adalah data training.
- Kemudian data training dilakukan iterasi atau dijalankan sebanyak 10 kali.
- Dan dalam satu epochs diberlakukan 32 file.
- Lalu diambil 20 persen data untuk dilakukan pengecekan cross validation.

Dari gambar 6.19 kita dapat melihat bahwa setiap satu epoch terdapat nilai akurasi dan loss.

## 10. Fungsi evaluate()

Berikut adalah kode program yang digunakan :

Parameter yang digunakan pada kode program :

- batch\_size = fungsi yang digunakan untuk menentukan berapa file yang digunakan.

Apabila kode program tersebut dijalankan, maka akan menghasilkan seperti pada gambar 6.20.

```
In [21]: loss, acc = model.evaluate(test_input, test_labels,
batch_size=32)
200/200 [=====] - 0s 475us/step
```

Figure 6.20: Gambar fungsi evaluate.

Penjelasan hasil pada gambar 6.20 :

- Membuat parameter loss dan acc untuk mengetahui dan mengevaluasi hasil prediksi data test, berapa kesalahannya dan berapa ketepatannya.

## 11. Fungsi predict()

Berikut adalah kode program yang digunakan :

Parameter yang digunakan pada kode program :

:1 = digunakan untuk menentukan batasa row.

Apabila kode program tersebut dijalankan, maka akan menghasilkan seperti pada gambar 6.21.

```
In [23]: model.predict(test_input[:1])
Out[23]:
array([[2.5113127e-01, 7.7192662e-03, 5.1440198e-02, 6.4860745e-03,
       6.4208927e-03, 6.2923378e-01, 3.7388336e-06, 2.6338635e-02,
       3.9711967e-03, 1.7254945e-02]], dtype=float32)
```

Figure 6.21: Gambar fungsi predict.

Penjelasan hasil pada gambar 6.21 :

- Mengetes hasil prediksi model dari data test\_input dengan batasan 1 row atau lagu nomor satu. Dan hasilnya berupa array dengan tipedata float32.
- Cara bacanya adalah 25 persen dari satu lagu tersebut bergenre blues, 77 persen bergenre classical, dan begitu pula seterusnya.
- Sehingga ditemukan paling tinggi dari lagu tersebut sebesar 64.8 persen yang bergenre disco.

### 6.1.3 Penanganan Error

Dari praktek pemrograman yang dilakukan di modul ini, error yang kita dapatkan(hasil komputer sendiri) di dokumentasikan dan di selesaikan(nilai 5 per error yang ditan-gani. Untuk hari kedua):

1. skrinsut error

(a) Error :

```
FileNotFoundException: [Errno 2] No such file or directory: 'D:\\Python-Artificial-Intelligence-Projects-for-Beginners-master\\Chapter04\\266093_stereo-surgeon_kick-loop-5.wav'
```

Figure 6.22: Gambar Sequential.

2. Tuliskan kode error dan jenis errornya

(a) Kode Error :

- Kode error : FileNotFoundError: [Errno 2] No such file or directory: 'D: Python Artificial-Intelligence-Projects-for-Beginners-master Chapter04 266093 stereo-surgeon kick-loop-5.wav'
- Jenis error : File Not FoundError

3. Solusi pemecahan masalah error

Error yang terjadi pada gambar 6.22 akibat dari kesalahan pada pengalamatan direktori sehingga file yang akan digunakan tidak ditemukan, oleh karena itu pengalamatan direktori disesuaikan dengan kebutuhan penggunaan.

# Chapter 7

## Discussion

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

### 7.1 Andri Fajar Sunandhar / 1164065

#### 7.1.1 Teori

1. Jelaskan kenapa file teks harus dilakukan tokenizer, dilengkapi dengan ilustrasi atau gambar.

Tokenizer adalah untuk membuat vektor dari teks. File teks harus dilakukan tokenizer karena dengan memfungsikan tokenizer teks dapat divektorkan. Maka dari itu harus mengimport tokenizer dari keras, sehingga teks yang telah telah divektorkan tersebut dapat terbaca pada Machine Learning. Ilustrasinya bisa dilihat pada gambar 7.1.

```
from keras.preprocessing.text import Tokenizer
```

Figure 7.1: Tokenizer

2. Menjelaskan konsep dasar K-Fold Cross Validation

```
kfold = StratifiedKFold(n_splits=5)
splits = kfold.split(d, d['CLASS'])
```

Penjelasan kode listingg :

- (a) Membuat variabel kfold yang memanggil fungsi StratifiedKFold. StratifiedKFold itu sendiri ialah variasi Kfold yang mengembalikan lipatan bertingkat. Yang dimana pada kode program tersebut jumlah lipatannya adalah 5 atau dibagi menjadi 5 bagian.

- (b) Membuat variabel split yang mempresentasikan variabel kfold untuk dibagi berdasarkan class.

Berikut adalah gambar 7.2 ilustrasi dari kosep dasar kfold.

```
In [30]: kfolds = StratifiedKFold(n_splits=5)
splits = kfolds.split(d, d['CLASS'])
```

Figure 7.2: StratifiedKFold

3. Jelaskan apa maksudnya kode program for train, test in splits, dilengkapi dengan ilustrasi atau gambar

Maksudnya yaitu untuk menguji apakah setiap data pada dataset sudah di split dan tidak terjadi penumpukan. Yang dimana maksudnya di setiap class tidak akan muncul id yang sama. Ilustrasinya misalkan kita memiliki 5 jam tangan dengan model yang berbeda. Kemudian kita bagikan jam tersebut kepada teman, tentunya teman tersebut yang menerima jam tangan itu tidak memiliki jam yang sama modelnya.

4. Menjelaskan maksud kode program

```
train_content = d[ 'CONTENT' ].iloc [ train_idx ]
test_content = d[ 'CONTENT' ].iloc [ test_idx ]
```

Dari kode program tersebut dapat dijelaskan bahwa membuat fungsi train dan test dengan menggunakan dataset yang hanya diambil kolom 'CONTENT' saja. iloc berfungsi sebagai pengindeksan posisi menggunakan integer.

Berikut ilustrasi dari kode program tersebut, perhatikan gambar 7.3.

```
In [41]: def train_and_test(train_idx, test_idx):
    train_content = d[ "CONTENT" ].iloc[train_idx]
    test_content = d[ "CONTENT" ].iloc[test_idx]
```

Figure 7.3: Ilustrasi Kode Program No.4

5. Apa Maksud Dari Fungsi Tokenizer = Tokenizer(num words=2000) Dan Tokenizer.fit on texts(train content), Dilengkapi Dengan Ilustrasi Gambar

```
tokenizer = Tokenizer ( num_words=2000 )
tokenizer . fit_on_texts ( train_content )
```

```
# Learn the training words (not the testing words!)
tokenizer.fit_on_texts(train_content)
```

Figure 7.4: fungsi tokenizer

Dari kode program membuat variabel tokenizer untuk memanggil fungsi tokenizer agar dapat dilakukan vektorisasi dari kata. Dimana pada kode program pada baris pertama menggunakan 2000 kata atau 2000 kolom.

Berikut adalah gambar 7.4 ilustrasi dari fungsi pada kode program tersebut.

6. Apa Maksud Dari Fungsi d train inputs = tokenizer.texts to matrix(train content, mode='tfidf ') dan d test inputs = tokenizer.texts to matrix(test content, mode='tfidf '), Dilengkapi Dengan Ilustrasi Kode Dan Atau Gambar
  - d\_train\_inputs = tokenizer.texts\_to\_matrix(train\_content , mode='tfidf')
  - d\_test\_inputs = tokenizer.texts\_to\_matrix(test\_content , mode='tfidf')

Dari fungsi pada kode program tersebut dapat dijelaskan bahwa :

- (a) Baris pertama membuat variabel d\_train\_inputs untuk memanggil fungsi tokenizer dan merubah data train yang berupa teks ke dalam bentuk matrix dengan menggunakan model tfidf.
- (b) Baris kedua membuat variabel d\_test\_inputs untuk memanggil fungsi tokenizer dan merubah data test yang berupa teks ke dalam bentuk matrix dengan menggunakan model tfidf.

Berikut adalah gambar 7.5 ilustrasi dari fungsi pada kode program tersebut.

```
# options for mode: binary, freq, tfidf
d_train_inputs = tokenizer.texts_to_matrix(train_content, mode='tfidf')
d_test_inputs = tokenizer.texts_to_matrix(test_content, mode='tfidf')
```

Figure 7.5: fungsi dtrain inputs

7. Apa Maksud Dari Fungsi d train inputs = d train inputs/np.amax(np.absolute(d\_train)) :
  - d\_train\_inputs = d\_train\_inputs/np.amax(np.absolute(d\_train\_inputs))
  - d\_test\_inputs = d\_test\_inputs/np.amax(np.absolute(d\_test\_inputs))

Dari fungsi pada kode program tersebut dapat dijelaskan bahwa fungsi tersebut akan membagi matrix tfidf tadi dengan amax yaitu mengembalikan maksimum array atau maksimum sepanjang sumbu. Yang hasilnya akan dimasukan

kedalam variabel d\_train\_inputs untuk data train dan d\_test\_inputs untuk data test dengan nominal absolut atau tanpa ada bilangan negatif dan koma.

Berikut adalah gambar 7.6 ilustrasi dari fungsi pada kode program tersebut.

```
# divide tfidf by max  
d_train_inputs = d_train_inputs/np.amax(np.absolute(d_train_inputs))  
d_test_inputs = d_test_inputs/np.amax(np.absolute(d_test_inputs))
```

Figure 7.6: dtrain iinputs

8. Jelaskan apa maksud dari fungsi d\_train\_outputs dan d\_test\_outputs, dilengkapi dengan ilustrasi atau gambar.  

```
d_train_outputs = np_utils.to_categorical(d['CLASS'].iloc[train_idx])  
d_test_outputs = np_utils.to_categorical(d['CLASS'].iloc[test_idx])
```

Dari fungsi pada kode program tersebut dapat dijelaskan bahwa fungsi tersebut ditujukan untuk melakukan one-hot encoding supaya bisa masuk dan digunakan pada neural network. One-hot encoding diambil dari 'CLASS' yang berarti hanya terdapat 2 neuron, yaitu satu nol(1,0) atau nol satu(0,1) karena pilihannya hanya ada dua (spam atau bukan).

Berikut adalah gambar 7.7 ilustrasi dari fungsi pada kode program tersebut.

```
d_train_outputs = np_utils.to_categorical(d['CLASS'].iloc[train_idx])  
d_test_outputs = np_utils.to_categorical(d['CLASS'].iloc[test_idx])
```

Figure 7.7: fungsi dtrain outputs

9. Jelaskan apa maksud dari fungsi di listing 7.2 , Gambarkan ilustrasi Neural Networknya.

```
model = Sequential()  
model.add(Dense(512, input_shape=(2000,)))  
model.add(Activation('relu'))  
model.add(Dropout(0.5))  
model.add(Dense(2))  
model.add(Activation('softmax'))
```

Dari fungsi pada kode program tersebut ditujukan untuk melakukan pemodelan dengan sequential, membandingkan setiap satu larik elemen dengan cara satu persatu secara beruntun. Dimana terdapat 512 neuron inputan dengan input shape 2000 vektor yang sudah dinormalisasi. Lalu model dilakukan aktivasi dengan fungsi 'relu'. Kemudian dilakukan pemotongan bobot supaya tidak overfitting sebesar 50 persen dari neuron inputan 512. Lalu pada layer

output terdapat 2 neuron outputan yaitu nol(1,0) atau nol satu(0,1). Kemudian outputan tersebut diaktivasi menggunakan fungsi softmax (mencari nilai maximal).

Berikut adalah gambar 7.8 ilustrasi dari fungsi pada kode program tersebut.

```
7 model = Sequential()
8     model.add(Dense(512, input_shape=(2000,)))
9     model.add(Activation('relu'))
10    model.add(Dropout(0.5))
11    model.add(Dense(2))
12    model.add(Activation('softmax'))
```

Figure 7.8: fungsi di listing 7.2

10. Jelaskan apa maksud dari fungsi di listing 7.3 .

```
model.compile(loss='categorical_crossentropy', optimizer='adamax',
```

Dari fungsi pada kode program tersebut model yang telah dibuat selanjutnya dicompile dengan menggunakan algoritma optimisasi, fungsi loss, dan fungsi metrik.

Berikut adalah gambar 7.9 ilustrasi dari fungsi pada kode program tersebut.

```
8 model.compile(loss='categorical_crossentropy', optimizer='adamax', metrics=['accuracy'])
```

Figure 7.9: fungsi di listing 7.3

11. Jelaskan apa itu Deep Learning.

Deep Learning merupakan cabang dari Machine Learning atau bagian keluarga yang lebih luas dari method machine learning berdasarkan pada representasi data pembelajaran. Deep Learning menggunakan Deep Neural Network dalam menyelesaikan suatu masalah yang terjadi pada Machine Learning.

12. Jelaskan apa itu Deep Neural Network, dan apa bedanya dengan Deep Learning.

Deep Neural Network atau DNN merupakan algoritma yang berbasis neural network yang digunakan untuk mengambil keputusan. Yang membedakan Deep Learning dengan Deep Neural Network (DNN) adalah DNN merupakan algoritma yang digunakan pada Deep Learning, sedangkan Deep Learning merupakan model yang menggunakan algoritma DNN.

13. Jelaskan dengan ilustrasi gambar langkah per langkah bagaimana perhitungan algoritma konvolusi dengan ukuran stride ( $NPM \bmod 3 + 1$ )  $\times$  ( $NPM \bmod 3 + 1$ ) yang terdapat max pooling !

- Pertama tentukan nilai  $(x,y)$  dan  $(x_1,y_1)$
- Nilai tersebut dibuat kedalam bentuk matrix
- Jika sudah berbentuk matrix lakukan perkalian antar baris dan deret
- Hasil perkalian tersebut dijumlahkan sehingga akan menghasilkan nilai matrix  $(3 \times 3)$

Untuk ilustrasi dapat dilihat pada figure 7.10

Dimana saya akan membuat perhitungan matrix kernel yang berukuran  $(3 \times 3)$  dikarenakan  $NPM \bmod 3 + 1 = 3$ . Diperoleh data seperti berikut.

$$f(x,y) \begin{bmatrix} 1 & 2 & 3 \\ 3 & 2 & 1 \\ 4 & 5 & 6 \end{bmatrix} \quad \text{dan } f(x_1,y_1) \begin{bmatrix} 1 & 2 & 3 \\ 2 & 1 & 2 \\ 3 & 2 & 1 \end{bmatrix}$$

Untuk menyelesaikan algoritma konvolusi diperlukan perkalian silang antara  $f(x,y)$  dan  $f(x_1,y_1)$  seperti berikut ini :

- $(1 \times 1) + (2 \times 2) + (3 \times 3) = 1 + 4 + 9 = 14$
- $(1 \times 2) + (2 \times 1) + (3 \times 2) = 2 + 2 + 6 = 10$
- $(1 \times 3) + (2 \times 2) + (3 \times 1) = 3 + 4 + 3 = 10$
- $(3 \times 1) + (2 \times 2) + (1 \times 3) = 3 + 4 + 3 = 10$
- $(3 \times 2) + (2 \times 1) + (1 \times 2) = 6 + 2 + 2 = 10$
- $(3 \times 3) + (2 \times 2) + (1 \times 1) = 9 + 4 + 1 = 14$
- $(4 \times 1) + (5 \times 2) + (6 \times 3) = 4 + 10 + 18 = 32$
- $(4 \times 2) + (5 \times 1) + (6 \times 2) = 8 + 5 + 12 = 25$
- $(4 \times 3) + (5 \times 2) + (6 \times 1) = 12 + 10 + 18 = 40$

Dari hasil perhitungan tersebut terbentuklah matrix  $(3 \times 3)$  sebagai berikut.

$$\begin{bmatrix} 14 & 10 & 10 \\ 10 & 10 & 14 \\ 32 & 25 & 40 \end{bmatrix}$$

Figure 7.10: Algoritma Konvolusi Dengan Matrix  $(3 \times 3)$ .

### 7.1.2 Praktek

1. Jelaskan kode program pada blok # In[1]

Berikut adalah kode program yang digunakan :

Listing 7.1: Kode Program 1

```
import csv
from PIL import Image as pil_image
import keras.preprocessing.image
```

Dari kode listing pada kode program 1, dapat dijelaskan seperti berikut :

- Baris 1 : Melakukan pengimportan file csv
- Baris 2 : Melakukan pemanggilan atau memasukkan module image sebagai pil\_image dari library PIL

- Baris 3 : Melakukan pengimportan fungsi keras.preprocessing.image

Jalankan kode program tersebut, maka menghasilkan seperti pada gambar 7.11.

```
In [1]: import csv
...: from PIL import Image as pil_image
...: import keras.preprocessing.image
Using TensorFlow backend.
```

Figure 7.11: Hasil kode program pada blok 1

2. Jelaskan kode program pada blok # In[2]

Berikut adalah kode program yang digunakan :

Listing 7.2: Kode Program 2

```
imgs = []
classes = []
with open('HASYv2/hasy-data-labels.csv') as csvfile:
    csvreader = csv.reader(csvfile)
    i = 0
    for row in csvreader:
        if i > 0:
            img = keras.preprocessing.image.img_to_array(pil_image.
                # neuron activation functions behave best when input va
                # so we rescale each pixel value to be in the range 0.0
                img /= 255.0
            imgs.append((row[0], row[2], img))
            classes.append(row[2])
        i += 1
```

Dari kode listing pada kode program 2, dapat dijelaskan seperti berikut :

- Baris 1 : Membuat variabel imgs tanpa ada parameter di dalamnya
- Baris 2 : Membuat variabel classes tanpa ada parameter didalamnya
- Baris 3 : Membuka file csv dari HASYv2/hasy-data-labels.csv sebagai csv-file
- Baris 4 : Membuat variabel csvreader yang difungsikan untuk membaca dari file csv yang dimasukkan
- Baris 5 : Membuat variabel i dengan parameter 0 atau nilai 0
- Baris 6 : Digunakan untuk melakukan eksekusi baris dari pembacaan csv

- Baris 7 : Mengaplikasikan atau menggunakan perintah "if" dengan variabel i lebih besar dari 0, yang selanjutnya akan dilanjutkan ke perintah berikutnya
- Baris 8 : Membuat variabel img yang berfungsi untuk mengubah image atau gambar menjadi bentuk array (bilangan) dari file HASYv2 yang dibuka dengan row berparameter 0.
- Baris 9 : Membuat variabel img dengan nilai bukan sama dengan 255.0
- Baris 10 : Mendefinisikan fungsi imgs.append yang digunakan untuk melakukan proses penggabungan data dengan file lain atau dataset lain yang telah ditentukan dengan 3 parameter yaitu row[0], row[2] dan variabel img.
- Baris 11 : Mendefinisikan fungsi append dari variabel classes dengan menggunakan parameter row[2].
- Baris 12 : Mengartikan  $i = i + 1$  yang dimana nilai sari variabel i akan ditambah 1 sehingga akan bernilai 1.

Jalankan kode program tersebut, maka menghasilkan seperti pada gambar 7.12.

```
In [2]: imgs = []
...: classes = []
...: with open('HASYv2/hasy-data-labels.csv') as csvfile:
...:     csvreader = csv.reader(csvfile)
...:     i = 0
...:     for row in csvreader:
...:         if i > 0:
...:             img = keras.preprocessing.image.img_to_array(pil_image.open("HASYv2/" + row[0]))
...:             # neuron activation functions behave best when input values are between 0.0 and 1.0
...:             # (or -1.0 and 1.0),
...:             # so we rescale each pixel value to be in the range 0.0 to 1.0 instead of 0-255
...:             img /= 255.0
...:             imgs.append((row[0], row[2], img))
...:             classes.append(row[2])
...:         i += 1
```

Figure 7.12: Hasil kode program pada blok 2

### 3. Jelaskan kode program pada blok # In[3]

Berikut adalah kode program yang digunakan :

Listing 7.3: Kode Program 3

```
import random
random.shuffle(imgs)
split_idx = int(0.8*len(imgs))
train = imgs[:split_idx]
test = imgs[split_idx:]
```

Dari kode listing pada kode program 3, dapat dijelaskan seperti berikut :

- Baris 1 : Memanggil dan menggunakan module random
- Baris 2 : Melakukan pengocokan menggunakan module random pada parameter variabel imgs
- Baris 3 : Membagi index data kedalam bentuk integer dengan mengalikan 0,8 dan len yang berfungsi mengembalikan jumlah item dalam datanya dari variabel imgs
- Baris 4 : Membuat variabel train yang digunakan untuk mengeksekusi imgs serta pemecahan index awal pada data untuk digunakan sebagai data training
- Baris 5 : Membuat variabel test yang digunakan untuk mengeksekusi imgs serta pemecahan index akhir pada data untuk digunakan sebagai data testing

Jalankan kode program tersebut, maka menghasilkan seperti pada gambar 7.13.

```
In [3]: import random
...: random.shuffle(imgs)
...: split_idx = int(0.8*len(imgs))
...: train = imgs[:split_idx]
...: test = imgs[split_idx:]
```

Figure 7.13: Hasil kode program pada blok 3

#### 4. Jelaskan kode program pada blok # In[4]

Berikut adalah kode program yang digunakan :

Listing 7.4: Kode Program 4

```
import numpy as np

train_input = np.asarray(list(map(lambda row: row[2], train)))
test_input = np.asarray(list(map(lambda row: row[2], test)))

train_output = np.asarray(list(map(lambda row: row[1], train)))
test_output = np.asarray(list(map(lambda row: row[1], test)))
```

Dari kode listing pada kode program 4, dapat dijelaskan seperti berikut :

- Baris 1 : Melakukan import library numpy sebagai np

- Baris 2 : Membuat variabel train\_input untuk mengubah inputan menjadi array menggunakan fungsi np.asarray dan fungsi list untuk mengkoleksi data yang dipilih serta data dapat diubah. Dan didalamnya melakukan penerapan fungsi map yang berfungsi untuk mengembalikan iterator dari data yang digunakan dan fungsi lamda pada row berparameter [2] difungsikan untuk membuat objek menjadi lebih kecil sehingga mudah dieksekusi dari variabel train.
- Baris 3 : Membuat variabel test\_input untuk mengubah inputan menjadi array menggunakan fungsi np.asarray dan fungsi list untuk mengkoleksi data yang dipilih serta data dapat diubah. Dan didalamnya melakukan penerapan fungsi map yang berfungsi untuk mengembalikan iterator dari data yang digunakan dan fungsi lamda pada row berparameter [2] difungsikan untuk membuat objek menjadi lebih kecil sehingga mudah dieksekusi dari variabel test.
- Baris 4 : Membuat variabel train\_input untuk mengubah inputan menjadi array menggunakan fungsi np.asarray dan fungsi list untuk mengkoleksi data yang dipilih serta data dapat diubah. Dan didalamnya melakukan penerapan fungsi map yang berfungsi untuk mengembalikan iterator dari data yang digunakan dan fungsi lamda pada row berparameter [1] difungsikan untuk membuat objek menjadi lebih kecil sehingga mudah dieksekusi dari variabel train.
- Baris 5 : Membuat variabel test\_input untuk mengubah inputan menjadi array menggunakan fungsi np.asarray dan fungsi list untuk mengkoleksi data yang dipilih serta data dapat diubah. Dan didalamnya melakukan penerapan fungsi map yang berfungsi untuk mengembalikan iterator dari data yang digunakan dan fungsi lamda pada row berparameter [1] difungsikan untuk membuat objek menjadi lebih kecil sehingga mudah dieksekusi dari variabel test.

Jalankan kode program tersebut, maka menghasilkan seperti pada gambar 7.14.

```
In [4]: import numpy as np
...
...: train_input = np.asarray(list(map(lambda row: row[2], train)))
...: test_input = np.asarray(list(map(lambda row: row[2], test)))
...
...: train_output = np.asarray(list(map(lambda row: row[1], train)))
...: test_output = np.asarray(list(map(lambda row: row[1], test)))
```

Figure 7.14: Hasil kode program pada blok 4

5. Jelaskan kode program pada blok # In[5]

Berikut adalah kode program yang digunakan :

Listing 7.5: Kode Program 5

```
from sklearn.preprocessing import LabelEncoder  
from sklearn.preprocessing import OneHotEncoder
```

Dari kode listing pada kode program 5, dapat dijelaskan seperti berikut :

- Baris 1 : Menggunakan fungsi LabelEncoder dari sklearn.preprocessing yang berfungsi untuk menormalkan label dimana label encoder hanya didefinisikan dengan nilai antara 0 dan -1.
- Baris 2 : Menggunakan fungsi OneHotEncoder dari sklearn.preprocessing yang berfungsi untuk mendefinisikan fitur input yang dimana mengambil nilai dalam kisaran [0, nilai maksimal].

Jalankan kode program tersebut, maka menghasilkan seperti pada gambar 7.15.

```
In [5]: from sklearn.preprocessing import LabelEncoder  
...: from sklearn.preprocessing import OneHotEncoder
```

Figure 7.15: Hasil kode program pada blok 5

6. Jelaskan kode program pada blok # In[6]

Berikut adalah kode program yang digunakan :

Listing 7.6: Kode Program 6

```
label_encoder = LabelEncoder()  
integer_encoded = label_encoder.fit_transform(classes)
```

Dari kode listing pada kode program 6, dapat dijelaskan seperti berikut :

- Baris 1 : Membuat variabel label\_encoder dengan penerapan modul / fungsi LabelEncoder tanpa parameter
- Baris 2 : Membuat variabel integer\_encoded dengan penerapan fungsi label\_encoder.fit\_transform yang berfungsi untuk melakukan ekstrasi fitur object dari variabel classes yang akan mengembalikan beberapa data yang diubah kembali.

Jalankan kode program tersebut, maka menghasilkan seperti pada gambar 7.16.

```
In [6]: label_encoder = LabelEncoder()
...: integer_encoded = label_encoder.fit_transform(classes)
```

Figure 7.16: Hasil kode program pada blok 6

7. Jelaskan kode program pada blok # In[7]

Berikut adalah kode program yang digunakan :

Listing 7.7: Kode Program 7

```
onehot_encoder = OneHotEncoder(sparse=False)
integer_encoded = integer_encoded.reshape(len(integer_encoded), 1)
onehot_encoder.fit(integer_encoded)
```

Dari kode listing pada kode program 7, dapat dijelaskan seperti berikut :

- Baris 1 : Membuat variabel onehot\_encoder yang memanggil fungsi OneHotEncoder tanpa mengembalikan matriks karena sparse=false.
- Baris 2 : Membuat variabel integer\_encoded memanggil variabel integer\_encoded pada kode program 6 untuk dieksekusi memberikan bentuk baru ke array tanpa mengubah datanya dari mengembalikan panjang nilai dari integer\_encoded.
- Baris 3 : Onehotencoding melakukan fitting pada integer\_encoded.

Jalankan kode program tersebut, maka menghasilkan seperti pada gambar 7.17.

```
In [7]: onehot_encoder = OneHotEncoder(sparse=False)
...: integer_encoded = integer_encoded.reshape(len(integer_encoded), 1)
...: onehot_encoder.fit(integer_encoded)
C:\Users\lsvapr\Anaconda3\lib\site-packages\sklearn\preprocessing\_encoders.py:371: FutureWarning: The handling of integer data will change in version 0.22. Currently, the categories are determined based on the range [0, max(values)], while in the future they will be determined based on the unique values.
If you want the future behaviour and silence this warning, you can specify "categories='auto'".
In case you used a LabelEncoder before this OneHotEncoder to convert the categories to integers, then you can now use the OneHotEncoder directly.
warnings.warn(msg, FutureWarning)
Out[7]:
OneHotEncoder(categorical_features=None, categories=None,
               dtype=<class 'numpy.float64'>, handle_unknown='error',
               n_values=None, sparse=False)
```

Figure 7.17: Hasil kode program pada blok 7

8. Jelaskan kode program pada blok # In[8]

Berikut adalah kode program yang digunakan :

Listing 7.8: Kode Program 8

```
train_output_int = label_encoder.transform(train_output)
train_output = onehot_encoder.transform(train_output_int.reshape(len(train_output_int), 1))
test_output_int = label_encoder.transform(test_output)
test_output = onehot_encoder.transform(test_output_int.reshape(len(test_output_int), 1))

num_classes = len(label_encoder.classes_)
print("Number of classes: %d" % num_classes)
```

Dari kode listing pada kode program 8, dapat dijelaskan seperti berikut :

- Baris 1 : Membuat variabel train\_output\_int yang mengeksekusi label\_encoder dengan mengubah nilai dari parameter variabel train\_output.
- Baris 2 : Membuat variabel train\_output yang mengeksekusi variabel one-hot\_encoder dari kode program 7 dengan mengubah nilai dari variabel parameter train\_output\_int yang datanya sudah diubah kedalam bentuk array dan panjang nilai dari train\_output\_int telah dikembalikan.
- Baris 3 : Membuat variabel test\_output\_int yang mengeksekusi label\_encoder dengan mengubah nilai dari parameter variabel test\_output.
- Baris 4 : Membuat variabel test\_output yang mengeksekusi variabel one-hot\_encoder dari kode program 7 dengan mengubah nilai dari variabel parameter test\_output\_int yang datanya sudah diubah kedalam bentuk array dan panjang nilai dari test\_output\_int telah dikembalikan.
- Baris 5 : Membuat variabel num\_classes untuk mengetahui jumlah class dari label\_encoder
- Baris 6 : Perintah print digunakan untuk memunculkan hasil dari variabel num\_classes

Jalankan kode program tersebut, maka menghasilkan seperti pada gambar 7.18.

```
In [8]: train_output_int = label_encoder.transform(train_output)
...: train_output = onehot_encoder.transform(train_output_int.reshape(len(train_output_int), 1))
...: test_output_int = label_encoder.transform(test_output)
...: test_output = onehot_encoder.transform(test_output_int.reshape(len(test_output_int), 1))
...:
...: num_classes = len(label_encoder.classes_)
...: print("Number of classes: %d" % num_classes)
Number of classes: 369
```

Figure 7.18: Hasil kode program pada blok 8

9. Jelaskan kode program pada blok # In[9]

Berikut adalah kode program yang digunakan :

Listing 7.9: Kode Program 9

```
from keras.models import Sequential
from keras.layers import Dense, Dropout, Flatten
from keras.layers import Conv2D, MaxPooling2D
```

Dari kode listing pada kode program 9, dapat dijelaskan seperti berikut :

- Baris 1 : Memanggil atau melakukan importing fungsi model sequential dari library keras.
- Baris 2 : Memanggil atau melakukan importing fungsi layer dense, dropout, dan flatten dari library keras.
- Baris 3 : Memanggil atau melakukan importing fungsi layer Conv2D dan MaxPooling2D dari library keras.

Jalankan kode program tersebut, maka menghasilkan seperti pada gambar 7.19.

```
In [9]: from keras.models import Sequential
...: from keras.layers import Dense, Dropout, Flatten
...: from keras.layers import Conv2D, MaxPooling2D
```

Figure 7.19: Hasil kode program pada blok 9

10. Jelaskan kode program pada blok # In[10]

Berikut adalah kode program yang digunakan :

Listing 7.10: Kode Program 10

```
model = Sequential()
model.add(Conv2D(32, kernel_size=(3, 3), activation='relu',
                 input_shape=np.shape(train_input[0])))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Conv2D(32, (3, 3), activation='relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Flatten())
model.add(Dense(1024, activation='tanh'))
model.add(Dropout(0.5))
model.add(Dense(num_classes, activation='softmax'))
```

```

model.compile(loss='categorical_crossentropy', optimizer='adam',
               metrics=['accuracy'])

print(model.summary())

```

Dari kode listing pada kode program 10, dapat dijelaskan seperti berikut :

- Baris 1 : Melakukan pemodelan Sequential.
- Baris 2 : Menambahkan Konvolusi 2D dengan 32 filter konvolusi menggunakan matriks kernel berukuran 3x3 dengan algoritma activation adalah relu, dimana data diambil dari train\_input yang dimulai dari baris ke-nol.
- Baris 3 : Menambahkan Max Pooling dengan matriks 2x2.
- Baris 4 : Melakukan penambahan konvolusi 2D lagi dengan 32 filter konvolusi menggunakan matrik kernel masing-masing berukuran 3x3 dengan algoritam activation relu.
- Baris 5 : Melakukan penambahan lagi Max Pooling dengan matriks 2x2.
- Baris 6 : Melakukan perataan pada inputan model.
- Baris 7 : Mendefinisikan inputan dengan 1024 neuron dan melakukan aktivasi menggunakan algoritma tanh.
- Baris 8 : Dropout dilakukan untuk melakukan pemotongan pada cabang yang terdiri dari pengaturan secara acak tingkat pecahan unit input ke 0 pada setiap pembaruan selama waktu pelatihan, yang membantu mencegah overfitting sebesar 50%.
- Baris 9 : Pada output layer menggunakan data dari variabel num\_classes dan menggunakan fungsi aktivasi softmax.
- Baris 10 : Melakukan konfigurasi proses pembelajaran, yang dilakukan melalui metode compile, sebelum melatih suatu model.
- Baris 11 : Menampilkan hasil representasi ringkasan dari model yang telah dibuat.

Jalankan kode program tersebut, maka menghasilkan seperti pada gambar ??.

#### 11. Jelaskan kode program pada blok # In[11]

Berikut adalah kode program yang digunakan :

```

In [10]: model = Sequential()
...:     model.add(Conv2D(32, kernel_size=(3, 3), activation='relu',
...:                     input_shape=inp.shape[train_input[0]]))
...:     model.add(MaxPooling2D(pool_size=(2, 2)))
...:     model.add(Conv2D(64, kernel_size=(3, 3), activation='relu'))
...:     model.add(MaxPooling2D(pool_size=(2, 2)))
...:     model.add(Flatten())
...:     model.add(Dense(1024, activation='tanh'))
...:     model.add(Dropout(0.5))
...:     model.add(Dense(num_classes, activation='softmax'))
...: 
...:     model.compile(loss='categorical_crossentropy', optimizer='adam',
...:                   metrics=['accuracy'])
...: 
...: print(model.summary())

```

WARNING:tensorflow:From C:\Users\liva\r\anaconda3\lib\site-packages\tensorflow\python\framework\ops\_def\_library.py:263: colocate\_with (from tensorflow.python.ops.ops) is deprecated and will be removed in a future version.  
 Instructions for updating:  
 Colocate operations explicitly by placing  
 colocate\_with as the first argument.  
 WARNING:tensorflow:From C:\Users\liva\r\anaconda3\lib\site-packages\keras\backend\tensorflow\_backend.py:3446: calling dropout (from tensorflow.python.ops.nn\_ops) with keep\_prob is deprecated and will be removed in a future version.  
 Instructions for updating:  
 Please use rate instead of 'keep\_prob'. Rate should be set to 'rate = 1 - keep\_prob'.

Layer (Type)	Output Shape	Param #
conv2d_1 (Conv2D)	(None, 30, 30, 32)	896
max_pooling2d_1 (MaxPooling2D)	(None, 15, 15, 32)	0
conv2d_2 (Conv2D)	(None, 13, 13, 32)	9248
max_pooling2d_2 (MaxPooling2D)	(None, 6, 6, 32)	0
flatten_1 (Flatten)	(None, 1152)	0
dense_1 (Dense)	(None, 1024)	1180672
dropout_1 (Dropout)	(None, 1024)	0
dense_2 (Dense)	(None, 369)	378225

Total params: 1,569,041  
 Trainable params: 1,569,041  
 Non-trainable params: 0

Figure 7.20: Hasil kode program pada blok 10

Listing 7.11: Kode Program 11

```

import keras.callbacks
tensorboard = keras.callbacks.TensorBoard(log_dir='./logs/mnist-style')

```

Dari kode listing pada kode program 11, dapat dijelaskan seperti berikut :

- Baris 1 : Melakukan importing library keras.callbacks yang memiliki fungsi penulisan log untuk TensorBoard, yang memungkinkan untuk memvisualisasikan grafik dinamis dari training dan metrik pengujian.
- Baris 2 : Membuat variabel tensorboard yang menggunakan fungsi TensorBoard dari keras.callbacks yang berfungsi sebagai alat visualisasi yang telah disediakan oleh TensorFlow. Kemudian untuk fungsi log\_dir memanggil data yaitu './logs/mnist-style' dari direktori.

Jalankan kode program tersebut, maka menghasilkan seperti pada gambar 7.21.

```

In [11]: import keras.callbacks
...: tensorboard = keras.callbacks.TensorBoard(log_dir='./logs/mnist-style')

```

Figure 7.21: Hasil kode program pada blok 11

## 12. Jelaskan kode program pada blok # In[12]

Berikut adalah kode program yang digunakan :

Listing 7.12: Kode Program 12

```
model.fit(train_input, train_output,  
          batch_size=32,  
          epochs=10,  
          verbose=2,  
          validation_split=0.2,  
          callbacks=[tensorboard])  
  
score = model.evaluate(test_input, test_output, verbose=2)  
print('Test loss:', score[0])  
print('Test accuracy:', score[1])
```

Dari kode listing pada kode program 12, dapat dijelaskan seperti berikut :

- Baris 1 : Menerapkan fungsi `model.fit` yang didalamnya memproses `train_input`, `train_output` dengan `batch_size`, `epochs`, `verbose`, `validation_split`, dan `callbacks`.
  - `Batch_size` merupakan jumlah sampel per pembaharuan sampel dari data yang diolah, sehingga apabila `batch_size`nya tidak ditemukan maka otomatis akan dijadikan nilai 32.
  - `Epochs` berfungsi untuk melakukan perulangan dimana perulangan dari berapa kali nilai yang digunakan untuk data, dan jumlahnya ialah 10.
  - `Verbose` digunakan sebagai opsi untuk menghasilkan informasi logging dari data yang ditentukan dengan nilai 2.
  - `Validation_split` berfungsi untuk memecah nilai dari perhitungan dengan validasinya sebesar 0,2. (Fraksi data pelatihan untuk digunakan sebagai data validasi).
  - `Callsbacks` mengeksekusi `tensorboard` yang berfungsi untuk memvisualisasikan parameter training, metrik, hiperparameter pada nilai/data yang diproses.
- Baris 2 : Membuat variabel `score` dengan menggunakan fungsi `evaluate` dari model yang ada dengan variabel parameter `test_input`, `tst_output` dan `verbose=2` yang berfungsi memprediksi output untuk input yang diberikan dan kemudian menghitung fungsi metrik yang ditentukan dalam modelnya.
- Baris 3 : Mencetak variabel `score` optimasi dari test dengan ketentuan nilai parameter 0

- Baris 4 : Mencetak variabel score akurasi dari test dengan ketentuan nilai parameter 1

Jalankan kode program tersebut, maka menghasilkan seperti pada gambar 7.22.

```
In [12]: model.fit(main_input, train_output,
...:     batch_size=32,
...:     epochs=10,
...:     verbose=2,
...:     validation_split=0.2,
...:     callbacks=[tensorboard])
...:
...: score = model.evaluate(test_input, test_output, verbose=2)
...: print('Test loss:', score[0])
...: print('Test accuracy:', score[1])
WARNING:tensorflow:From C:\Users\liva\rp\anaconda3\lib\site-packages\tensorflow\python\ops\math_ops.py:3066: to_int32 (from tensorflow.python.ops.math_ops) is deprecated and will be removed in a future version.
Instructions for updating:
Use tf.cast instead.
Train on 107668 samples, validate on 26918 samples
Epoch 1/10
  2/21 - loss: 1.5626 - acc: 0.6232 - val_loss: 1.0003 - val_acc: 0.7257
Epoch 2/10
  2/25 - loss: 0.9856 - acc: 0.7277 - val_loss: 0.9087 - val_acc: 0.7443
Epoch 3/10
  2/19 - loss: 0.8764 - acc: 0.7505 - val_loss: 0.8710 - val_acc: 0.7556
Epoch 4/10
  2/19 - loss: 0.8085 - acc: 0.7679 - val_loss: 0.8437 - val_acc: 0.7564
Epoch 5/10
  2/19 - loss: 0.7533 - acc: 0.7770 - val_loss: 0.8318 - val_acc: 0.7684
Epoch 6/10
  2/18 - loss: 0.7891 - acc: 0.7866 - val_loss: 0.8451 - val_acc: 0.7617
Epoch 7/10
  2/44 - loss: 0.6774 - acc: 0.7927 - val_loss: 0.8559 - val_acc: 0.7622
Epoch 8/10
  2/55 - loss: 0.6447 - acc: 0.8000 - val_loss: 0.8500 - val_acc: 0.7603
Epoch 9/10
  2/76 - loss: 0.6219 - acc: 0.8054 - val_loss: 0.8877 - val_acc: 0.7621
Epoch 10/10
  2/21 - loss: 0.6095 - acc: 0.8092 - val_loss: 0.8827 - val_acc: 0.7603
Test loss: 0.5731205600954306
Test accuracy: 0.7642269887794817
```

Figure 7.22: Hasil kode program pada blok 12

### 13. Jelaskan kode program pada blok # In[13]

Berikut adalah kode program yang digunakan :

Listing 7.13: Kode Program 13

```
import time

results = []
for conv2d_count in [1, 2]:
    for dense_size in [128, 256, 512, 1024, 2048]:
        for dropout in [0.0, 0.25, 0.50, 0.75]:
            model = Sequential()
            for i in range(conv2d_count):
                if i == 0:
                    model.add(Conv2D(32, kernel_size=(3, 3), activation='relu'))
                else:
                    model.add(Conv2D(32, kernel_size=(3, 3), activation='relu'))
                    model.add(MaxPooling2D(pool_size=(2, 2)))
            model.add(Flatten())
            model.add(Dense(dense_size, activation='tanh'))
            if dropout > 0.0:
                model.add(Dropout(dropout))
            model.add(Dense(num_classes, activation='softmax'))
```

Dari kode listing pada kode program 13, dapat dijelaskan seperti berikut :

- Baris 1 : Melakukan importing modul time.
- Baris 2 : Membuat variabel result berisikan array kosong.
- Baris 3 : Menggunakan convolution 2D yang dimana akan memiliki 1 atau 2 layer
- Baris 4 : Mendefinisikan dense\_size dengan ukuran 128, 256, 512, 1024, 2048
- Baris 5 : Mendefinisikan drop\_out dengan 0, 25%, 50%, dan 75%
- Baris 6 : Melakukan pemodelan Sequential
- Baris 7 : Untuk i dalam cakupan conv2d\_count
- Baris 8 : Jika ini adalah layer pertama, kita perlu memasukkan bentuk input.
- Baris 9 : Kalau tidak kita hanya akan menambahkan layer.
- Baris 10 : Kemudian, setelah menambahkan layer konvolusi, akan dilakukan hal yang sama dengan max pooling.
- Baris 11 : Lalu, melakukan perataan atau flatten dan menambahkan dense size ukuran apa pun yang berasal dari dense\_size. Dimana akan selalu menggunakan algoritma tanh.
- Baris 12 : Jika dropout digunakan, maka akan menambahkan layer dropout. Katakanlah 50% dilakukan dropout, bahwa setiap kali ia memperbarui bobot setelah setiap batch, ada peluang 50% untuk setiap bobot yang tidak akan diperbarui dan menempatkannya di antara dua lapisan padat untuk diaktivasi serta melindunginya dari overfitting.
- Baris 13 : Lapisan terakhir akan selalu menjadi jumlah kelas karena itu harus, dan menggunakan softmax. Dan dikompilasi dengan cara yang sama.
- Baris 14 : Mengatur direktori log yang berbeda untuk TensorBoard sehingga dapat membedakan konfigurasi yang berbeda.
- Baris 15 : Variabel start akan memanggil modul time atau waktu
- Baris 16 : Melakukan fit atau compile
- Baris 17 : Melakukan scoring dengan .evaluate yang berfungsi menampilkan data loss dan accuracy dari model

- Baris 18 : end merupakan variabel untuk melihat waktu akhir pada saat pemodelan berhasil dilakukan.
- Baris 19 : Menampilkan hasil dari run skrip

Jalankan kode program tersebut, maka menghasilkan seperti pada gambar 7.23.

```
In [13]: import time
...
...: results = []
...: for conv2d_count in [1, 2]:
...:     for dense_size in [128, 256, 512, 1024, 2048]:
...:         for dropout in [0.0, 0.25, 0.50, 0.75]:
...:             model = Sequential()
...:             for i in range(conv2d_count):
...:                 if i == 0:
...:                     model.add(Conv2D(32, kernel_size=(3, 3), activation='relu', input_shape=np.shape(train_input[0])))
...:                 else:
...:                     model.add(Conv2D(32, kernel_size=(3, 3), activation='relu'))
...:             model.add(Flatten())
...:             model.add(Dense(dense_size, activation='tanh'))
...:             if dropout > 0.0:
...:                 model.add(Dropout(dropout))
...:             model.add(Dense(num_classes, activation='softmax'))
...:             model.compile(loss='categorical_crossentropy', optimizer='adam', metrics=['accuracy'])
...:             log_dir = './Logs/conv2d_%d-dense_%d-dropout_%d' % (conv2d_count, dense_size, dropout)
...:             tensorboard = keras.callbacks.TensorBoard(log_dir=log_dir)
...:
...:             start = time.time()
...:             model.fit(train_input, train_output, batch_size=32, epochs=10,
...:                       verbose=0, validation_split=0.2, callbacks=[tensorboard])
...:             score = model.evaluate(test_input, test_output, verbose=2)
...:             end = time.time()
...:
...:             elapsed = end - start
...:
...:             print("Conv2D count: %d, Dense size: %d, Dropout: %.2f - Loss: %.2f, Accuracy: %.2f, Time: %.2f sec" % (conv2d_count, dense_size, dropout,
...: score[0], score[1], elapsed))
...:             results.append((conv2d_count, dense_size, dropout, score[0], score[1], elapsed))
Conv2D count: 1, Dense size: 128, Dropout: 0.00 - Loss: 1.16, Accuracy: 0.74, Time: 1478 sec
Conv2D count: 1, Dense size: 128, Dropout: 0.25 - Loss: 0.94, Accuracy: 0.76, Time: 1533 sec
Conv2D count: 1, Dense size: 128, Dropout: 0.50 - Loss: 0.89, Accuracy: 0.78, Time: 1525 sec
Conv2D count: 1, Dense size: 128, Dropout: 0.75 - Loss: 0.77, Accuracy: 0.81, Time: 1530 sec
Conv2D count: 1, Dense size: 256, Dropout: 0.00 - Loss: 1.32, Accuracy: 0.76, Time: 2030 sec
Conv2D count: 1, Dense size: 256, Dropout: 0.25 - Loss: 1.12, Accuracy: 0.76, Time: 2030 sec
Conv2D count: 1, Dense size: 256, Dropout: 0.50 - Loss: 0.98, Accuracy: 0.78, Time: 2047 sec
```

Figure 7.23: Hasil kode program pada blok 13

#### 14. Jelaskan kode program pada blok # In[14]

Berikut adalah kode program yang digunakan :

Listing 7.14: Kode Program 14

```
model = Sequential()
model.add(Conv2D(32, kernel_size=(3, 3), activation='relu', input_size=(28, 28, 3)))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Conv2D(32, (3, 3), activation='relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Flatten())
model.add(Dense(128, activation='tanh'))
model.add(Dropout(0.5))
model.add(Dense(num_classes, activation='softmax'))
model.compile(loss='categorical_crossentropy', optimizer='adam', metrics=['accuracy'])
print(model.summary())
```

Dari kode listing pada kode program 14, dapat dijelaskan seperti berikut :

- Baris 1: Membuat model dengan menggunakan pemodelan Sequential

- Baris 2: Pada layer pertama melakukan penambahan Convolutio 2D dengan dimensi 32, dan ukuran matriks kernel 3x3 dengan menggunakan fungsi aktivasi relu dan menampilkan input\_shape
- Baris 3: Melakukan Max Pooling 2D dengan ukuran matriks 2x2
- Baris 4: Pada layer kedua, dilakukan convolusi lagi dengan kriteria yang sama tanpa ada penambahan input, hal ini dilakukan untuk mendapatkan data terbaik
- Baris 5: Melakukan Flatten untuk meratakan inputan
- Baris 6: Menambahkan dense input sebanyak 128 neuron dengan menggunakan fungsi aktivasi tanh.
- Baris 7: Melakukan Dropout sebanyak 50% untuk menghindari overfitting
- Baris 8: Menambahkan dense pada model untuk output dimana layer ini akan menjadi jumlah dari class yang ada.
- Baris 9: Melakukan compiling dari model yang telah didefinisikan
- Baris 10: Menampilkan ringkasan dari pemodelan yang dilakukan

Jalankan kode program tersebut, maka menghasilkan seperti pada gambar 7.24.

```
In [14]: model = Sequential()
...: model.add(Conv2D(32, kernel_size=(3, 3), activation='relu', input_shape=np.shape(train_input[0])))
...: model.add((MaxPooling2D(pool_size=(2, 2)))
...: model.add((Conv2D(32, (3, 3), activation='relu'))
...: model.add((MaxPooling2D(pool_size=(2, 2)))
...: model.add(Flatten())
...: model.add(Dense(128, activation='tanh'))
...: model.add(Dropout(0.5))
...: model.add(Dense(num_classes, activation='softmax'))
...: model.compile(loss='categorical_crossentropy', optimizer='adam', metrics=['accuracy'])
...: print(model.summary())
None
```

Layer (type)	Output Shape	Param #
conv2d_11 (Conv2D)	(None, 15, 15, 32)	896
max_pooling2d_11 (MaxPooling2D)	(None, 7, 7, 32)	0
conv2d_12 (Conv2D)	(None, 7, 7, 32)	9248
max_pooling2d_12 (MaxPooling2D)	(None, 3, 3, 32)	0
flatten_10 (Flatten)	(None, 36)	0
dense_19 (Dense)	(None, 128)	16384
dropout_8 (Dropout)	(None, 128)	0
dense_20 (Dense)	(None, 36)	4326

Total params: 205,329  
Trainable params: 205,329  
Non-trainable params: 0

Figure 7.24: Hasil kode program pada blok 14

### 15. Jelaskan kode program pada blok # In[15]

Berikut adalah kode program yang digunakan :

Listing 7.15: Kode Program 15

```
model.fit(np.concatenate((train_input, test_input)),  
         np.concatenate((train_output, test_output)),  
         batch_size=32, epochs=10, verbose=2)
```

Dari kode listing pada kode program 15, dapat dijelaskan seperti berikut :

- Baris 1: Melakukan fit atau fitting (pencocokan data) dengan penggabungan dari data train dan test agar dapat dilakukan pelatihan untuk jaringan pada semua data yang dimiliki.

Jalankan kode program tersebut, maka menghasilkan seperti pada gambar 7.25.

```
In [15]: model.fit(np.concatenate((train_input, test_input)),  
...:  
...:  
...:  
Epoch 1/10  
- 247s - loss: 1.7758 - acc: 0.5868  
Epoch 2/10  
- 227s - loss: 1.0771 - acc: 0.7063  
Epoch 3/10  
- 228s - loss: 0.9602 - acc: 0.7312  
Epoch 4/10  
- 229s - loss: 0.9026 - acc: 0.7441  
Epoch 5/10  
- 229s - loss: 0.8662 - acc: 0.7515  
Epoch 6/10  
- 229s - loss: 0.8316 - acc: 0.7597  
Epoch 7/10  
- 228s - loss: 0.8081 - acc: 0.7632  
Epoch 8/10  
- 229s - loss: 0.7907 - acc: 0.7664  
Epoch 9/10  
- 230s - loss: 0.7750 - acc: 0.7705  
Epoch 10/10  
- 230s - loss: 0.7630 - acc: 0.7723  
Out[15]: <keras.callbacks.History at 0x22c12d05b70>
```

Figure 7.25: Hasil kode program pada blok 15

16. Jelaskan kode program pada blok # In[16]

Berikut adalah kode program yang digunakan :

Listing 7.16: Kode Program 16

```
model.save("mathsymbols.model")
```

Dari kode listing pada kode program 16, dapat dijelaskan seperti berikut :

- Baris 1 : Menyimpan model sebagai mathsymbols.model

Jalankan kode program tersebut, maka menghasilkan seperti pada gambar 7.26.

17. Jelaskan kode program pada blok # In[17]

Berikut adalah kode program yang digunakan :

```
In [16]: model.save("mathsymbols.model")
```

Figure 7.26: Hasil kode program pada blok 16

Listing 7.17: Kode Program 17

```
np.save('classes.npy', label_encoder.classes_)
```

Dari kode listing pada kode program 17, dapat dijelaskan seperti berikut :

- Baris 1 : Menyimpan array dari label\_encoder.classes\_ ke file biner dalam format NumPy .npy.

Jalankan kode program tersebut, maka menghasilkan seperti pada gambar 7.27.

```
In [17]: np.save('classes.npy', label_encoder.classes_)
```

Figure 7.27: Hasil kode program pada blok 17

18. Jelaskan kode program pada blok # In[18]

Berikut adalah kode program yang digunakan :

Listing 7.18: Kode Program 18

```
import keras.models  
model2 = keras.models.load_model("mathsymbols.model")  
print(model2.summary())
```

Dari kode listing pada kode program 18, dapat dijelaskan seperti berikut :

- Baris 1 : Melakukan importing modul models dari library keras
- Baris 2 : Membuat variabel model2 untuk memanggil dan membaca file mathsymbols.model
- Baris 3 : Berfungsi untuk menampilkan dan memeriksa hasil dari variabel parameter model2

Jalankan kode program tersebut, maka menghasilkan seperti pada gambar 7.28.

19. Jelaskan kode program pada blok # In[19]

Berikut adalah kode program yang digunakan :

```

In [18]: import keras.models
...: model2 = keras.models.load_model("mathsymbols.model")
...: print(model2.summary())
Layer (type)          Output Shape         Param #
===== =====
conv2d_11 (Conv2D)    (None, 30, 30, 32)   896
max_pooling2d_11 (MaxPooling) (None, 15, 15, 32)   0
conv2d_12 (Conv2D)    (None, 13, 13, 32)   9248
max_pooling2d_12 (MaxPooling) (None, 6, 6, 32)   0
flatten_10 (Flatten)  (None, 1152)        0
dense_19 (Dense)     (None, 128)         147584
dropout_8 (Dropout)  (None, 128)         0
dense_20 (Dense)     (None, 369)         47601
=====
Total params: 205,329
Trainable params: 205,329
Non-trainable params: 0
=====
None

```

Figure 7.28: Hasil kode program pada blok 18

Listing 7.19: Kode Program 19

```

label_encoder2 = LabelEncoder()
label_encoder2.classes_ = np.load('classes.npy')

def predict(img_path):
    newimg = keras.preprocessing.image.img_to_array(pil_image.open(
    newimg /= 255.0

    # do the prediction
    prediction = model2.predict(newimg.reshape(1, 32, 32, 3))

    # figure out which output neuron had the highest score, and reverse it
    inverted = label_encoder2.inverse_transform([np.argmax(prediction)])
    print("Prediction: %s, confidence: %.2f" % (inverted[0], np.max(prediction)))

```

Dari kode listing pada kode program 19, dapat dijelaskan seperti berikut :

- Baris 1 : Membuat variabel label\_encoder2 yang melakukan pemanggilan fungsi LabelEncoder untuk melakukan penyandian pada label dengan nilai antara 1 dan 0
- Baris 2 : Variabel label\_encoder akan memanggil class yang telah disimpan sebelumnya.
- Baris 3 : Function Predict akan mengubah gambar kedalam bentuk array
- Baris 4 : Variabel prediction akan melakukan prediksi untuk model2 dengan Memberikan bentuk baru tanpa mengubah data dari variabel newimg dengan bentuk array 4D.

- Baris 5 : Variabel inverted akan mencari nilai tertinggi output dari hasil prediksi.

Jalankan kode program tersebut, maka menghasilkan seperti pada gambar 7.29.

```
In [19]: label_encoder2 = LabelEncoder()
...: label_encoder2.classes_ = np.load('classes.npy')
...:
...: def predict(img_path):
...:     newimg = keras.preprocessing.image.img_to_array(pil_image.open(img_path))
...:     newimg /= 255.0
...:
...:     # do the prediction
...:     prediction = model2.predict(newimg.reshape(1, 32, 32, 3))
...:
...:     # figure out which output neuron had the highest score, and reverse the one-hot encoding
...:     inverted = label_encoder2.inverse_transform([np.argmax(prediction)]) # argmax finds highest-scoring output
...:     print("Prediction: %s, confidence: %.2f" % (inverted[0], np.max(prediction)))
```

Figure 7.29: Hasil kode program pada blok 19

20. Jelaskan kode program pada blok # In[20]

Berikut adalah kode program yang digunakan :

Listing 7.20: Kode Program 20

```
predict ("HASYv2/hasy-data/v2-00010.png")
predict ("HASYv2/hasy-data/v2-00500.png")
predict ("HASYv2/hasy-data/v2-00700.png")
```

Dari kode listing pada kode program 20, dapat dijelaskan seperti berikut :

- Baris 1 : Melakukan prediksi terhadap file v2-00010.png yang diambil dari direktori
- Baris 2 : Melakukan prediksi terhadap file v2-00500.png yang diambil dari direktori
- Baris 3 : Melakukan prediksi terhadap file v2-00700.png yang diambil dari direktori

Jalankan kode program tersebut, maka menghasilkan seperti pada gambar 7.30.

```
In [20]: predict("HASYv2/hasy-data/v2-00010.png")
...
...
...
...
Prediction: A, confidence: 0.82
Prediction: \pi, confidence: 0.80
Prediction: \alpha, confidence: 0.88
```

Figure 7.30: Hasil kode program pada blok 20

### 7.1.3 Penanganan Error

Dari praktek pemrograman yang dilakukan di modul ini, error yang kita dapatkan(hasil komputer sendiri) di dokumentasikan dan di selesaikan(nilai 5 per error yang ditan-gani. Untuk hari kedua):

1. skrinsut error

```
In [2]: import random
...: random.shuffle(imgs)
...: split_idx = int(0.8*len(imgs))
...: train = imgs[:split_idx]
...: test = imgs[split_idx:]
Traceback (most recent call last):
File "<ipython-input-2-dd892d606c29>", line 2, in <module>
    random.shuffle(imgs)
NameError: name 'imgs' is not defined
```

Figure 7.31: Screenshot Error

2. Tuliskan kode error dan jenis errornya

- (a) Kode Error :

- Kode error : "Name Error" name 'imgs' is not defined"
- Jenis error : Jenis Error: Images tidak terdefinisi

3. Solusi pemecahan masalah error

Menentukan atau membuka file explorer dari file yang ditempatkan atau dis-esuaikan dengan letak filenya.

# **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 sistem (0,5) c. Lengkap dan bersistem (1)
6	Pemanfaatan Instrumen Pendukung	Pemanfaatan Instrumen Pendukung seperti gambar dan tabel	1	a. Takermanfaatkan (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)
<b>TOTAL</b>			<b>36</b>	
Catatan : Nilai minimal untuk diterima <b>25</b>				

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.