

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

1 Mengenal Kecerdasan Buatan dan Scikit-Learn	1
1.1 Teori	1
1.2 Instalasi	2
1.3 Penanganan Error	2
1.4 Teori/Mhd Zulfikar Akram Nasution/1164081	2
1.5 Jesron Marudut Hatuan/1164077	5
1.5.1 Teori	5
1.5.2 Instalasi	7
1.5.2.1 Instalasi Library Scikit dari Anaconda	7
1.6 Teori/Puad Hamdani/1164084	8
1.7 Instalasi/Mhd Zulfikar Akram Nasution/1164081	13
1.7.1 Installasi	15
1.7.1.1 Loading an Example Datasets	15
1.8 Learning and Predicting	15
1.8.0.1 Model Presistence	16
1.8.0.2 Conventions	17
1.9 Penanganan Error	19
2 Related Works	24
2.1 Mhd Zulfikar Akram Nasution/ 1164081	24
2.1.1 Teori	24
2.2 Mhd Zulfikar Akram Nasution/ 1164081	28
2.3 puad hamdani/1164084	28
2.3.1 binary classification	28
2.3.2 supervised learning dan unsupervised learning	29
2.3.3 evaluasi dan akurasi	29
2.3.4 bagaimana cara membuat dan membaca confusion matrix, buat confusion matrix	30

2.3.5	bagaimana K-fold cross validation bekerja dengan gambar ilustrasi	31
2.3.6	decision tree	31
2.3.7	Information Gain	31
2.4	Puad Hamdani/ 1164084	31
2.4.1	Scikit-Learn	31
2.5	Penanganan Error	43
2.6	Same Topics	44
2.6.1	Topic 1	45
2.6.2	Topic 2	45
2.7	Same Method	45
2.7.1	Method 1	45
2.7.2	Method 2	45
2.8	Teori/Jesron Marudut Hatuan/1164077	45
2.8.1	Binary classification dilengkapi ilustrasi gambar	45
2.8.2	Pengertian Supervised Learning, Unsupervised Learning dan Clustering dan Illustrasi gambar	46
2.8.3	Evaluasi dan akurasi dan Illustrasi gambar	48
2.8.4	Cara membuat dan membaca confusion matrix, buat confusion matrix	48
2.8.5	Membuat cara K-fold cross validation bekerja dengan gambar ilustrasi	49
2.8.6	Decision tree dengan gambar ilustrasi	49
2.8.7	Information Gain dan entropi dengan gambar ilustrasi	50
2.8.8	Scikit-learn	51
2.8.9	Penanganan Eror	56
3	Methods	57
3.1	JESRON MARUDUT HATUAN/1164077	57
3.1.1	Teori	57
3.2	The data	60
3.3	Puad Hamdani/ 1164084	60
3.3.1	Teori	60
3.4	Jesron Marudut Hatuan / 1164077	63
3.4.1	Praktek	63
3.5	Puad Hamdani / 1164084	70

3.5.1	Praktek	70
3.6	Mhd Zulfikar Akram Nasution / 1164081	77
3.6.1	Teori	77
3.6.2	Praktek	79
3.7	Method 1	83
3.8	Method 2	83
4	Klasifikasi Teks	98
4.1	Jesron Marudut Hatuan/1164077	98
4.1.1	Teori	98
4.2	Jesron Marudut Hatuan / 1164077	100
4.2.1	Praktek	100
4.2.2	Penanganan Eror	104
5	Conclusion	107
5.1	Jesron Marudut Hatuan /1164077	107
5.1.1	Teori	107
5.1.2	Praktek Program	110
5.1.3	Penanganan Eror	121
5.2	Jesron Marudut Hatuan/1164077	123
5.2.1	Teori	123
5.2.2	Praktek Program	129
5.2.3	Penanganan Eror	135
5.3	Jesron Marudut Hatuan/1164077	137
5.3.1	Teori	137
5.3.2	Praktek Program	144
5.3.3	Penanganan Eror	165
6	Discussion	167
7	Discussion	168
8	Discussion	169
9	Discussion	170
10	Discussion	171
11	Discussion	172

12 Discussion	173
A Form Penilaian Jurnal	174
B FAQ	177
Bibliography	179

List of Figures

1.1	Install Scikit-Learn Conda	4
1.2	Install Scikit-Learn ke Python	4
1.3	Kompilasi Kode	5
1.4	Import Datasets	5
1.5	Buat variable iris	5
1.6	Buat variable digits	6
1.7	Applikasi Anaconda.	7
1.8	Versi Anaconda.	7
1.9	Instalasi.	8
1.10	Langkah installasi anaconda.	8
1.11	Langkah terakhir.	9
1.12	Proses Instalasi	11
1.13	Gabung Conda dan Python	11
1.14	Kompilasi Kode	12
1.15	Variable Digits	12
1.16	13
1.17	13
1.18	14
1.19	14
1.20	15
1.21	15
1.22	16
1.23	16
1.24	17
1.25	17
1.26	Import file svm	18
1.27	Buat variable Classifier	18
1.28	Lihat array baru dengan syntac Python	19

1.29 Lihat classifier array	19
1.30 Import file	20
1.31 Variable classifier	20
1.32 Variable iris	20
1.33 Penyesuaian Classifier	20
1.34 Import Pickle	20
1.35 Import numpy	20
1.36 Variable rng	20
1.37 Variable X dan hasil random	20
1.38 Variable transformer random	20
1.39 Variable X new type pada transformer	21
1.40 Hasil dari X new type pada transformer	21
1.41 Screenshoot Error	21
1.42 Install Joblib	21
1.43 Solusi Error	21
1.44 Perintah sklearn import datasets	21
1.45 Perintah Variabel Iris	22
1.46 Perintah Variabel Digits	22
1.47 Error Import	22
1.48 Install Library Joblib	22
1.49 Berhasil Import Library Joblib	23
2.1 Binary Classification	24
2.2 Supervised Learning	25
2.3 Unsupervised Learning	26
2.4 Clustering	26
2.5 K-Fold Cross Validation	28
2.6 Decision Tree	29
2.7 Gain dan Entropi	30
2.8 Load Dataset	32
2.9 Load Dataset	32
2.10 Generate Binary Label	33
2.11 Generate Binary Label	33
2.12 Pemanggilan get dummies dari lontong	34
2.13 Pemanggilan get dummies	34
2.14 Mendefinisikan pembagian data	36

2.15 Mendefinisikan pembagian data	36
2.16 Membuktikan pengujian	37
2.17 Membuktikan pengujian	37
2.18 Gambaran decision tree	38
2.19 Library Graphviz	39
2.20 Menampilkan hasil perhitungan 2 parameter	39
2.21 Library Graphviz	39
2.22 Menampilkan hasil perhitungan 2 parameter	40
2.23 Mendefinisikan library sklearn	40
2.24 Mendefinisikan library sklearn	40
2.25 Menampilkan hasil fungsi max depth dan accuracy	41
2.26 Menampilkan hasil fungsi max depth dan accuracy	41
2.27 Menjelaskan variable kari	42
2.28 Menjelaskan variable kari	43
2.29 Menjelaskan dan menampilkan gambar grafik	43
2.30 Menjelaskan dan menampilkan gambar grafik	44
2.31 ScreenShoot Error	44
2.32 Penanganan Error	45
2.33 Klasifikasi Binari	46
2.34 Supervised Learning	47
2.35 Unsupervised Learning	47
2.36 Cluster	48
2.37 Evaluasi dan Akurasi	48
2.38 K-fold cross validation	50
2.39 Decision Tree	50
2.40 Information gain	51
2.41 Entropi	51
2.42 Load Dataset	51
2.43 Generate Binary label	52
2.44 Use one-hot encoding	52
2.45 Shuffle rows	53
2.46 Fit a Decision tree	53
2.47 Visualize tree	53
2.48 Save tree	54
2.49 Score	54
2.50 Show Average score	54

2.51 Max depth	55
2.52 Depth Acc	55
2.53 Import	56
3.1 Gambar Random Forest	58
3.2 Gambar Code Python	58
3.3 Gambar Output	59
3.4 Gambar Import Dataset	59
3.5 Gambar Hasil Dataset Sel	60
3.6 Gambar Masukkan Perintah	60
3.7 Pohon Keputusan	61
3.8 Gambar Data Testing	61
3.9 Gambar Voting Random Forest	62
3.10 Random Forest	62
3.11 Applikasi sederhana Pandas	63
3.12 Applikasi Numpy	64
3.13 Applikasi Matplotlib	64
3.14 Gambar ke-1	64
3.15 Gambar ke-2	65
3.16 Gambar ke-3	65
3.17 Gambar ke-4	65
3.18 Gambar ke-5	66
3.19 Gambar ke-6	66
3.20 Gambar ke-7	67
3.21 Gambar ke-8	67
3.22 Gambar ke-9	68
3.23 Gambar ke-10	68
3.24 Gambar ke-11	69
3.25 Gambar ke-12	69
3.26 Gambar ke-13	70
3.27 Gambar ke-14	70
3.28 Gambar ke-15	71
3.29 Gambar ke-16	71
3.30 Gambar ke-17	72
3.31 Gambar ke-18	72
3.32 Gambar ke-19	73

3.33 Gambar ke-20	73
3.34 Gambar ke-21	74
3.35 Gambar ke-22	75
3.36 Gambar ke-23	75
3.37 Gambar SVM	76
3.38 Decission Tree	76
3.39 Cross Validation 1	77
3.40 Cross Validation 2	77
3.41 Cross Validation 3	78
3.42 Program Pengamatan Komponen Informasi 1	79
3.43 Program Pengamatan Komponen Informasi 2	80
3.44 Error	81
3.45 Gambar1	81
3.46 Gambar2	82
3.47 Gambar3	82
3.48 Gambar 4	83
3.49 Gambar 5	83
3.50 Gambar 6	84
3.51 Gambar 7	84
3.52 Gambar 8	84
3.53 Gambar 9	84
3.54 Gambar 10	84
3.55 Gambar 11	84
3.56 Gambar 12	84
3.57 Gambar 13	85
3.58 Gambar 14	85
3.59 Gambar 15	85
3.60 Gambar 16	85
3.61 Gambar 17	85
3.62 Gambar 18	85
3.63 Gambar 19	85
3.64 Gambar 20	86
3.65 Gambar 21	86
3.66 Gambar 22	86
3.67 Gambar 23	87
3.68 SVM	87

3.69 Decission Tree	87
3.70 Cross Validation 1	87
3.71 Cross Validation 2	87
3.72 Cross Validation 3	88
3.73 Program Pengamatan Komponen Informasi 1	88
3.74 Program Pengamatan Komponen Informasi 2	89
3.75 Error	89
3.76 Random Forest	90
3.77 Confusion Matrix	90
3.78 Voting	90
3.79 Aplikasi pandas	90
3.80 Hasil Pandas	91
3.81 Aplikasi Numpy	91
3.82 Hasil Numpy	91
3.83 Aplikasi Matplotlib	91
3.84 Hasil Matplotlib	91
3.85 Membaca Data File	92
3.86 Melihat sebagian data	92
3.87 Melihat jumlah data	92
3.88 Ubah atribut jadi kolom	92
3.89 Membaca dataset label	92
3.90 Menggabungkan field	92
3.91 Memisahkan dan memilih label	92
3.92 Melihat isi data	93
3.93 Pembagian data	93
3.94 Kelas random forest	93
3.95 Membuat fitting	93
3.96 Melihat hasil	93
3.97 Hasil score	93
3.98 Memetakan ke confusion matrix	93
3.99 Melihat hasil	94
3.100Melakukan Plot	94
3.101Plotting nama data	94
3.102Melakukan perintah plot	95
3.103Klasifikasi menggunakan decision tree	95
3.104Klasifikasi menggunakan SVM	95

3.105	Pengecekan cross validation random forest	95
3.106	Pengecekan cross validation decision tree	95
3.107	Pengecekan cross validation SVM	96
3.108	Pengamatan Komponen	96
3.109	Plot informasi	97
4.1	Klasifikasi teks	98
4.2	Klasifikasi bunga	99
4.3	Bag of Word	100
4.4	TF-IDF	101
4.5	Data Dummy 500 Data	101
4.6	Membagi 2 Dataframe	102
4.7	Vektorisasi dan Klasifikasi Data	102
4.8	Data Content	102
4.9	DataFrame Kata-kata Pada Content	103
4.10	Klasifikasi SVM Dari Data Vektorisasi	103
4.11	Klasifikasi Decision Tree Dari Data Vektorisasi	103
4.12	Plot Confusion Matrix Menggunakan Matplotlib	104
4.13	Program Cross Validation Pada Data Vektorisasi	104
4.14	Program Pengamatan Komponen Informasi	105
4.15	Eror matplotlib.pyplot	105
5.1	Ilustrasi Soal No. 1	107
5.2	Ilustrasi Soal No. 2	108
5.3	Ilustrasi Soal No. 3	109
5.4	Ilustrasi Soal No. 4	109
5.5	Ilustrasi Soal No. 5	110
5.6	Ilustrasi Soal No. 6	110
5.7	Love	111
5.8	Faith	111
5.9	Fall	111
5.10	Sick	112
5.11	Clear	112
5.12	Shine	112
5.13	Bag	112
5.14	Car	113
5.15	Wash	113

5.16 Motor	113
5.17 Cycle	114
5.18 Similariti Pada Kata Motor dan Cycle	114
5.19 Similariti Pada Kata Wash dan Motor	114
5.20 Extract_Words	115
5.21 Permute Sentences	115
5.22 Gensim TaggedDocument dan Doc2vec	116
5.23 Aclimbdb	117
5.24 Aclimbdb	117
5.25 Aclimbdb	118
5.26 Infer Code	119
5.27 Cosine Similarity	120
5.28 Cross Validation Metode 1	120
5.29 Cross Validation Metode 2	121
5.30 Cross Validation Metode 3	121
5.31 eror	121
5.32 Suara ke MFCC	123
5.33 Neural Network	124
5.34 Konsep Pembobotan pada Neural Network	125
5.35 Fungsi Aktivasi pada Neural Network	126
5.36 Membaca Hasil Plot dari MFCC	127
5.37 One-Hot Encoding	127
5.38 np.unique	128
5.39 to_categorical	128
5.40 One-Hot Encoding	129
5.41 GTZAN	130
5.42 Display MFCC	130
5.43 Generate Features and Labels	132
5.44 Training Data Sebesar 80%	132
5.45 Fungsi Compile	133
5.46 Fungsi Fit	134
5.47 Fungsi Evaluate	135
5.48 Fungsi Predict	135
5.49 Eror	136
5.50 Tokenizer	137
5.51 StartifiedKFold	138

5.52 Fungsi Tokenizer	139
5.53 Matrix TFID	140
5.54 Matrix Training dan Testing	141
5.55 Konvolusi	143
5.56 Algoritma Perhitungan Konvolusi	143
5.57 In[1]	144
5.58 In[2]	146
5.59 In[3]	147
5.60 In[4]	148
5.61 In[5]	149
5.62 In[6]	149
5.63 In[7]	150
5.64 In[8]	151
5.65 In[9]	152
5.66 In[10]	154
5.67 In[11]	154
5.68 In[12]	156
5.69 In[13]	158
5.70 In[14]	160
5.71 In[15]	161
5.72 In[16]	162
5.73 In[17]	162
5.74 In[18]	163
5.75 In[19]	165
5.76 In[20]	165
5.77 Eror	166
A.1 Form nilai bagian 1.	175
A.2 form nilai bagian 2.	176

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)

iiiiii HEAD

1.4 Teori/Mhd Zulfikar Akram Nasution/1164081

1. Definisi, Sejarah dan Perkembangan Kecerdasan Buatan

- Definisi

Kecerdasan Buatan adalah kecerdasan yang ditambahkan kepada suatu sistem yang bisa diatur dalam konteks ilmiah yang berhubungan dengan pemanfaatan mesin untuk memecahkan persoalan yang rumit dengan cara yang lebih manusiawi.

- Sejarah dan Perkembangan

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

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

- Supervised Learning dan Unsupervised Learning

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

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

- Regresi

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

- Data Set

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

- Training Set

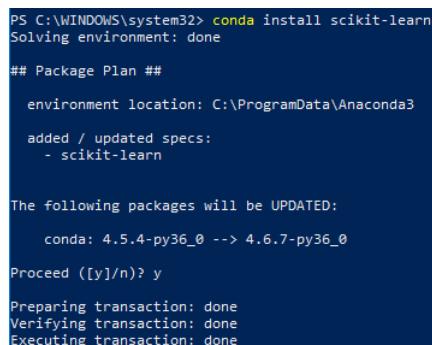
Training set yaitu jika pasangan objek, dan kelas yang menunjuk pada objek tersebut adalah suatu contoh yang telah diberi label akan menghasilkan suatu algoritma pembelajaran.

- Testing Set

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

3. Instalasi Scikit-Learn dari Anaconda

- Pertama install Anaconda di pc masing-masing
- Kemudian buka cmd untuk menginstall scikit-learn
- Ketik perintah ”conda install scikit-learn” dan pilih ”y”



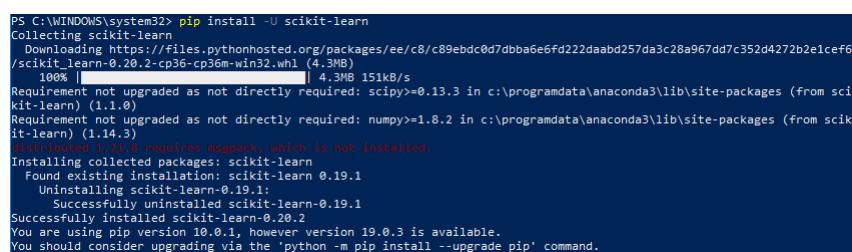
```
PS C:\WINDOWS\system32> conda install scikit-learn
Solving environment: done
## Package Plan ##
environment location: C:\ProgramData\Anaconda3
added / updated specs:
- scikit-learn

The following packages will be UPDATED:
  conda: 4.5.4-py36_0 --> 4.6.7-py36_0

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

Figure 1.1: Install Scikit-Learn Conda

- Lalu ketik ”pip install -U scikit-learn” untuk memasukkan anaconda ke python



```
PS C:\WINDOWS\system32> 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 15kB/s
Requirement not upgraded as not directly required: scipy>=0.13.3 in c:\programdata\anaconda3\lib\site-packages (from scikit-learn) (1.1.0)
Requirement not upgraded as not directly required: numpy>=1.8.2 in c:\programdata\anaconda3\lib\site-packages (from scikit-learn) (1.14.3)
skipped because it is a purewheel which is not installed.
Installing collected packages: scikit-learn
  Found existing installation: scikit-learn 0.19.1
    Uninstalling scikit-learn-0.19.1:
      Successfully uninstalled scikit-learn-0.19.1
      Successfully installed scikit-learn-0.20.2
You are using pip version 10.0.1, however version 19.0.3 is available.
You should consider upgrading via the 'python -m pip install --upgrade pip' command.
```

Figure 1.2: Install Scikit-Learn ke Python

- Setelah itu, kompilasi kode di dalam python dengan ketik ”python”, lalu ”print(‘Zulfikar’)” maka akan menghasilkan seperti gambar berikut.

```
PS C:\WINDOWS\system32> python
Python 3.6.5 |Anaconda, Inc.| (default, Mar 29 2018, 13:23:52) [MSC v.1900 32 bit (Intel)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> print('Zulfikar')
Zulfikar
```

Figure 1.3: Kompilasi Kode

4. Loading an Example Dataset

- Ketik perintah berikut ”from sklearn import datasets” untuk mengimport dataset dari sklearn.

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

Figure 1.4: Import Datasets

- Kemudian ketik perintah berikut untuk membuat variable iris yang berisi datasets.

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

Figure 1.5: Buat variable iris

- Lalu ketik perintah berikut untuk membuat variable digits yang berisi datasets, dan juga untuk melihat isi data dari datasets seperti gambar 1.6

.

1.5 Jesron Marudut Hatuan/1164077

1.5.1 Teori

1. Definisi, sejarah, dan perkembangan kecerdasan buatan.

Kecerdasan Buatan (Artificial Intelligence atau AI) dapat didefinisikan sebagai kecerdasan yang ditunjukkan oleh suatu entitas buatan. Sistem seperti ini biasanya dianggap komputer. Kecerdasan diciptakan lalu dimasukkan ke dalam suatu mesin atau komputer supaya dapat melakukan pekerjaan-pekerjaan yang dapat dilakukan manusia.

Sebenarnya area Kecerdasan Buatan (Artificial Intelligence) atau disingkat dengan AI, dimulai dari munculanya komputer sekitar tahun 1940-an, meskipun

```

>>> digits = datasets.load_digits()
>>> print(digits.data)
[[ 0.  0.  5. ... 0.  0.  0.]
 [ 0.  0.  0. ... 10. 0.  0.]
 [ 0.  0.  0. ... 16. 9.  0.]
 ...
 [ 0.  0.  1. ... 6.  0.  0.]
 [ 0.  0.  2. ... 12. 0.  0.]
 [ 0.  0.  10. ... 12. 1.  0.]]
```

Figure 1.6: Buat variable digits

sejarah perkembangannya dapat dilacak dari zaman Mesir kuno. Pada akhir tahun 1955, Newell dan Simon mengembangkan The Logic Theorist atau program AI terdahulu. Program ini merepresentasikan masalah sebagai model pohon, lalu penyelesaiannya dengan memilih cabang yang akan menghasilkan kesimpulan terbenar. Program tersebut berdampak besar dan menjadi batu loncatan dalam mengembangkan bidang AI. Pada tahun 1956 John McCarthy dari Massachusetts Institute of Technology dianggap sebagai bapak AI, menyelenggarakan konferensi untuk menarik para ahli komputer bertemu, dengan nama kegiatan The Dartmouth Summer Research Project On AI. Konferensi Dartmouth saat itu mempertemukan para pendiri dalam AI, dan bertugas untuk meletakkan dasar bagi masa depan pengembangan dan penelitian AI. John McCarthy disaat itu mengusulkan definisi AI adalah AI merupakan cabang dari ilmu komputer yang berfokus pada pengembangan komputer agar mempunyai kemampuan dan berprilaku seperti manusia.

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

Supervised learning merupakan sebuah pendekatan dimana sudah terdapat data yang dilatih, dan terdapat variable yang ditargetkan sehingga tujuan dari pendekatan ini adalah 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.

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

ketika diberi foto seseorang, kita ingin memprediksi umur, tinggi, dan berat orang yang ada pada foto tersebut.

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

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

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

1.5.2 Instalasi

1.5.2.1 Instalasi Library Scikit dari Anaconda

1. Sediakan aplikasi Anaconda terlebih dahulu



Figure 1.7: Applikasi Anaconda.

2. Setelah di install, masukkan script dibawah ini untuk melihat versi Python dan Anacondanya

```
C:\WINDOWS\system32>conda --version  
conda 4.6.7  
  
C:\WINDOWS\system32>python --version  
Python 3.6.5 :: Anaconda, Inc.  
  
C:\WINDOWS\system32>
```

Figure 1.8: Versi Anaconda.

3. Selanjutnya masukkan perintah 'pip install -U scikit-learn'
4. Selanjutnya masukkan perintah 'conda install scikit-learn'

```
C:\WINDOWS\system32>pip install -U scikit-learn
Collecting scikit-learn
  Using cached https://files.pythonhosted.org/packages/ee/c8/c89ebdc0d7dbba6e6fd222daabd257da3
c28a967d7c352d4272b2e1cef6/scikit_learn-0.20.2-cp36-cp36m-win32.whl
Requirement not upgraded as not directly required: numpy>=1.8.2 in c:\programdata\anaconda3\lib\site-packages (from scikit-learn) (1.14.3)
Requirement not upgraded as not directly required: scipy>=0.13.3 in c:\programdata\anaconda3\lib\site-packages (from scikit-learn) (1.1.0)
distributed 1.21.3 requires msgpack, which is not installed.
Installing collected packages: scikit-learn
  Found existing installation: scikit-learn 0.19.1
    Uninstalling scikit-learn-0.19.1:
      Successfully uninstalled scikit-learn-0.19.1
Successfully installed scikit-learn-0.20.2
You are using pip version 10.0.1, however version 19.0.3 is available.
You should consider upgrading via the 'python -m pip install --upgrade pip' command.
```

Figure 1.9: Instalasi.

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

## Package Plan ##

environment location: C:\ProgramData\Anaconda3

added / updated specs:
- scikit-learn

The following packages will be UPDATED:

  conda: 4.5.4-py36_0 --> 4.6.7-py36_0

Proceed ([y]/n)? y

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

C:\WINDOWS\system32>
```

Figure 1.10: Langkah instalasi anaconda.

5. Selanjutnya masukkan perintah 'python' dan 'print ('jesron')

```
||||| HEAD
=====
```

1.6 Teori/Puad Hamdani/1164084

1. Definisi, Sejarah dan Perkembangan Kecerdasan Buatan

- Definisi

Kecerdasan buatan adalah ilmu pengetahuan yang berhubungan dengan mesin untuk memecahkan persoalan rumit dengan cara yang mudah,dilakukan

```
C:\WINDOWS\system32>python
Python 3.6.5 |Anaconda, Inc.| (default, Mar 29 2018, 13:23:52) [MSC v.1900 32 bit (Intel)] on
win32
Type "help", "copyright", "credits" or "license" for more information.
>>> print('jesron')
jesron
>>>
```

Figure 1.11: Langkah terakhir.

dengan mengikuti kecerdasan manusia dan menerapkannya di computer sebagai algoritma

- Sejarah dan Perkembangan

AI (artificial Intelligence) di kenal sekitar tahun 1943 Teori tentang jaringan saraf tiruan (artificial neuron network, ANN) menyatakan bahwa setiap neuron dapat dimisalkan dalam keadaan biner, yaitu ON dan OFF. Dari setiap percobaan, setiap fungsi perhitungan dapat diselesaikan melalui jaringan neuron yang dimodelkan. Pada tahun 1965, Lotfi Zadeh, professor teknik elektro di University of California, memublikasikan konsepnya yang disebut dengan “fuzzy sets”. Beliau menjabarkan FL dengan pernyataan matematis dan visual yang mudah dipahami. Karena kajian ini berkaitan dengan sistem kontrol, konsep tersebut banyak dikembangkan dalam konteks pemrograman komputer hingga saat ini.

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

- Supervised Learning

Supervised learning adalah pembelajaran yang terawasi dimana jika output yang diharapkan telah diketahui sebelumnya. Biasanya pembelajaran ini dilakukan dengan menggunakan data yang telah ada

- Unsupervised Learning

Unsupervised learning adalah pembelajaran yang tidak terawasi dimana tidak memerlukan target output. Metode ini tidak dapat ditentukan hasil seperti apa yang diharapkan selama proses pembelajaran, Nilai bobot yang disusun dalam proses range tertentu tergantung pada output yang diberikan.

- Klasifikasi

Klasifikasi adalah Proses pengelompokan berdasarkan ciri-ciri persamaan dan perbedaan

- Regresi

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

- Data Set

Data set adalah objek yang merepresentasikan data dan relasinya di memory, Strukturnya hampir mirip dengan data di data base. Data set berisi koleksi dari data table dan data relation

- Training Set

Training set adalah bagian dataset yang kita latih untuk membuat prediksi atau algoritma ML lainnya sesuai tujuannya masing-masing. Kita memberikan petunjuk melalui algoritma agar mesin yang kita latih bisa mencari korelasinya sendiri. Walau demikian proses belajar harusnya proporsional. Layaknya seorang murid yang terlalu diforsir belajar, maka hasilnya pun tidak akan baik. Dalam istilah ML disebut dengan overfitting. Akan lebih mudah memahami konsep overfitting melalui praktik.

- Testing Set

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

3. Instalasi Scikit-Learn dari Anaconda

- Pertama install Anaconda di pc masing-masing
- Kemudian buka cmd untuk menginstall scikit-learn
- Ketikan ”conda install scikit-learn” dan pilih ”y”
- ketik ”pip install -U scikit-learn” untuk menggabungkan anaconda dan python
- Setelah itu, kompilasi kode di dalam python dengan ketik ”python”, lalu ”print('puad')” maka akan menghasilkan seperti gambar berikut.

4. Loading an Example Dataset

- Ketik perintah berikut ”from sklearn import datasets” untuk mengimport dataset dari sklearn.

```

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

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

## Package Plan ##

environment location: C:\ProgramData\Anaconda3

added / updated specs:
- scikit-learn

The following packages will be UPDATED:

  conda: 4.5.4-py36_0 --> 4.6.7-py36_0

Proceed ([y]/n)? y

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

C:\WINDOWS\system32>

```

Figure 1.12: Proses Instalasi

```

C:\WINDOWS\system32>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 (4.3M)
    100% |██████████| 4.3MB 1.2MB/s
Requirement not upgraded as not directly required: scipy>=0.13.3 in c:\programdata\anaconda3\lib\site-packages (from scikit-learn) (1.1.0)
Requirement not upgraded as not directly required: numpy>=1.8.2 in c:\programdata\anaconda3\lib\site-packages (from scikit-learn) (1.14.3)
distributed 1.21.8 requires msgpack, which is not installed.
Installing collected packages: scikit-learn
  Found existing installation: scikit-learn 0.19.1
    Uninstalling scikit-learn-0.19.1:
      Successfully uninstalled scikit-learn-0.19.1
Successfully installed scikit-learn-0.20.2
You are using pip version 10.0.1, however version 19.0.3 is available.
You should consider upgrading via the 'python -m pip install --upgrade pip' command.

C:\WINDOWS\system32>

```

Figure 1.13: Gabung Conda dan Python

- ketik perintah ”iris = datasets.load iris” untuk membuat variable iris yang berisi datasets.
- ketik perintah berikut”digits = datasets.load digits”untuk membuat variabel digits yang berisi datasets, dan juga ”print(digits.data)” untuk melihat isi data dari datasets seperti gambar
- kemudian ketik ”digits target”
- kemudian ketik ”digits.images[0] ”
- kemudian ketik ”from sklearn import svm” dan kemudian clf = svm.SVC(gamma=0.001, C=100.) ”
- kemudian ketik ”clf.fit(digits.data[:-1], digits.target[:-1]) ”

```
C:\WINDOWS\system32>python
Python 3.6.5 |Anaconda, Inc.| (default, Mar 29 2018, 13:23:52) [MSC v.1900 32 bit (Intel)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> print('puad')
puad
>>>
```

Figure 1.14: Kompilasi Kode

```
>>> from sklearn import datasets
>>> iris = datasets.load_iris()
>>> digits = datasets.load_digits()
>>> print(digits.data)
[[ 0.  0.  5. ... 0.  0.  0.]
 [ 0.  0.  0. ... 10.  0.  0.]
 [ 0.  0.  0. ... 16.  9.  0.]
 ...
 [ 0.  0.  1. ... 6.  0.  0.]
 [ 0.  0.  2. ... 12.  0.  0.]
 [ 0.  0.  10. ... 12.  1.  0.]]
```

Figure 1.15: Variable Digits

- kemudian ketik ”clf.predict(digits.data[-1:])”
- kemudian ketik ”from sklearn import svm”
- kemudian ketik ”from sklearn import datasets”
- kemudian ketik ”clf = svm.SVC(gamma='scale')”
- kemudian ketik ”iris = datasets.load(andeskore)iris()”
- kemudian ketik ”X, y = iris.data, iris.target”
- kemudian ketik ”clf.fit(X, y) ”
- kemudian ketik ”import pickle”
- kemudian ketik ”s = pickle.dumps(clf)”
- kemudian ketik ”clf2 = pickle.loads(s)”
- kemudian ketik ”clf2.predict(X[0:1])”
- kemudian ketik ” y[0]”
- kemudian ketik ”from joblib import dump, load”
- kemudian ketik ”dump(clf, 'filename.joblib')”
- conventions

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

Figure 1.16:

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

Figure 1.17:

- ketikan "import numpy as np"
- ketikan "from sklearn import random(andeskor)projection"
- ketikan "rng = np.random.RandomState(0)"
- ketikan "X = rng.rand(10, 2000)"
- ketikan "X = np.array(X, dtype='float32')"
- ketikan "X.dtype"
- ketikan "transformer = random projection.GaussianRandomProjection"
- ketikan "X new = transformer.fit transform(X)"
- ketikan "X new.dtype"

5. screenshoot eror

6. kode eror "no module named 'joblib'"

7. penanganannya instal joblib dengan mengetikan"conda instal -c anaconda joblib"

lliiliii efee6eca5df5ee495d6544dd76ec774eee595c8a ===== jjjjjj HEAD

1.7 Instalasi/Mhd Zulfikar Akram Nasution/1164081

1. Menjelaskan Kode dari Learning and Predicting

- Pertama import file smv dari sklearn seperti pada gambar 1.12
- Kemudian buat variabel clf seperti pada gambar 1.13
- Lalu ketik kode berikut untuk meliat array baru dari syntax python [-1] seperti pada gambar 1.14

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

Figure 1.18:

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

Figure 1.19:

- Selanjutnya ketikkan kode berikut untuk melihat penggolongan array seperti pada gambar 1.15

2. Model Persistence

- Pertama Import dulu file dari sklearn
- Kemudian buat variable classifier dengan gamma=scale
- Lalu buat variable iris dan (X,y)
- Selanjutnya kita akan melihat penyesuaian classifier
- Kemudian import pickle untuk melihat hasil array dan hasil y

3. Conventions

- Pertama import numpy menjadi np serta import random projection
- Kemudian buat variable rng dengan type random
- Lalu buat variable X, dan lihat hasil rng random yang keluar
- Setelah itu buat variable transformer dengan type random
- Berikutnya itu buat variable X new dengan type yang ada pada transformer
- Kemudian lihat hasil dari X new

```

>>> from sklearn import svm
>>> from sklearn import datasets
>>> clf = svm.SVC(gamma='scale')
>>> iris = datasets.load_iris()
>>> X,y =iris.data, iris.target
>>> clf.fit(X, y)
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 1.20:

```

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

```

Figure 1.21:

4. Screenshot Error pada gambar 1.27
 5. Kode yang error yaitu ”joblib” karena belum ada library nya seperti pada gambar 1.28
 6. Solusi dari masalah yang error seperti pada gambar 1.29
- =====

1.7.1 Installasi

1.7.1.1 Loading an Example Datasets

1. Loading an Example Dataset

- Ketik perintah berikut ”from sklearn import datasets” untuk mengimport dataset dari file sklearn tadi.
- Selanjutnya ketik perintah berikut ini untuk membuat variable iris yang berisi datasets.
- Masukkan perintah ini untuk membuat variable digits yang berisi datasets, dapat juga untuk melihat isi data dari datasets tadi.

1.8 Learning and Predicting

- from sklearn import svm (pada baris berikut ini merupakan sebuah perintah untuk mengimport class svm dari package sklearn).

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

Figure 1.22:

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

Figure 1.23:

- clf = svm.SVC (gamma=0.001, C=100.) (pada baris kedua ini clf sebagai estimator atau parameter, svm.SVC menjadi sebuah class, dan gamma sebagai parameter untuk menetapkan nilai secara manual)
- clf.fit(digits.data[:-1], digits.target[:-1]) (pada baris ketiga ini clf sebagai estimator atau parameter, fit sebagai metode, digits.data sebagai item, [-1] sebagai syntax pythonnya dan menampilkan outputannya)
- clf.predict(digits.data[-1:])

1.8.0.1 Model Persistence

- from sklearn import svm
- from sklearn import datasets
- clf = svm.SVC(gamma='scale')
- iris = datasets.load_iris()
- X, y = iris.data, iris.target
- clf.fit(X, y)hasil
- import pickle
- s = pickle.dumps(clf)
- clf2 = pickle.loads(s)
- clf2.predict(X[0:1])hasil
- y[0]hasil
- from joblib import dump, load eror
- dump(clf, 'filename.joblib')eror
- clf = load('filename.joblib')eror

```

>>> import numpy as np
>>> from sklearn import random_projection
>>> rng = np.random.RandomState(0)
>>> X = rng.rand(10, 2000)
>>> X = np.array(X, dtype='float32')
>>> X.dtype
dtype('float32')
>>> transformer = random_projection.GaussianRandomProjection()
>>> X_new = transformer.fit_transform(X)
>>> X_new.dtype
dtype('float64')
>>>

```

Figure 1.24:

```

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

```

Figure 1.25:

1.8.0.2 Conventions

1. Type Casting

- from sklearn import svm
- from sklearn import random_projection
- rng = np.random.RandomState(0)
- X = rng.rand(10, 2000)
- X = np.array(X, dtype='float32')
- X.dtype hasil
- transformer = random_projection.GaussianRandomProjection()
- X_new = transformer.fit_transform(X)
- X_new.dtype hasil
- from sklearn import datasets
- from sklearn.svm import SVC
- iris = datasets.load_iris()
- clf = SVC(gamma='scale')
- clf.fit(iris.data, iris.target)hasil
- list(clf.predict(iris.data[:3])) hasil
- clf.fit(iris.data, iris.target_names[iris.target]) hasil
- list(clf.predict(iris.data[:3])) hasil

2. Refitting and Updating Parameters

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

Figure 1.26: Import file svm

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

Figure 1.27: Buat variable Classifier

- 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) hasil
- clf.predict(X_test) hasil
- clf.set_params(kernel='rbf', gamma='scale').fit(X, y) hasil
- clf.predict(X_test) hasil

3. Multiclass vs. Multilabel Fitting

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

```
>>> 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 1.28: Lihat array baru dengan syntac Python

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

Figure 1.29: Lihat classifier array

1.9 Penanganan Error

1. Dibawah ini merupakan error yang ditemukan pada saat melakukan percobaan import.
2. Pada gambar diatas, terjadi error ketika sedang mengimport modul yang telah ditetapkan.
3. Solusinya dapat dilakukan dengan berikut ini :
Error tadi terjadi akibat Library Joblib pada PC belum terinstall. Oleh sebab itu, install terlebih dahulu.
4. Dengan membuka CMD (Admin), kemudian masukkan perintah ”pip install joblib” dan tunggu sampai installasi berhasil seperti gambar berikut.
5. Ketika sudah terinstall, maka bisa dilakukan lagi import library joblib, dan hasilnya akan tampil seperti dibawah ini

```
iiiiiii iiii f594df5d4cf2e0a7297e57a9c544d6f3fac837ab iiii 2951ffb45da2a049bd0331ec
```

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

Figure 1.30: Import file

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

Figure 1.31: Variable classifier

```
>>> iris = datasets.load_iris()  
>>> X, y = iris.data, iris.target
```

Figure 1.32: Variable iris

```
>>> clf.fit(X,y)  
SVC(C=1.0, cache_size=200, class_weight=None, coef0=0  
.0,  
     decision_function_shape='ovr', degree=3, gamma='sca  
le', kernel='rbf',  
     max_iter=-1, probability=False, random_state=None,  
     shrinking=True,  
     tol=0.001, verbose=False)
```

Figure 1.33: Penyesuaian Classifier

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

Figure 1.34: Import Pickle

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

Figure 1.35: Import numpy

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

Figure 1.36: Variable rng

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

Figure 1.37: Variable X dan hasil random

```
>>> transformer = random_projection.GaussianRandomPro  
jection()
```

Figure 1.38: Variable transformer random

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

Figure 1.39: Variable X new type pada transformer

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

Figure 1.40: Hasil dari X new type pada transformer

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

Figure 1.41: Screenshot Error

```
C:\Users\user>conda install -c anaconda joblib
Collecting package metadata: done
Solving environment: done

## Package Plan ##

environment location: C:\ProgramData\Anaconda3

added / updated specs:
  - joblib

The following packages will be downloaded:

          package                    build
ca-certificates-2018.03.07          0      155 KB  anaconda
certifi-2018.4.16                  py36_0    143 KB  anaconda
conda-4.4.12                        0      1.9 MB  anaconda
joblib-0.13.2                       py36_0    361 KB  anaconda
openssl-1.0.2                      h8ea7d7f_0   4.2 MB  anaconda
qt-5.9.5                            vc74hna4qad_0   9.2 MB  anaconda

Total:                           96.7 MB

The following NEW packages will be INSTALLED:
joblib                         anaconda/win-32::joblib-0.13.2-py36_0

The following packages will be SUPERSEDED by a higher-priority channel:
ca-certificates
certifi
conda
openssl
qt

Proceed ((y/n))? y

Downloading and Extracting Packages
joblib-0.13.2          |████████| 361 KB  100%
certifi-2018.4.16      |████████| 155 KB  100%
ca-certificates-2018  |████████| 155 KB  100%
qt-5.9.5               |████████| 90.2 MB 100%
```

Figure 1.42: Install Joblib

```
>>> from sklearn import svm  
>>> clf = svm.SVC(gamma='scale')  
>>> from joblib import dump, load  
>>> dump(clf, 'filename.joblib')  
['filename.joblib']
```

Figure 1.43: Solusi Error

```
C:\WINDOWS\system32>python
Python 3.6.5 |Anaconda, Inc.| (default, Mar 29 2018, 13:23:52) [MSC v.1
900 32 bit (Intel)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> from sklearn import datasets
```

Figure 1.44: Perintah sklearn import datasets

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

Figure 1.45: Perintah Variabel Iris

```
>>> digits = datasets.load_digits()
>>> print(digits.data)
[[ 0.  0.  5. ...  0.  0.  0.]
 [ 0.  0.  0. ... 10.  0.  0.]
 [ 0.  0.  0. ... 16.  9.  0.]
 ...
 [ 0.  0.  1. ...  6.  0.  0.]
 [ 0.  0.  2. ... 12.  0.  0.]
 [ 0.  0. 10. ... 12.  1.  0.]]
>>>
```

Figure 1.46: Perintah Variabel Digits

```
C:\WINDOWS\system32>python
Python 3.6.5 |Anaconda, Inc.| (default, Mar 29 2018, 13:23:52) [MSC v.1900 32 bit (Intel)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> from joblib import dump, load
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
ModuleNotFoundError: No module named 'joblib'
```

Figure 1.47: Error Import

```
C:\WINDOWS\system32>pip install joblib
Collecting joblib
  Downloading https://files.pythonhosted.org/packages/cd/c1/50a753e8247561e58cb87305b1e90b171b8c767b15b12a1734001f41d356/joblib-0.13.2-py3-none-any.whl (278 kB)
    11% [███████] | 30kB 453kB/s eta 0:00:01 14%
    22% [██████████] | 40kB 593kB/s eta 0:00:0 18%
    33% [██████████] | 51kB 721kB/s eta 0:00:0 22%
    44% [██████████] | 61kB 842kB/s eta 0:00:0 25%
    55% [██████████] | 71kB 969kB/s eta 0:00:0 29%
    66% [██████████] | 81kB 1.1MB/s eta 0:00:0 33%
    77% [██████████] | 92kB 1.2MB/s eta 0:00:0 36%
    88% [██████████] | 102kB 1.3MB/s eta 0:00:0 40%
    99% [██████████] | 112kB 1.4MB/s eta 0:00:0 44%
   100% [██████████] | 122kB 1.5MB/s eta 0:00:0 47%
    0% [██████████] | 133kB 1.6MB/s eta 0:00:0 51%
    22% [██████████] | 143kB 1.7MB/s eta 0:00:0 55%
    44% [██████████] | 153kB 1.8MB/s eta 0:00:0 58%
    66% [██████████] | 163kB 1.9MB/s eta 0:00:0 62%
    88% [██████████] | 174kB 2.0MB/s eta 0:00:0 65%
    100% [██████████] | 184kB 2.1MB/s eta 0:00:0 69%
    0% [██████████] | 194kB 2.2MB/s eta 0:00:0 73%
    22% [██████████] | 204kB 2.3MB/s eta 0:00:0 77%
    44% [██████████] | 215kB 2.4MB/s eta 0:00:0 81%
    66% [██████████] | 225kB 2.5MB/s eta 0:00:0 85%
    88% [██████████] | 235kB 2.6MB/s eta 0:00:0 89%
    100% [██████████] | 245kB 2.7MB/s eta 0:00:0 93%
    0% [██████████] | 255kB 2.8MB/s eta 0:00:0 97%
    22% [██████████] | 265kB 2.9MB/s eta 0:00:0 101%
    44% [██████████] | 275kB 3.0MB/s eta 0:00:0 105%
    66% [██████████] | 285kB 3.1MB/s eta 0:00:0 109%
    88% [██████████] | 295kB 3.2MB/s eta 0:00:0 113%
    100% [██████████] | 305kB 3.3MB/s eta 0:00:0 117%
    0% [██████████] | 315kB 3.4MB/s eta 0:00:0 121%
    22% [██████████] | 325kB 3.5MB/s eta 0:00:0 125%
    44% [██████████] | 335kB 3.6MB/s eta 0:00:0 129%
    66% [██████████] | 345kB 3.7MB/s eta 0:00:0 133%
    88% [██████████] | 355kB 3.8MB/s eta 0:00:0 137%
    100% [██████████] | 365kB 3.9MB/s eta 0:00:0 141%
    0% [██████████] | 375kB 4.0MB/s eta 0:00:0 145%
    22% [██████████] | 385kB 4.1MB/s eta 0:00:0 149%
    44% [██████████] | 395kB 4.2MB/s eta 0:00:0 153%
    66% [██████████] | 405kB 4.3MB/s eta 0:00:0 157%
    88% [██████████] | 415kB 4.4MB/s eta 0:00:0 161%
    100% [██████████] | 425kB 4.5MB/s eta 0:00:0 165%
    0% [██████████] | 435kB 4.6MB/s eta 0:00:0 169%
    22% [██████████] | 445kB 4.7MB/s eta 0:00:0 173%
    44% [██████████] | 455kB 4.8MB/s eta 0:00:0 177%
    66% [██████████] | 465kB 4.9MB/s eta 0:00:0 181%
    88% [██████████] | 475kB 5.0MB/s eta 0:00:0 185%
    100% [██████████] | 485kB 5.1MB/s eta 0:00:0 189%
    0% [██████████] | 495kB 5.2MB/s eta 0:00:0 193%
    22% [██████████] | 505kB 5.3MB/s eta 0:00:0 197%
    44% [██████████] | 515kB 5.4MB/s eta 0:00:0 201%
    66% [██████████] | 525kB 5.5MB/s eta 0:00:0 205%
    88% [██████████] | 535kB 5.6MB/s eta 0:00:0 209%
    100% [██████████] | 545kB 5.7MB/s eta 0:00:0 213%
    0% [██████████] | 555kB 5.8MB/s eta 0:00:0 217%
    22% [██████████] | 565kB 5.9MB/s eta 0:00:0 221%
    44% [██████████] | 575kB 6.0MB/s eta 0:00:0 225%
    66% [██████████] | 585kB 6.1MB/s eta 0:00:0 229%
    88% [██████████] | 595kB 6.2MB/s eta 0:00:0 233%
    100% [██████████] | 605kB 6.3MB/s eta 0:00:0 237%
    0% [██████████] | 615kB 6.4MB/s eta 0:00:0 241%
    22% [██████████] | 625kB 6.5MB/s eta 0:00:0 245%
    44% [██████████] | 635kB 6.6MB/s eta 0:00:0 249%
    66% [██████████] | 645kB 6.7MB/s eta 0:00:0 253%
    88% [██████████] | 655kB 6.8MB/s eta 0:00:0 257%
    100% [██████████] | 665kB 6.9MB/s eta 0:00:0 261%
    0% [██████████] | 675kB 7.0MB/s eta 0:00:0 265%
    22% [██████████] | 685kB 7.1MB/s eta 0:00:0 269%
    44% [██████████] | 695kB 7.2MB/s eta 0:00:0 273%
    66% [██████████] | 705kB 7.3MB/s eta 0:00:0 277%
    88% [██████████] | 715kB 7.4MB/s eta 0:00:0 281%
    100% [██████████] | 725kB 7.5MB/s eta 0:00:0 285%
    0% [██████████] | 735kB 7.6MB/s eta 0:00:0 289%
    22% [██████████] | 745kB 7.7MB/s eta 0:00:0 293%
    44% [██████████] | 755kB 7.8MB/s eta 0:00:0 297%
    66% [██████████] | 765kB 7.9MB/s eta 0:00:0 301%
    88% [██████████] | 775kB 8.0MB/s eta 0:00:0 305%
    100% [██████████] | 785kB 8.1MB/s eta 0:00:0 309%
    0% [██████████] | 795kB 8.2MB/s eta 0:00:0 313%
    22% [██████████] | 805kB 8.3MB/s eta 0:00:0 317%
    44% [██████████] | 815kB 8.4MB/s eta 0:00:0 321%
    66% [██████████] | 825kB 8.5MB/s eta 0:00:0 325%
    88% [██████████] | 835kB 8.6MB/s eta 0:00:0 329%
    100% [██████████] | 845kB 8.7MB/s eta 0:00:0 333%
    0% [██████████] | 855kB 8.8MB/s eta 0:00:0 337%
    22% [██████████] | 865kB 8.9MB/s eta 0:00:0 341%
    44% [██████████] | 875kB 9.0MB/s eta 0:00:0 345%
    66% [██████████] | 885kB 9.1MB/s eta 0:00:0 349%
    88% [██████████] | 895kB 9.2MB/s eta 0:00:0 353%
    100% [██████████] | 905kB 9.3MB/s eta 0:00:0 357%
    0% [██████████] | 915kB 9.4MB/s eta 0:00:0 361%
    22% [██████████] | 925kB 9.5MB/s eta 0:00:0 365%
    44% [██████████] | 935kB 9.6MB/s eta 0:00:0 369%
    66% [██████████] | 945kB 9.7MB/s eta 0:00:0 373%
    88% [██████████] | 955kB 9.8MB/s eta 0:00:0 377%
    100% [██████████] | 965kB 9.9MB/s eta 0:00:0 381%
    0% [██████████] | 975kB 10.0MB/s eta 0:00:0 385%
    22% [██████████] | 985kB 10.1MB/s eta 0:00:0 389%
    44% [██████████] | 995kB 10.2MB/s eta 0:00:0 393%
    66% [██████████] | 1005kB 10.3MB/s eta 0:00:0 397%
    88% [██████████] | 1015kB 10.4MB/s eta 0:00:0 401%
    100% [██████████] | 1025kB 10.5MB/s eta 0:00:0 405%
    0% [██████████] | 1035kB 10.6MB/s eta 0:00:0 409%
    22% [██████████] | 1045kB 10.7MB/s eta 0:00:0 413%
    44% [██████████] | 1055kB 10.8MB/s eta 0:00:0 417%
    66% [██████████] | 1065kB 10.9MB/s eta 0:00:0 421%
    88% [██████████] | 1075kB 11.0MB/s eta 0:00:0 425%
    100% [██████████] | 1085kB 11.1MB/s eta 0:00:0 429%
    0% [██████████] | 1095kB 11.2MB/s eta 0:00:0 433%
    22% [██████████] | 1105kB 11.3MB/s eta 0:00:0 437%
    44% [██████████] | 1115kB 11.4MB/s eta 0:00:0 441%
    66% [██████████] | 1125kB 11.5MB/s eta 0:00:0 445%
    88% [██████████] | 1135kB 11.6MB/s eta 0:00:0 449%
    100% [██████████] | 1145kB 11.7MB/s eta 0:00:0 453%
    0% [██████████] | 1155kB 11.8MB/s eta 0:00:0 457%
    22% [██████████] | 1165kB 11.9MB/s eta 0:00:0 461%
    44% [██████████] | 1175kB 12.0MB/s eta 0:00:0 465%
    66% [██████████] | 1185kB 12.1MB/s eta 0:00:0 469%
    88% [██████████] | 1195kB 12.2MB/s eta 0:00:0 473%
    100% [██████████] | 1205kB 12.3MB/s eta 0:00:0 477%
    0% [██████████] | 1215kB 12.4MB/s eta 0:00:0 481%
    22% [██████████] | 1225kB 12.5MB/s eta 0:00:0 485%
    44% [██████████] | 1235kB 12.6MB/s eta 0:00:0 489%
    66% [██████████] | 1245kB 12.7MB/s eta 0:00:0 493%
    88% [██████████] | 1255kB 12.8MB/s eta 0:00:0 497%
    100% [██████████] | 1265kB 12.9MB/s eta 0:00:0 501%
    0% [██████████] | 1275kB 13.0MB/s eta 0:00:0 505%
    22% [██████████] | 1285kB 13.1MB/s eta 0:00:0 509%
    44% [██████████] | 1295kB 13.2MB/s eta 0:00:0 513%
    66% [██████████] | 1305kB 13.3MB/s eta 0:00:0 517%
    88% [██████████] | 1315kB 13.4MB/s eta 0:00:0 521%
    100% [██████████] | 1325kB 13.5MB/s eta 0:00:0 525%
    0% [██████████] | 1335kB 13.6MB/s eta 0:00:0 529%
    22% [██████████] | 1345kB 13.7MB/s eta 0:00:0 533%
    44% [██████████] | 1355kB 13.8MB/s eta 0:00:0 537%
    66% [██████████] | 1365kB 13.9MB/s eta 0:00:0 541%
    88% [██████████] | 1375kB 14.0MB/s eta 0:00:0 545%
    100% [██████████] | 1385kB 14.1MB/s eta 0:00:0 549%
    0% [██████████] | 1395kB 14.2MB/s eta 0:00:0 553%
    22% [██████████] | 1405kB 14.3MB/s eta 0:00:0 557%
    44% [██████████] | 1415kB 14.4MB/s eta 0:00:0 561%
    66% [██████████] | 1425kB 14.5MB/s eta 0:00:0 565%
    88% [██████████] | 1435kB 14.6MB/s eta 0:00:0 569%
    100% [██████████] | 1445kB 14.7MB/s eta 0:00:0 573%
    0% [██████████] | 1455kB 14.8MB/s eta 0:00:0 577%
    22% [██████████] | 1465kB 14.9MB/s eta 0:00:0 581%
    44% [██████████] | 1475kB 15.0MB/s eta 0:00:0 585%
    66% [██████████] | 1485kB 15.1MB/s eta 0:00:0 589%
    88% [██████████] | 1495kB 15.2MB/s eta 0:00:0 593%
    100% [██████████] | 1505kB 15.3MB/s eta 0:00:0 597%
    0% [██████████] | 1515kB 15.4MB/s eta 0:00:0 601%
    22% [██████████] | 1525kB 15.5MB/s eta 0:00:0 605%
    44% [██████████] | 1535kB 15.6MB/s eta 0:00:0 609%
    66% [██████████] | 1545kB 15.7MB/s eta 0:00:0 613%
    88% [██████████] | 1555kB 15.8MB/s eta 0:00:0 617%
    100% [██████████] | 1565kB 15.9MB/s eta 0:00:0 621%
    0% [██████████] | 1575kB 16.0MB/s eta 0:00:0 625%
    22% [██████████] | 1585kB 16.1MB/s eta 0:00:0 629%
    44% [██████████] | 1595kB 16.2MB/s eta 0:00:0 633%
    66% [██████████] | 1605kB 16.3MB/s eta 0:00:0 637%
    88% [██████████] | 1615kB 16.4MB/s eta 0:00:0 641%
    100% [██████████] | 1625kB 16.5MB/s eta 0:00:0 645%
    0% [██████████] | 1635kB 16.6MB/s eta 0:00:0 649%
    22% [██████████] | 1645kB 16.7MB/s eta 0:00:0 653%
    44% [██████████] | 1655kB 16.8MB/s eta 0:00:0 657%
    66% [██████████] | 1665kB 16.9MB/s eta 0:00:0 661%
    88% [██████████] | 1675kB 17.0MB/s eta 0:00:0 665%
    100% [██████████] | 1685kB 17.1MB/s eta 0:00:0 669%
    0% [██████████] | 1695kB 17.2MB/s eta 0:00:0 673%
    22% [██████████] | 1705kB 17.3MB/s eta 0:00:0 677%
    44% [██████████] | 1715kB 17.4MB/s eta 0:00:0 681%
    66% [██████████] | 1725kB 17.5MB/s eta 0:00:0 685%
    88% [██████████] | 1735kB 17.6MB/s eta 0:00:0 689%
    100% [██████████] | 1745kB 17.7MB/s eta 0:00:0 693%
    0% [██████████] | 1755kB 17.8MB/s eta 0:00:0 697%
    22% [██████████] | 1765kB 17.9MB/s eta 0:00:0 701%
    44% [██████████] | 1775kB 18.0MB/s eta 0:00:0 705%
    66% [██████████] | 1785kB 18.1MB/s eta 0:00:0 709%
    88% [██████████] | 1795kB 18.2MB/s eta 0:00:0 713%
    100% [██████████] | 1805kB 18.3MB/s eta 0:00:0 717%
    0% [██████████] | 1815kB 18.4MB/s eta 0:00:0 721%
    22% [██████████] | 1825kB 18.5MB/s eta 0:00:0 725%
    44% [██████████] | 1835kB 18.6MB/s eta 0:00:0 729%
    66% [██████████] | 1845kB 18.7MB/s eta 0:00:0 733%
    88% [██████████] | 1855kB 18.8MB/s eta 0:00:0 737%
    100% [██████████] | 1865kB 18.9MB/s eta 0:00:0 741%
    0% [██████████] | 1875kB 19.0MB/s eta 0:00:0 745%
    22% [██████████] | 1885kB 19.1MB/s eta 0:00:0 749%
    44% [██████████] | 1895kB 19.2MB/s eta 0:00:0 753%
    66% [██████████] | 1905kB 19.3MB/s eta 0:00:0 757%
    88% [██████████] | 1915kB 19.4MB/s eta 0:00:0 761%
    100% [██████████] | 1925kB 19.5MB/s eta 0:00:0 765%
    0% [██████████] | 1935kB 19.6MB/s eta 0:00:0 769%
    22% [██████████] | 1945kB 19.7MB/s eta 0:00:0 773%
    44% [██████████] | 1955kB 19.8MB/s eta 0:00:0 777%
    66% [██████████] | 1965kB 19.9MB/s eta 0:00:0 781%
    88% [██████████] | 1975kB 20.0MB/s eta 0:00:0 785%
    100% [██████████] | 1985kB 20.1MB/s eta 0:00:0 789%
    0% [██████████] | 1995kB 20.2MB/s eta 0:00:0 793%
    22% [██████████] | 2005kB 20.3MB/s eta 0:00:0 797%
    44% [██████████] | 2015kB 20.4MB/s eta 0:00:0 801%
    66% [██████████] | 2025kB 20.5MB/s eta 0:00:0 805%
    88% [██████████] | 2035kB 20.6MB/s eta 0:00:0 809%
    100% [██████████] | 2045kB 20.7MB/s eta 0:00:0 813%
    0% [██████████] | 2055kB 20.8MB/s eta 0:00:0 817%
    22% [██████████] | 2065kB 20.9MB/s eta 0:00:0 821%
    44% [██████████] | 2075kB 21.0MB/s eta 0:00:0 825%
    66% [██████████] | 2085kB 21.1MB/s eta 0:00:0 829%
    88% [██████████] | 2095kB 21.2MB/s eta 0:00:0 833%
    100% [██████████] | 2105kB 21.3MB/s eta 0:00:0 837%
    0% [██████████] | 2115kB 21.4MB/s eta 0:00:0 841%
    22% [██████████] | 2125kB 21.5MB/s eta 0:00:0 845%
    44% [██████████] | 2135kB 21.6MB/s eta 0:00:0 849%
    66% [██████████] | 2145kB 21.7MB/s eta 0:00:0 853%
    88% [██████████] | 2155kB 21.8MB/s eta 0:00:0 857%
    100% [██████████] | 2165kB 21.9MB/s eta 0:00:0 861%
    0% [██████████] | 2175kB 22.0MB/s eta 0:00:0 865%
    22% [██████████] | 2185kB 22.1MB/s eta 0:00:0 869%
    44% [██████████] | 2195kB 22.2MB/s eta 0:00:0 873%
    66% [██████████] | 2205kB 22.3MB/s eta 0:00:0 877%
    88% [██████████] | 2215kB 22.4MB/s eta 0:00:0 881%
    100% [██████████] | 2225kB 22.5MB/s eta 0:00:0 885%
    0% [██████████] | 2235kB 22.6MB/s eta 0:00:0 889%
    22% [██████████] | 2245kB 22.7MB/s eta 0:00:0 893%
    44% [██████████] | 2255kB 22.8MB/s eta 0:00:0 897%
    66% [██████████] | 2265kB 22.9MB/s eta 0:00:0 901%
    88% [██████████] | 2275kB 23.0MB/s eta 0:00:0 905%
    100% [██████████] | 2285kB 23.1MB/s eta 0:00:0 909%
    0% [██████████] | 2295kB 23.2MB/s eta 0:00:0 913%
    22% [██████████] | 2305kB 23.3MB/s eta 0:00:0 917%
    44% [██████████] | 2315kB 23.4MB/s eta 0:00:0 921%
    66% [██████████] | 2325kB 23.5MB/s eta 0:00:0 925%
    88% [██████████] | 2335kB 23.6MB/s eta 0:00:0 929%
    100% [██████████] | 2345kB 23.7MB/s eta 0:00:0 933%
    0% [██████████] | 2355kB 23.8MB/s eta 0:00:0 937%
    22% [██████████] | 2365kB 23.9MB/s eta 0:00:0 941%
    44% [██████████] | 2375kB 24.0MB/s eta 0:00:0 945%
    66% [██████████] | 2385kB 24.1MB/s eta 0:00:0 949%
    88% [██████████] | 2395kB 24.2MB/s eta 0:00:0 953%
    100% [██████████] | 2405kB 24.3MB/s eta 0:00:0 957%
    0% [██████████] | 2415kB 24.4MB/s eta 0:00:0 961%
    22% [██████████] | 2425kB 24.5MB/s eta 0:00:0 965%
    44% [██████████] | 2435kB 24.6MB/s eta 0:00:0 969%
    66% [██████████] | 2445kB 24.7MB/s eta 0:00:0 973%
    88% [██████████] | 2455kB 24.8MB/s eta 0:00:0 977%
    100% [██████████] | 2465kB 24.9MB/s eta 0:00:0 981%
    0% [██████████] | 2475kB 25.0MB/s eta 0:00:0 985%
    22% [██████████] | 2485kB 25.1MB/s eta 0:00:0 989%
    44% [██████████] | 2495kB 25.2MB/s eta 0:00:0 993%
    66% [██████████] | 2505kB 25.3MB/s eta 0:00:0 997%
    88% [██████████] | 2515kB 25.4MB/s eta 0:00:0 1001%
    100% [██████████] | 2525kB 25.5MB/s eta 0:00:0 1005%
```

Figure 1.48: Install Library Joblib

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

Figure 1.49: Berhasil Import Library Joblib

Chapter 2

Related Works

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

|||||| HEAD

2.1 Mhd Zulfikar Akram Nasution/ 1164081

2.1.1 Teori

1. Binary Classification atau diartikan kedalam bahasa indonesia yaitu Klasifikasi Biner adalah tugas dalam mengklarifikasi elemen-elemen dari himpunan yang diberikan kedalam dua kelompok berdasarkan aturan klarifikasi. Pada umumnya klarifikasi biner akan jatuh ke dalam domain Supervised Learning dan dimana kasus khusus hanya memiliki dua kelas.

- Contoh Binary Classification pada gambar 2.1

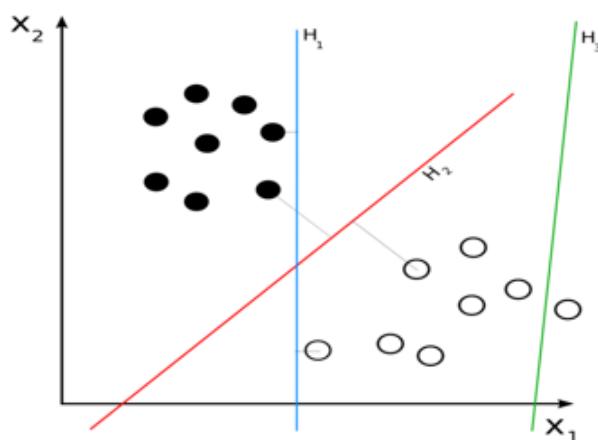


Figure 2.1: Binary Classification

2. Supervised Learning, Unsupervised Learning, dan Clustering

- Supervised Learning

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). Contoh pada gambar 2.2

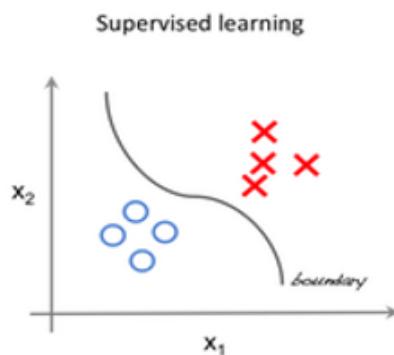


Figure 2.2: Supervised Learning

- Unsupervised Learning

Unsupervised Learning merupakan sebuah data yang belum ditentukan variabelnya jadi hanya berupa data saja. Dalam sebuah kasus Unsupervised Learning adalah aggap saja anda belum pernah membeli buku sama sekali dan pada suatu hari anda telah membeli buku dengan sangat banyak dalam kategori yang berbeda. Sehingga buku tersebut belum di kategorikan dan hanya berupa data buku saja. Coontoh seperti pada gambar 2.3

- Clustering

Classtering merupakan sebuah proses untuk mengklasifikasikan sebuah data dalam satu parameter. Dalam kasus ini dapat dijelaskan ada beberapa orang yang memiliki kekuatan tubuh yang sehat dan kekuatan tubuh yang lemah. Parameter bagi orang yang memiliki tubuh yang kuat adalah orang yang terlihat bugar dan sehat maka dengan orang yang memiliki parameter adalah orang

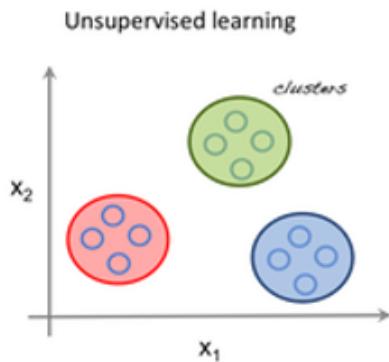


Figure 2.3: Unsupervised Learning

yang memiliki kekuatan tubuh yang kuat dan untuk kekuatan tubuh yang lemah adalah sebaliknya. Contoh seperti pada gambar 2.4

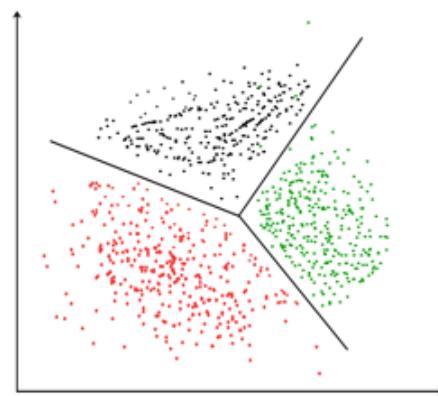


Figure 2.4: Clustering

3. Evaluasi dan Akurasi

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.

4. Cara Membuat dan Membaca Confusion Matrix

- Tentukan pokok permasalahan dan atributnya

- Buat Decicion Tree
- Buat Data Testing
- Mencari nilai variabelnya misal a,b,c, dan d
- Mencari nilai recall, percision, accuracy, dan error rate

contoh confusion matrix

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

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

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

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

5. Cara Kerja K-Fold Cross Validation

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

Berikut ilustrasi K-Fold Cross Validation seperti pada gambar 2.5

6. Decision Tree

Decision Tree adalah sebuah metode pembelajaran yang digunakan untuk melakukan klarifikasi dan regresi. Decision Tree digunakan untuk membuat sebuah model yang dapat memprediksi sebuah nilai variabel target dengan cara mempelajari aturan keputusan dari fitur data. Contohnya seperti pada gambar 2.6

7. Gain dan Entropi



Figure 2.5: K-Fold Cross Validation

- Gain adalah pengurangan yang diharapkan dalam entropy. Dalam machine learning, gain dapat digunakan untuk menentukan sebuah urutan atribut atau memperkecil atribut yang telah dipilih. Urutan ini akan membentuk decision tree. atribut gain dipilih yang paling besar.
- Entropi adalah ukuran ketidakpastian sebuah variabel acak sehingga dapat diartikan entropi adalah ukuran ketidakpastian dari sebuah atribut.

Contoh seperti pada gambar 2.7

2.2 Mhd Zulfikar Akram Nasution/ 1164081

=====

2.3 puad hamdani/1164084

2.3.1 binary classification

1. Output aktual dari banyak algoritma binary classification adalah skor prediksi. Skor menunjukkan kepastian sistem bahwa pengamatan yang diberikan adalah milik kelas positif. Untuk membuat keputusan tentang apakah pengamatan harus diklasifikasikan sebagai positif atau negatif

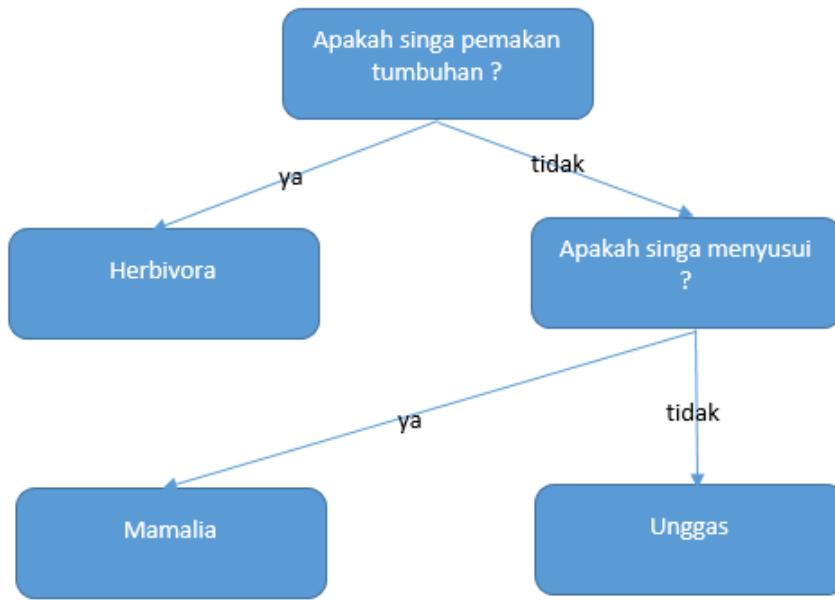


Figure 2.6: Decision Tree

2.3.2 supervised learning dan unsupervised learning

1. Supervised learning adalah sebuah pendekatan dimana sudah terdapat data yang dilatih, dan terdapat variable yang di targetkan sehingga tujuan dari pendekatan ini adalah mengelompokkan suatu data ke data yang sudah ada.
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. Unsupervised learning tidak memiliki data latih, sehingga dari data yang ada, kita mengelompokkan data tersebut menjadi dua bagian atau tiga bagian dan seterusnya.

2.3.3 evaluasi dan akurasi

1. Evaluasi merupakan suatu proses identifikasi untuk mengukur atau menilai apakah suatu kegiatan/program yang dilaksanakan sesuai dengan perencanaan atau tujuan yang ingin dicapai. Akurasi merupakan tingkat kedekatan pengukuran kuantitas terhadap nilai yang sebenarnya. Kepresisian dari suatu sistem pengukuran



Figure 2.7: Gain dan Entropi

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
- 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.

2.3.6 decision tree

1. Decision tree merupakan model prediksi menggunakan struktur pohon atau struktur berhirarki.

2.3.7 Information Gain

1. Information gain. Metode tersebut akan melakukan proses komputasi untuk mendapatkan atribut-atribut yang paling berpengaruh terhadap dataset

2.4 Puad Hamdani/ 1164084

abaa009ed6a2bf07ff821b0fed9603a84340748c

2.4.1 Scikit-Learn

```
1. # load dataset (student mat pakenya)
import pandas as pd
<<<<< HEAD
lontong      = pd.read_csv('student-mat.csv', sep=';')
len(lontong)
```

```

In [1]: import pandas as pd
...: lontong = pd.read_csv('student-por.csv', sep=';')
...: len(lontong)
Out[1]: 649

```

Figure 2.8: Load Dataset

Codingan pertama ini akan meload (menampilkan) data pada file yang ditentukan. Untuk codingan ini file yang dieksekusi ialah ” student-mat.csv ” . Secara jelasnya, dalam codingan dapat dilihat bahwa variabel lontong didefinisikan untuk pembacaan csv dari ” lontong ” dimana untuk pemisahnya yaitu separation berupa ; . Setelah itu variabel lontong di ”print” dengan perintah menampilkan ”len” panjang ataupun jumlah dan hasilnya berupa angka 649 .

=====

```

sate      = pd.read_ csv('student-mat.csv', sep=';')
len(sate)

```

```

In [2]: import pandas as pd
...: sate = pd.read_csv('student-por.csv', sep=';')
...: len(sate)
Out[2]: 649

```

Figure 2.9: Load Dataset

Codingan ini akan meload (menampilkan) data pada file yang ditentukan. hasilnya berupa angka 649 .

```

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

```

Codingan kedua ini secara keseluruhan menampilkan baris G1, G2 dan G3 (berdasarkan kriterianya) untuk kolom PASS pada variabel lontong. Untuk lebih jelasnya, pada codingan terdapat pendefinisian pembacaan ”lambda” (panjang gelombang) dari baris G1, G2 dan G3. Apabila row-row tersebut bernilai lebih dari 35 maka akan terdefinisikan angka ”1” apabila tidak, maka akan terdefinisikan angka ”0” pada kolom PASS (sesuai permintaan awal).

```

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

[5 rows x 31 columns]

```

Figure 2.10: Generate Binary Label

Selanjutnya variabelnya di ”print” sehingga menampilkan keluaran. Tidak lupa terdapat juga jumlah dari baris dan kolom yang terubah sesuai dengan baris yang dieksekusi.

```

3. # use one-hot encoding on categorical columns
lontong = pd.get_dummies(lontong, columns=['sex', 'school', 'address',
=====

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

In [3]: sate['pass'] = sate.apply(lambda row: 1 if (row['G1']+row['G2']+row['G3']) >= 35 else 0, axis=1)
...: sate = sate.drop(['G1', 'G2', 'G3'], axis=1)
...: sate.head()
Out[3]:

```

Figure 2.11: Generate Binary Label

Codingan ini secara keseluruhan menampilkan baris G1, G2 dan G3 (berdasarkan kriterianya). Untuk lebih jelasnya, pada codingan terdapat pendefinisian pembacaan ”lambda” (panjang gelombang) dari baris G1, G2 dan G3. Apabila row-row tersebut bernilai lebih dari 35 maka akan terdefinisikan angka ”1” apabila tidak, maka akan terdefinisikan angka ”0”

```

4. # use one-hot encoding on categorical columns
sate = pd.get_dummies(sate, columns=['sex', 'school', 'address',
>>>>> abaa009ed6a2bf07ff821b0fed9603a84340748c
'famsize',
'Pstatus', 'Mjob', 'Fjob',
'reason', 'guardian', 'schoolsupsup',
'famsup', 'paid', 'activities',
'nursery', 'higher', 'internet',

```

```

'romantic'])

<<<<< HEAD

lontong.head()

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

```

Figure 2.12: Pemanggilan get dummies dari lontong

Secara keseluruhan, codingan ini mendefinisikan pemanggilan get dummies ke dalam variabel lontong. Di dalam get dummies sendiri akan terdefinisikan variabel lontong dengan kolom-kolom yang akan dieksekusi seperti school, address dll. Kemudian variabel tersebut di definisikan untuk mendapatkan kembalian berupa keluaran dari eksekusi perintah variabel lontong beserta dengan jumlah baris dan kolom data yang dieksekusi. =====

```

sate.head()

In [4]: sate = pd.get_dummies(sate, columns=['sex', 'school', 'address', 'famsize', 'Pstatus', 'Mjob', 'Fjob',
...:                                         'reason', 'guardian', 'schoolsup', 'famsup', 'paid', 'activities',
...:                                         'nursery', 'higher', 'internet', 'romantic'])
...: sate.head()
Out[4]:
   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

```

Figure 2.13: Pemanggilan get dummies

Secara keseluruhan, codingan ini mendefinisikan pemanggilan get dummies ke dalam variabel sate. *ffffabaa009ed6a2bf07ff821b0fed9603a84340748c*

```

5. # shuffle rows
<<<<< HEAD

lontong = lontong.sample(frac=1)
# split training and testing data
lontong_train = d[:500]
lontong_test = d[500:]

```

```

lontong_train_att = lontong_train.drop(['pass'], axis=1)
lontong_train_pass = lontong_train['pass']

lontong_test_att = lontong_test.drop(['pass'], axis=1)
lontong_test_pass = lontong_test['pass']

lontong_att = lontong.drop(['pass'], axis=1)
lontong_pass = lontong['pass']
=====

sate = sate.sample(frac=1)
# split training and testing data
sate_train = d[:500]
sate_test = d[500:]

sate_train_att = sate_train.drop(['pass'], axis=1)
sate_train_pass = sate_train['pass']

sate_test_att = sate_test.drop(['pass'], axis=1)
sate_test_pass = sate_test['pass']

sate_att = sate.drop(['pass'], axis=1)
sate_pass = sate['pass']
>>>>> abaa009ed6a2bf07ff821b0fed9603a84340748c

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

```

Secara keseluruhan codingan ini difungsikan untuk mendefinisikan pembagian data yang berupa training dan testing data. Secara jelasnya pertama-tama variabel lontong akan mendefinisikan sampel yang akan digunakan (berupa shuffle row). Nah kemudian masing2 parameter yaitu lontong train dan lontong test akan berjumlah 500 data (telah dibagi untuk training dan testing). Selanjutnya dilakukan pengeksekusian untuk kolom Pass, apabila sesuai dengan

```
In [4]: lontong = lontong.sample(frac=1)
...: # split training and testing data
...: lontong_train = lontong[:500]
...: lontong_test = lontong[500:]
...:
...: lontong_train_att = lontong_train.drop(['pass'], axis=1)
...: lontong_train_pass = lontong_train['pass']
...:
...: lontong_test_att = lontong_test.drop(['pass'], axis=1)
...: lontong_test_pass = lontong_test['pass']
...:
...: lontong_att = lontong.drop(['pass'], axis=1)
...: lontong_pass = lontong['pass']
...:
...: # number of passing students in whole dataset:
...: import numpy as np
...: print("Passing: %d out of %d (%.2f%%)" % (np.sum(lontong_pass), len(lontong_pass), 100*float(np.sum(lontong_pass)) / len(lontong_pass)))
Passing: 328 out of 649 (50.54%)
```

Figure 2.14: Mendefinisikan pembagian data

”axis=1” maka eksekusi fungsi berhasil. Selain itu juga disertakan jumlah dari peserta yang lolos dari semua nilai data setnya. =====

```
100*float(np.sum(sate_pass)) / len(sate_pass)))
```

```
In [5]: sate = sate.sample(frac=1)
...: # split training and testing data
...: sate_train = sate[:500]
...: sate_test = sate[500:]
...:
...: sate_train_att = sate_train.drop(['pass'], axis=1)
...: sate_train_pass = sate_train['pass']
...:
...: sate_test_att = sate_test.drop(['pass'], axis=1)
...: sate_test_pass = sate_test['pass']
...:
...: sate_att = sate.drop(['pass'], axis=1)
...: sate_pass = sate['pass']
...:
...: # number of passing students in whole dataset:
...: import numpy as np
...: print("Passing: %d out of %d (%.2f%%)" % (np.sum(sate_pass), len(sate_pass), 100*float(np.sum(sate_pass)) / len(sate_pass)))
Passing: 328 out of 649 (50.54%)
```

Figure 2.15: Mendefinisikan pembagian data

Secara keseluruhan codingan ini difungsikan untuk mendefinisikan pembagian data yang berupa training dan testing data &&&&& abaa009ed6a2bf07ff821b0fed9603a8434074

```
6. # fit a decision tree
from sklearn import tree
<<<<< HEAD
soto = tree.DecisionTreeClassifier(criterion="entropy", max_depth=5)
soto = soto.fit(lontong_train_att, lontong_train_pass)
```

Secara keseluruhan, codingan ini hanya membuktikan pengujian dari Klasifikasi Decision Tree yang ada, apakah true atau tidak dan hasilnya true. Apabila dibahas secara lengkap maka pada codingan ini di definisikan library sklearn untuk mengimpor atau menampilkan tree. Variabel soto difungsikan untuk

```
In [5]: from sklearn import tree
...: soto = tree.DecisionTreeClassifier(criterion="entropy", max_depth=5)
...: soto = soto.fit(lontong_train_att, lontong_train_pass)
```

Figure 2.16: Membuktikan pengujian

membaca klasifikasi decision tree dari tree itu sendiri dengan 2 parameternya yaitu kriteria="entropy" dan max depth=5. Maka selanjutnya variabel soto akan masuk dan terbaca dalam module fit dengan 2 parameter yaitu lontong trai att dan lontong train pass. =====

```
rendang = tree.DecisionTreeClassifier(criterion="entropy", max_depth=5)
rendang = rendang.fit(sate_train_att, sate_train_pass)
```

```
In [6]: from sklearn import tree
...: rendang = tree.DecisionTreeClassifier(criterion="entropy", max_depth=5)
...: rendang = rendang.fit(sate_train_att, sate_train_pass)
```

Figure 2.17: Membuktikan pengujian

Secara keseluruhan, codingan ini hanya membuktikan pengujian dari Klasifikasi Decision Tree yang ada, apakah true atau tidak dan hasilnya true ????? abaa009ed6a2bf07ff821b0fed9603a84340748c

```
7. # visualize tree
import graphviz
<<<<< HEAD
dot_data = tree.export_graphviz(soto, out_file=None, label="all",
                                impurity=False, proportion=True,
                                feature_names=list(lontong_train_att),
=====

dot_data = tree.export_graphviz(rendang, out_file=None, label="all",
                                impurity=False, proportion=True,
                                feature_names=list(sate_train_att),
>>>>> abaa009ed6a2bf07ff821b0fed9603a84340748c
                                class_names=["fail", "pass"],
                                filled=True, rounded=True)

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

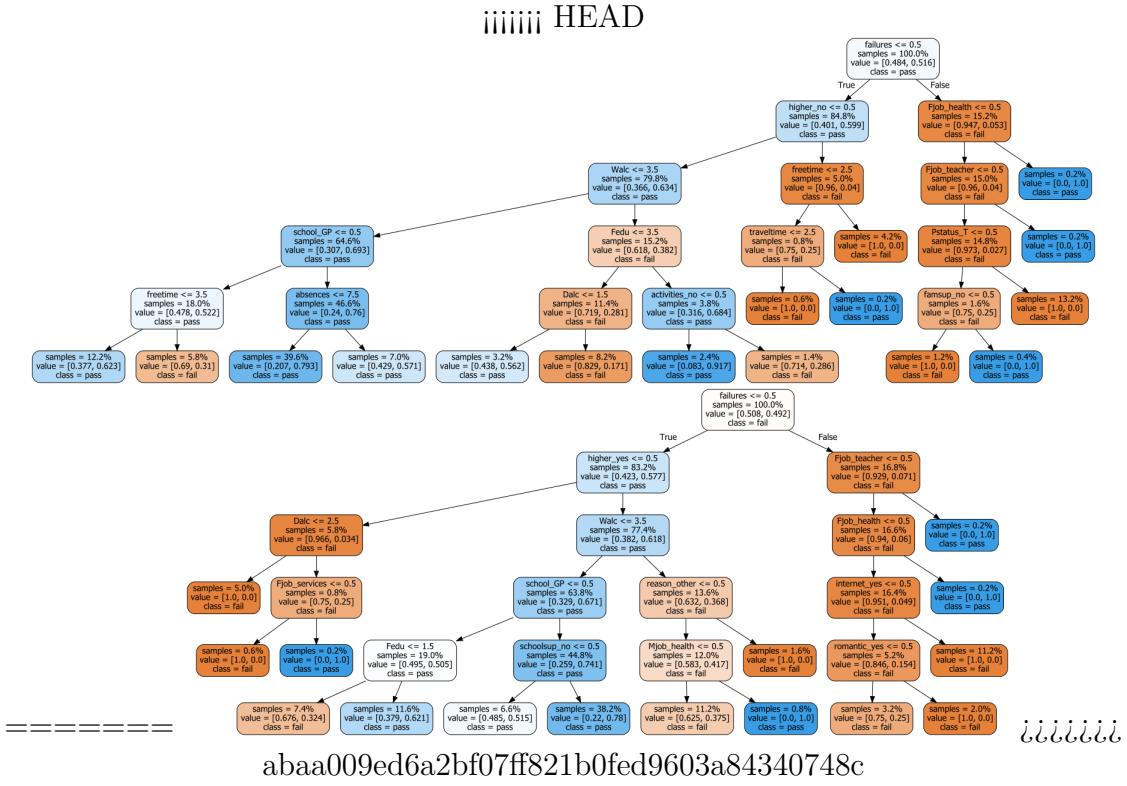


Figure 2.18: Gambaran decision tree

Codingan ini memberikan gambaran dari klasifikasi decision tree dari pengolahan parameter yang dieksekusi kedalam variabel soto. Tentunya dengan permanfaatan library graphviz yang telah diimport dan difungsikan.

```

8. # save tree
<<<<< HEAD
tree.export_graphviz(soto, out_file="student-performance.dot",
label="all", impurity=False,
proportion=True,
feature_names=list(lontong_train_att),
=====
tree.export_graphviz(rendang, out_file="student-performance.dot",
label="all", impurity=False,
proportion=True,
feature_names=list(sate_train_att),
>>>>> abaa009ed6a2bf07ff821b0fed9603a84340748c
class_names=["fail", "pass"],
filled=True, rounded=True)

```

```

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

```

Figure 2.19: Library Graphviz

Pada gambar 7 akan menampilkan yang terdapat pada Library Graphviz, apabila benar akan menampilkan hasil output seperti yang terdapat pada gambar.

9. soto.score(lontong_test_att, lontong_test_pass)

```

In [8]: soto.score(lontong_test_att, lontong_test_pass)
Out[8]: 0.6375838926174496

```

Figure 2.20: Menampilkan hasil perhitungan 2 parameter

Menampilkan hasil perhitungan dari kedua parameter yang terdapat pada code tersebut. Yang merupakan perhitungan hasil prediksi silang akan kemungkinan nilai di masa mendatang. =====

```

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

```

Figure 2.21: Library Graphviz

Pada gambar 4.6 akan menampilkan yang terdapat pada Library Graphviz,

10. rendang.score(sate_test_att, sate_test_pass)

merupakan perhitungan hasil prediksi silang akan kemungkinan nilai di masa mendatang. jjjjjj abaa009ed6a2bf07ff821b0fed9603a84340748c

```

11. from sklearn.model_selection import cross_val_score
<<<<< HEAD
kari = cross_val_score(soto, lontong_att, lontong_pass, cv=5)
# show average score and +/- two standard deviations away
#(covering 95% of scores)
print("Accuracy: %0.2f (+/- %0.2f)" % (kari.mean(), kari.std() * 2))

```

Kodingan tersebut mendefinisikan library sklearn model selection dan import cross val score. Dan kemudian variabel kari mengeksekusi fungsi cross val score(soto, lontong att, lontong pass, cv=5). Kemudian akan menampilkan nilai dari fungsi akurasinya.

```
In [10]: rendang.score(sate_test_att, sate_test_pass)
Out[10]: 0.6778523489932886
```

Figure 2.22: Menampilkan hasil perhitungan 2 parameter

```
In [9]: from sklearn.model_selection import cross_val_score
...: kari = cross_val_score(soto, lontong_att, lontong_pass, cv=5)
...: # show average score and +/- two standard deviations away (covering 95% of scores)
...: print("Accuracy: %0.2f (+/- %0.2f)" % (kari.mean(), kari.std() * 2))
Accuracy: 0.67 (+/- 0.05)
```

Figure 2.23: Mendefinisikan library sklearn

```
12. for max_depth in range(1, 20):
    soto = tree.DecisionTreeClassifier(criterion="entropy",
    max_depth=max_depth)
    kari = cross_val_score(soto, lontong_att, lontong_pass, cv=5)
    print("Max depth: %d, Accuracy: %0.2f (+/- %0.2f)" %
    (max_depth, kari.mean(), kari.std() * 2))
    =====
    semur = cross_val_score(rendang, sate_att, sate_pass, cv=5)
    # show average score and +/- two standard deviations away
    #(covering 95% of scores)
    print("Accuracy: %0.2f (+/- %0.2f)" % (semur.mean(), semur.std() * 2))

In [11]: from sklearn.model_selection import cross_val_score
...: semur = cross_val_score(rendang, sate_att, sate_pass, cv=5)
...: # show average score and +/- two standard deviations away (covering 95% of scores)
...: print("Accuracy: %0.2f (+/- %0.2f)" % (semur.mean(), semur.std() * 2))
Accuracy: 0.69 (+/- 0.02)
```

Figure 2.24: Mendefinisikan library sklearn

Kodingan tersebut mendefinisikan library sklearn model selection dan import cross val score

```
13. for max_depth in range(1, 20):
    rendang = tree.DecisionTreeClassifier(criterion="entropy",
    max_depth=max_depth)
    semur = cross_val_score(rendang, sate_att, sate_pass, cv=5)
    print("Max depth: %d, Accuracy: %0.2f (+/- %0.2f)" %
    (max_depth, semur.mean(), semur.std() * 2))
>>>>> abaa009ed6a2bf07ff821b0fed9603a84340748c
    )
```

```

In [10]: for max_depth in range(1, 20):
...:     soto = tree.DecisionTreeClassifier(criterion="entropy", max_depth=max_depth)
...:     kari = cross_val_score(soto, lontong_att, lontong_pass, cv=5)
...:     print("Max depth: %d, Accuracy: %.2f (+/- %.2f)" % (max_depth, kari.mean(), kari.std() * 2))
Max depth: 1, Accuracy: 0.62 (+/- 0.05)
Max depth: 2, Accuracy: 0.69 (+/- 0.03)
Max depth: 3, Accuracy: 0.69 (+/- 0.07)
Max depth: 4, Accuracy: 0.69 (+/- 0.03)
Max depth: 5, Accuracy: 0.67 (+/- 0.06)
Max depth: 6, Accuracy: 0.68 (+/- 0.08)
Max depth: 7, Accuracy: 0.67 (+/- 0.06)
Max depth: 8, Accuracy: 0.66 (+/- 0.05)
Max depth: 9, Accuracy: 0.67 (+/- 0.05)
Max depth: 10, Accuracy: 0.67 (+/- 0.07)
Max depth: 11, Accuracy: 0.66 (+/- 0.06)
Max depth: 12, Accuracy: 0.65 (+/- 0.09)
Max depth: 13, Accuracy: 0.64 (+/- 0.12)
Max depth: 14, Accuracy: 0.65 (+/- 0.06)
Max depth: 15, Accuracy: 0.64 (+/- 0.08)
Max depth: 16, Accuracy: 0.64 (+/- 0.12)
Max depth: 17, Accuracy: 0.65 (+/- 0.08)
Max depth: 18, Accuracy: 0.63 (+/- 0.08)
Max depth: 19, Accuracy: 0.64 (+/- 0.09)

```

|||||| HEAD

Figure 2.25: Menampilkan hasil fungsi max depth dan accuracy

Pada gambar di atas kodingan nya berfungsi untuk menampilkan hasil dari fungsi Max Depth dan Accuraccy dari dari Decission Tree. Yaitu menmpilkan data dari angka 1-20.

```

In [12]: for max_depth in range(1, 20):
...:     rendang = tree.DecisionTreeClassifier(criterion="entropy", max_depth=max_depth)
...:     semur = cross_val_score(rendang, sate_att, sate_pass, cv=5)
...:     print("Max depth: %d, Accuracy: %.2f (+/- %.2f)" % (max_depth, semur.mean(), semur.std() * 2))
Max depth: 1, Accuracy: 0.64 (+/- 0.05)
Max depth: 2, Accuracy: 0.69 (+/- 0.08)
Max depth: 3, Accuracy: 0.70 (+/- 0.05)
Max depth: 4, Accuracy: 0.69 (+/- 0.06)
Max depth: 5, Accuracy: 0.69 (+/- 0.01)
Max depth: 6, Accuracy: 0.66 (+/- 0.05)
Max depth: 7, Accuracy: 0.66 (+/- 0.05)
Max depth: 8, Accuracy: 0.66 (+/- 0.08)
Max depth: 9, Accuracy: 0.66 (+/- 0.07)
Max depth: 10, Accuracy: 0.64 (+/- 0.07)
Max depth: 11, Accuracy: 0.62 (+/- 0.09)
Max depth: 12, Accuracy: 0.63 (+/- 0.08)
Max depth: 13, Accuracy: 0.62 (+/- 0.04)
Max depth: 14, Accuracy: 0.63 (+/- 0.09)
Max depth: 15, Accuracy: 0.62 (+/- 0.08)
Max depth: 16, Accuracy: 0.63 (+/- 0.08)
Max depth: 17, Accuracy: 0.62 (+/- 0.07)
Max depth: 18, Accuracy: 0.63 (+/- 0.05)
Max depth: 19, Accuracy: 0.62 (+/- 0.06)

```

Figure 2.26: Menampilkan hasil fungsi max depth dan accuracy

menmpilkan data dari angka 1-20.

```

14. depth_acc = np.empty((19,3), float)
    i = 0
    for max_depth in range(1, 20):
        soto = tree.DecisionTreeClassifier(criterion="entropy",
        max_depth=max_depth)
    <<<<< HEAD
        kari = cross_val_score(soto, lontong_att, lontong_pass, cv=5)
        depth_acc[i,0] = max_depth

```

```

depth_acc[i,1] = kari.mean()
depth_acc[i,2] = kari.std() * 2
=====
semur = cross_val_score(rendang, sate_att, sate_pass, cv=5)
depth_acc[i,0] = max_depth
depth_acc[i,1] = semur.mean()
depth_acc[i,2] = semur.std() * 2
>>>>> abaa009ed6a2bf07ff821b0fed9603a84340748c
i += 1

depth_acc

In [11]: depth_acc = np.empty((19,3), float)
...: i = 0
...: for max_depth in range(1, 20):
...:     soto = tree.DecisionTreeClassifier(criterion="entropy", max_depth=max_depth)
...:     kari = cross_val_score(soto, lontong_att, lontong_pass, cv=5)
...:     depth_acc[i,0] = max_depth
...:     depth_acc[i,1] = kari.mean()
...:     depth_acc[i,2] = kari.std() * 2
...:     i += 1
...:
...:
...: depth_acc
Out[11]:
array([[ 1.        ,  0.61642391,  0.04864866],
       [ 2.        ,  0.68717239,  0.03294537],
       [ 3.        ,  0.69024858,  0.06707175],
       [ 4.        ,  0.69028454,  0.02835026],
       [ 5.        ,  0.68400024,  0.06793328],
       [ 6.        ,  0.67011794,  0.07465326],
       [ 7.        ,  0.66244912,  0.0626756 ],
       [ 8.        ,  0.67170392,  0.04275525],
       [ 9.        ,  0.66715919,  0.04225535],
       [10.       ,  0.67166796,  0.08363672],
       [11.       ,  0.65321744,  0.09661167],
       [12.       ,  0.65792805,  0.0735665 ],
       [13.       ,  0.65341855,  0.10020057],
       [14.       ,  0.63796237,  0.10110564],
       [15.       ,  0.65824731,  0.06521759],
       [16.       ,  0.62419861,  0.11884634],
       [17.       ,  0.6348616 ,  0.10853295],
       [18.       ,  0.64887435,  0.10398688],
       [19.       ,  0.63648336,  0.08535061]]))

|||||| HEAD

```

Figure 2.27: Menjelaskan variable kari

Dijelaskan bahwa variable kari akan menampilkan atau mendefinisikan nilai dari variabel score yang mana isi dari variable score yaitu soto, lontong att, lon-tong pass, cv=5. Yang mana hasil tampilan dari kodingannya adalah outputan seperti gambar 11. =====

Bahwa variable semur akan menampilkan atau mendefinisikan nilai dari variabel score yang mana isi dari variable score yaitu rendang ||||||, abaa009ed6a2bf07ff821b0fed9603a84340748c

```

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

```

```

In [13]: depth_acc = np.empty((19,3), float)
...: i = 0
...: for max_depth in range(1, 20):
...:     rendang = tree.DecisionTreeClassifier(criterion="entropy", max_depth=max_depth)
...:     semur = cross_val_score(rendang, state_att, state_pass, cv=5)
...:     depth_acc[i,0] = max_depth
...:     depth_acc[i,1] = semur.mean()
...:     depth_acc[i,2] = semur.std() * 2
...:     i += 1
...:
...:
...: depth_acc
Out[13]:
array([[ 1.        ,  0.63779741,  0.0507489 ],
       [ 2.        ,  0.68704156,  0.07858131],
       [ 3.        ,  0.70247443,  0.0525174 ],
       [ 4.        ,  0.69185823,  0.06240303],
       [ 5.        ,  0.69801281,  0.02145731],
       [ 6.        ,  0.65641344,  0.05413215],
       [ 7.        ,  0.660083947,  0.06754535],
       [ 8.        ,  0.65177421,  0.06648372],
       [ 9.        ,  0.65482764,  0.06422626],
       [10.        , 0.625552459,  0.09252526],
       [11.        , 0.64392777,  0.08378081],
       [12.        , 0.62396282,  0.05819135],
       [13.        , 0.62238803,  0.10767083],
       [14.        , 0.62551321,  0.08211362],
       [15.        , 0.63327653,  0.06220894],
       [16.        , 0.62548899,  0.07332788],
       [17.        , 0.6270636 ,  0.06425702],
       [18.        , 0.6193001 ,  0.07943198],
       [19.        , 0.61016364,  0.05593769]])

```

Figure 2.28: Menjelaskan variable kari

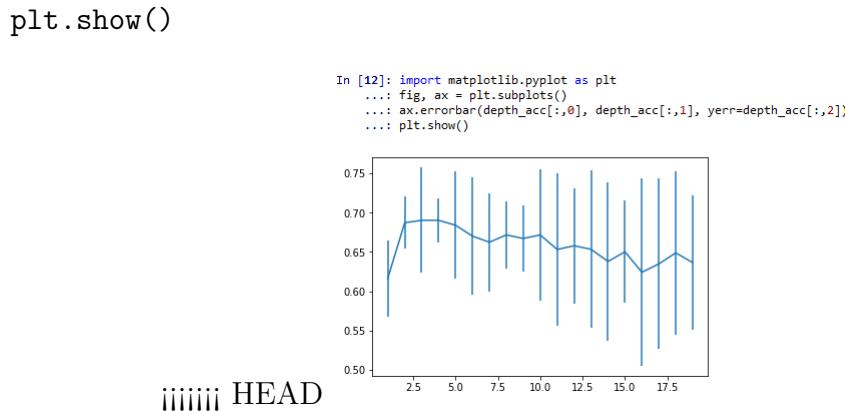


Figure 2.29: Menjelaskan dan menampilkan gambar grafik

Pada gambar di atas dijelaskan bahwa pada library matplotlib akan menampilkan gambar grafik pada gambar 12 dari eksekusi fungsi `ax.errorbar`. ======
menampilkan gambar grafik pada gambar 4.11 dari eksekusi fungsi `ax.errorbar`.
===== abaa009ed6a2bf07ff821b0fed9603a84340748c

2.5 Penanganan Error

===== HEAD Dari percobaan yang dilakukan di atas, error yang kita dapatkan di dokumentasikan dan di selesaikan(nilai 5 hari kedua):

1. ScreenShoot Error

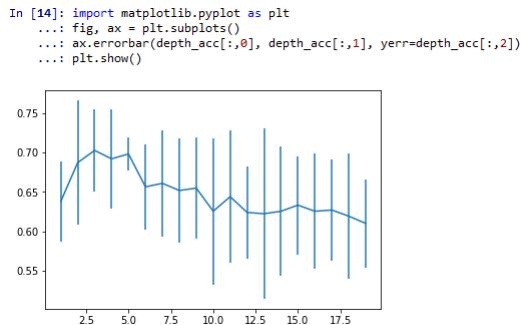


Figure 2.30: Menjelaskan dan menampilkan gambar grafik

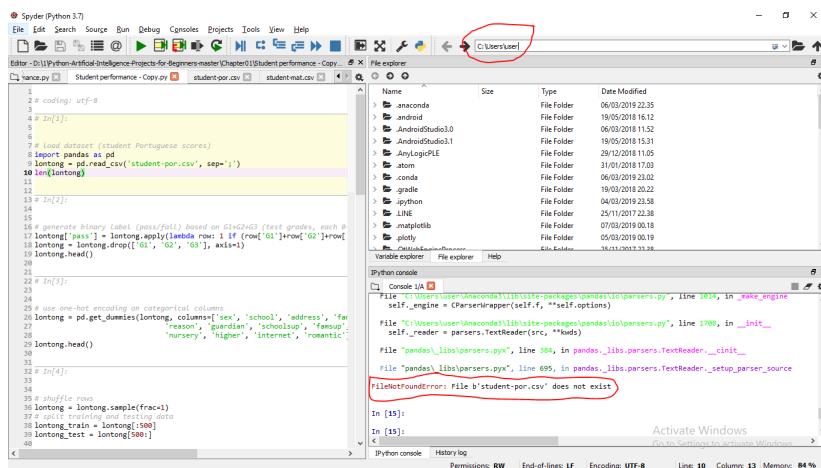


Figure 2.31: ScreenShot Error

2. Tuliskan kode eror dan jenis errornya

Error ini disebabkan karena pada direktori C tidak terdapat file tersebut.

3. Solusi pemecahan masalah error tersebut

- Masuk ke folder dimana file dataset berada, dapat dilihat dibawah ini
- Setelah diganti, jalankan kembali skrip tersebut pasti akan berhasil

===== Dari percobaan yang dilakukan di atas, tidak ada eror:

???????? abaa009ed6a2bf07ff821b0fed9603a84340748c

2.6 Same Topics

Cite every latest journal with same topic

```

1 # coding: utf-8
2
3 # In[1]:
4
5
6
7 # load dataset (Student Portuguese scores)
8 import pandas as pd
9 lontong = pd.read_csv('student-por.csv', sep=';')
10 len(lontong)
11
12
13 # In[2]:
14
15
16 # generate binary labels (pass/fail) based on grades<3 (test grades much <
17 lontong['pass'] = lontong.apply(lambda row: 1 if (row['G1']+row['G2']+row['G3'])/3<3 else 0, axis=1)
18 lontong = lontong.drop(['G1', 'G2', 'G3'], axis=1)
19 lontong.head()
20
21
22 # In[3]:
23
24
25 # use one-hot encoding on categorical columns
26 lontong = pd.get_dummies(lontong, columns=['sex', 'school', 'address', 'fam-
27 'reason', 'guardian', 'schoolsup', 'famsup',
28 'nursery', 'higher', 'internet', 'romantic'],
29 lontong.head()
30
31
32 # In[4]:
33
34
35 # shuffle rows
36 lontong = lontong.sample(frac=1)
37 # split training and testing data
38 lontong_train = lontong[300:]
39 lontong_test = lontong[300:]
40

```

Figure 2.32: Penanganan Error

2.6.1 Topic 1

cite for first topic

2.6.2 Topic 2

if you have two topics you can include here to

2.7 Same Method

write and cite latest journal with same method

2.7.1 Method 1

cite and paraphrase method 1

2.7.2 Method 2

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

||||| HEAD

2.8 Teori/Jesron Marudut Hatuan/1164077

2.8.1 Binary classification dilengkapi ilustrasi gambar

1. Binary classification yaitu berupa kelas-kelas positif dan kelas-kelas negatif. Klasifikasi biner adalah dikotomisasi yang diterapkan untuk tujuan praktis dan

dalam banyak masalah klasifikasi biner praktis, kedua kelompok tidak simetris - daripada 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 atau tidak mendeteksi penyakit ketika hadir.

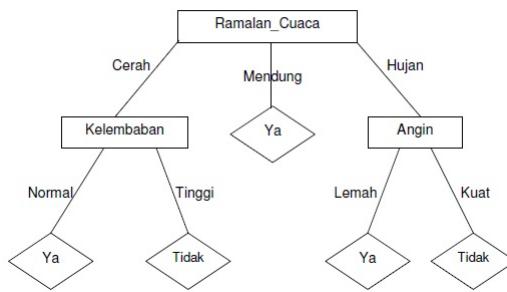


Figure 2.33: Klasifikasi Binari

2.8.2 Pengertian Supervised Learning, Unsupervised Learning dan Clustering dan Illustrasi gambar

1. Supervised learning adalah sesuatu pembelajaran yang terawasi yang dimana jika output yang diharapkan telah diketahui sebelum-sebelumnya. Biasanya Supervised Learning ini dilakukan dengan menggunakan data yang sudah ada. Pada metode ini, setiap pola yang diberikan kedalam jaringan saraf tiruan setelah diketahui outputnya. Satu pola input akan diberikan ke satu neuron pada lapisan-lapisan input. Pola-pola ini akan dirambatkan di sepanjang jaringan syaraf hingga sampai ke neuron pada lapisan output tersebut. 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.
2. Unsupervised learning merupakan sebuah pembelajaran yang tidak terawasi dimana tidak memerlukan target output. Pada metode ini tidak dapat ditentukan hasil seperti apa yang diharapkan selama proses pembelajaran, nilai bobot yang disusun dalam proses range tertentu tergantung pada sebuah nilai output yang telah diberikan. Tujuan metode unsupervised learning ini agar dapat mengelompokkan unit-unit yang hampir sama dalam satu area tertentu. Pembelajaran ini biasanya sangat cocok untuk klasifikasi pola-pola. Contohnya algoritma

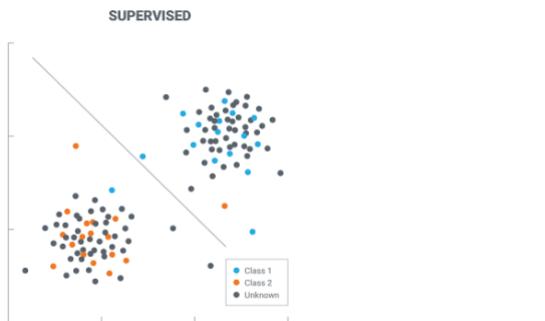


Figure 2.34: Supervised Learning

jaringan saraf tiruan yang menggunakan metode unsupervised ini merupakan competitive, kohonen, LVQ(Learning Vector Quantization), neocognitron.

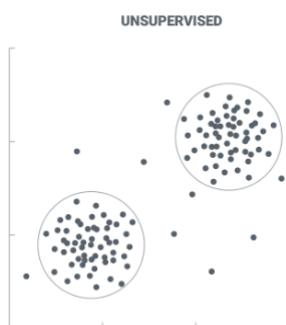


Figure 2.35: Unsupervised Learning

3. Analisis Cluster adalah sebuah teknik statistika yang dapat berguna untuk mengelompokkan sebuah objek-objek ataupun variable-variable ke dalam beberapa grup tertentu dimana pada setiap objek atau variable yang telah terbentuk mempunyai sifat dan karakteristik yang berdekatan tersebut. 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.

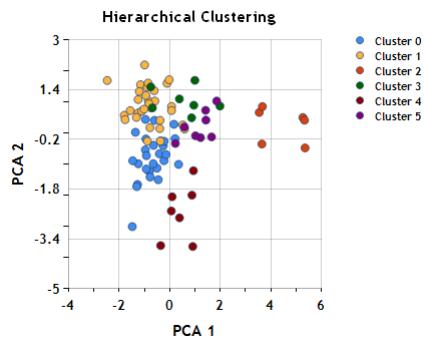


Figure 2.36: Cluster

2.8.3 Evaluasi dan akurasi dan Illustrasi gambar

1. Evaluasi merupakan suatu langkah bagaimana agar dapat mengevaluasi seberapa baik model bekerja dengan cara mengukur tingkat akurasinya. Dan akurasi tersebut akan didefinisikan menjadi sebuah persentasi studi kasus yang telah diklasifikasikan dengan benar. Dapat juga untuk menganalisis kesalahan yang dibuat oleh sebuah model, atau tingkat kebingungannya, menggunakan matriks kebingungan. Matriks kebingungan mengacu pada sebuah kebingungan dalam model, tetapi matriks kebingungan ini bisa menjadi sedikit sulit untuk dipahami ketika mereka menjadi sangat besar.

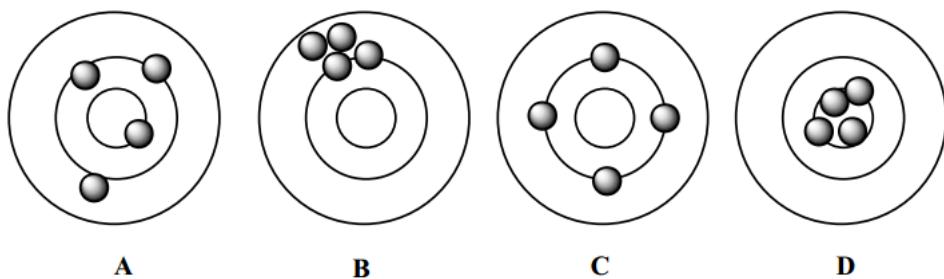


Figure 2.37: Evaluasi dan Akurasi

2.8.4 Cara membuat dan membaca confusion matrix, buat confusion matrix

1. Cara membuat dan membaca confusion matrix :

- 1) Menentukan pokok sebuah permasalahan dan atribut-atributnya, misal gaji dan listik.
- 2) Buat pohon keputusan

- 3) Lalu data testingnya
 - 4) Lalu mencari nilai a, b, c, dan d. Semisal a = 5, b = 1, c = 1, dan d = 3.
 - 5) Selanjutnya mencari nilai recall, precision, accuracy, serta dan error rate.
2. Berikut adalah contoh dari confusion matrix :
- Recall = $3/(1+3) = 0,75$
 - Precision = $3/(1+3) = 0,75$
 - Accuracy = $(5+3)/(5+1+1+3) = 0,8$
 - Error Rate = $(1+1)/(5+1+1+3) = 0,2$

2.8.5 Membuat cara 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.

2.8.6 Decision tree dengan gambar ilustrasi

1. Decision Tree adalah metode pembelajaran yang diawasi non-parametrik yang digunakan untuk mengklasifikasi dan regresi. Tujuannya adalah untuk membuat model yang memprediksi nilai variabel target dengan mempelajari aturan keputusan sederhana yang disimpulkan dari fitur data. Decision tree memadukan antara eksplorasi data dan sebuah pemodelan, sehingga sangat bagus sebagai

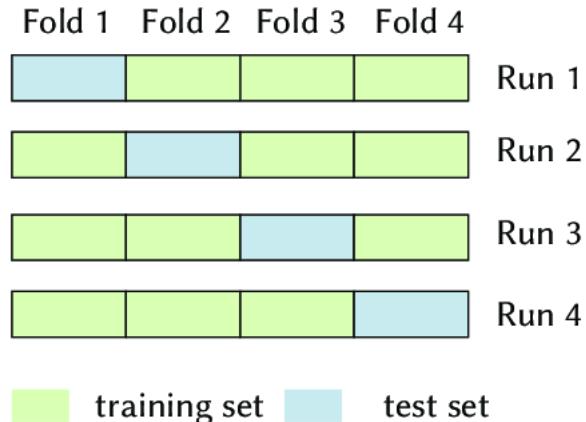


Figure 2.38: K-fold cross validation

langkah awal dalam proses pemodelan bahkan ketika dijadikan menjadi sebuah model akhir dari beberapa teknik lain.

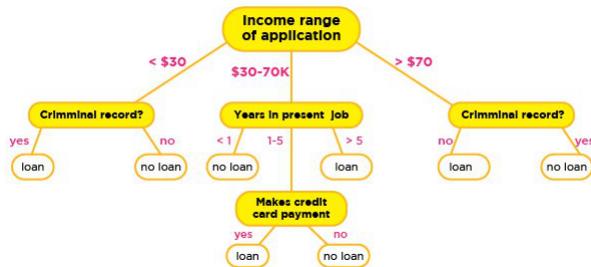


Figure 2.39: Decision Tree

2.8.7 Information Gain dan entropi dengan gambar ilustrasi

- Information gain dapat didasarkan kepada penurunan entropi setelah dataset yang sebelumnya dapat dibagi pada atributnya. Untuk membangun sebuah decision tree, merupakan cara agar semua tentang menemukan atribut yang mengembalikan perolehan informasi tertinggi dari yang lainnya.
- Entropi adalah cara untuk mengukur keacakan dalam sebuah sistem informasi yang sedang diproses. Semakin tinggi entropi, maka hasilnya semakin sulit untuk menarik kesimpulan dari informasi tersebut. Melempar koin merupakan salah satu contoh tindakan yang memberikan informasi yang random. Untuk koin

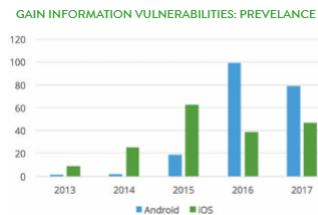


Figure 2.40: Information gain

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.

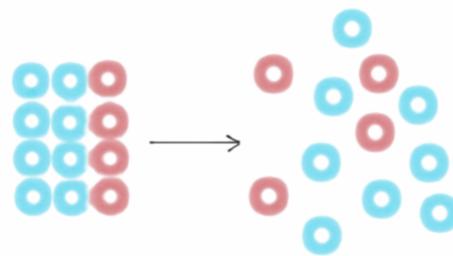


Figure 2.41: Entropi

2.8.8 Scikit-learn

- Penjelasan kodingan ini akan menampilkan data pada file yang ditentukan. Untuk codingan ini file yang dieksekusi untuk digunakan ialah student-mat.csv. Dalam codingan dapat dilihat bahwa variabel Rambutan dapat didefinisikan untuk pembacaan file csv dari Durian dimana untuk pemisahnya yaitu separation berupa ; . Setelah itu variabel Rambutan di tampilkan dengan perintah menampilkan len panjang ataupun jumlah dan hasilnya berupa angka 395 .

```
In [1]: import pandas as Durian
...: Rambutan = Durian.read_csv('D:\KULIAH\COLLAGE SPACE\SEMESTER
6\Chapter01\dataset\student-mat.csv',sep=';')
...: len(Rambutan)
Out[1]: 395
```

Figure 2.42: Load Dataset

- Kodingan berikut berguna untuk menampilkan setiap baris G1, G2 dan G3 untuk kolom PASS pada variabel Rambutan. Untuk lebih jelasnya, pada kodingan terdapat pendefinisian pembacaan lamda (panjang gelombang) dari baris G1,

G2 dan G3. Apabila row-row tersebut bernilai lebih dari 35 maka akan terdefinisikan angka 1 apabila tidak, maka akan terdefinisikan angka 0 pada kolom PASS (sesuai permintaan awal). Selanjutnya variabelnya di ditampilkan sehingga menampilkan keluaran. Tidak lupa terdapat juga jumlah dari baris dan kolom yang terubah sesuai dengan baris yang dieksekusi.

```
In [2]: Rambutan['pass'] = Rambutan.apply(lambda row: 1 if (row['G1']+row['G2']+row['G3']) >= 35 else 0, axis=1)
....: Rambutan = Rambutan.drop(['G1', 'G2', 'G3'], axis=1)
....: Rambutan.head()
Out[2]:
   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 2.43: Generate Binary label

1. Kodingan berikut mendefinisikan pemanggilan get dummies dari Durian dalam variabel Rambutan. Di dalam get dummies sendiri akan terdefinisikan variabel Rambutan dengan kolom-kolom yang akan dieksekusi seperti school, address dll. Kemudian variabel tersebut diartikan untuk mendapatkan kembalian berupa keluaran dari eksekusi perintah variabel Rambutan beserta dengan jumlah baris dan kolom data yang dieksekusi.

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

Figure 2.44: Use one-hot encoding

1. Kodingan ini dipakai untuk mengartikan pembagian data yang berupa training dan testing data. Pertama-tama variabel Rambutan akan mengartikan sampel yang akan digunakan. Selanjutnya masing-masing parameter yaitu Rambutan train dan Rambutan test akan berjumlah 500 data. Selanjutnya dilakukan pengeksekusian untuk kolom Pass, apabila sesuai dengan axis=1 maka eksekusi fungsi berhasil. Selain itu juga disertakan jumlah dari peserta yang lolos dari semua nilai data setnya.

```

In [4]: Rambutan = Rambutan.sample(frac=1)
...: # split training and testing data
...: Rambutan_train = Rambutan[:500]
...: Rambutan_test = Rambutan[500:]
...:
...: Rambutan_train_att = Rambutan_train.drop(['pass'], axis=1)
...: Rambutan_train_pass = Rambutan_train['pass']
...:
...: Rambutan_test_att = Rambutan_test.drop(['pass'], axis=1)
...: Rambutan_test_pass = Rambutan_test['pass']
...:
...: Rambutan_att = Rambutan.drop(['pass'], axis=1)
...: Rambutan_pass = Rambutan['pass']
...:
...: # number of passing students in whole dataset:
...: import numpy as np
...: print("Passing: %d out of %d (%.2f%%)" % (np.sum(Rambutan_pass),
...: len(Rambutan_pass), 100*float(np.sum(Rambutan_pass)) / len(Rambutan_pass)))
Passing: 166 out of 395 (42.03%)

```

Figure 2.45: Shuffle rows

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

```

In [5]: from sklearn import tree
...: timun = tree.DecisionTreeClassifier(criterion="entropy", max_depth=5)
...: timun = timun.fit(Rambutan_train_att, Rambutan_train_pass)

```

Figure 2.46: Fit a Decision tree

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

```

File "C:\ProgramData\Anaconda3\lib\site-packages\graphviz
\bckend.py", line 150, in run
    raise ExecutableNotFoundError(cmd)

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

Out[6]: <graphviz.files.Source at 0x9a8eb10>

```

Figure 2.47: Visualize tree

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

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

Figure 2.48: Save tree

- Penjelasan kodingan ini membaca skore dari variabel timun dimana terdapat 2 parameter yang dihitung dan diuji yaitu Rambutan test att dan Rambutan test pass. Untuk hasilnya sendiri mengapa berupa angka, dikarenakan pada parameter yang dieksekusi memang memiliki data sehingga dieksekusi dan menghasilkan keluaran dari score tersebut.

```
File "C:\ProgramData\Anaconda3\lib\site-packages\sklearn
utils\validation.py", line 582, in check_array
context)

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

Figure 2.49: Score

- Penjelasan kodingan ini membahas mengenai pengkesekusian fungsi dan variabel dari library yang didefinisikan dan yang diimport. Penjelasan lebih jelasnya ialah kodingan ini mendefinisikan library sklearn.model.selection kemudian mengimport cross_val_score. Kemudian variabel score mendefinisikan cross_val_score yang telah diimport tadi dengan 4 parameter yaitu timun, Rambutan att, Rambutan pass dan cv=5 untuk dieksekusi. Setelah semua pemrosesan tersebut maka hasil yang ditampilkan ialah rata-rata perhitungan dari variabel score dimana standar dari plus minusnya tentunya dengan ketentuan parameter Accuracy .

```
In [9]: from sklearn.model_selection import cross_val_score
...: scores = cross_val_score(timun, Rambutan_att, Rambutan_pass, cv=5)
...: # show average score and +/- two standard deviations away (covering 95%
of scores)
...: print("Accuracy: %.2f (+/- %.2f)" % (scores.mean(), scores.std() *
2))
Accuracy: 0.56 (+/- 0.06)
```

Figure 2.50: Show Average score

- Penjelasan kodingan ini mendefinisikan max_depth dalam jarak angka antara parameter 1 dan 20. Variabel timun mendefinisikan klasifikasi decision tree dengan 2 parameter. Kemudian variabel score mengeksekusi parameter lainnya yaitu seperti timun, Rambutan att, Rambutan pass dan cv=5). Hasil yang ditampilkan ialah dari max_depth, accuracy dan plus minusnya dan akhirnya hasil outputannya keluar.

```

In [10]: for max_depth in range(1, 20):
...:     timun = tree.DecisionTreeClassifier(criterion="entropy",
...:                                         max_depth=max_depth)
...:     scores = cross_val_score(timun, Rambutan_att, Rambutan_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: 0.59 (+/- 0.10)
Max depth: 3, Accuracy: 0.57 (+/- 0.07)
Max depth: 4, Accuracy: 0.57 (+/- 0.08)
Max depth: 5, Accuracy: 0.56 (+/- 0.06)
Max depth: 6, Accuracy: 0.58 (+/- 0.09)
Max depth: 7, Accuracy: 0.59 (+/- 0.09)
Max depth: 8, Accuracy: 0.58 (+/- 0.07)
Max depth: 9, Accuracy: 0.63 (+/- 0.11)
Max depth: 10, Accuracy: 0.62 (+/- 0.17)
Max depth: 11, Accuracy: 0.66 (+/- 0.11)
Max depth: 12, Accuracy: 0.66 (+/- 0.12)
Max depth: 13, Accuracy: 0.59 (+/- 0.11)
Max depth: 14, Accuracy: 0.59 (+/- 0.10)
Max depth: 15, Accuracy: 0.58 (+/- 0.11)
Max depth: 16, Accuracy: 0.58 (+/- 0.10)
Max depth: 17, Accuracy: 0.60 (+/- 0.12)
Max depth: 18, Accuracy: 0.60 (+/- 0.11)
Max depth: 19, Accuracy: 0.59 (+/- 0.12)

```

Figure 2.51: Max depth

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

```

In [11]: depth_acc = np.empty((19,3), float)
...: i = 0
...: for max_depth in range(1, 20):
...:     timun = tree.DecisionTreeClassifier(criterion="entropy",
...:                                         max_depth=max_depth)
...:     scores = cross_val_score(timun, Rambutan_att, Rambutan_pass, cv=5)
...:     depth_acc[i,0] = max_depth
...:     depth_acc[i,1] = scores.mean()
...:     depth_acc[i,2] = scores.std() * 2
...:     i += 1
...:
...: depth_acc
Out[11]:
array([[1.00000000e+00, 5.79751704e-01, 6.30768599e-03],
[2.00000000e+00, 5.92247647e-01, 1.00846269e-01],
[3.00000000e+00, 5.66996105e-01, 7.07275934e-02],
[4.00000000e+00, 5.74782538e-01, 8.45615429e-02],
[5.00000000e+00, 5.56837777e-01, 4.66204714e-02],
[6.00000000e+00, 5.64465271e-01, 5.69082201e-02],
[7.00000000e+00, 5.92313372e-01, 8.54962326e-02],
[8.00000000e+00, 5.79689224e-01, 4.21178600e-02],
[9.00000000e+00, 6.22663096e-01, 9.17537994e-02],
[1.00000000e+01, 6.07344206e-01, 1.45170768e-01],
[1.10000000e+01, 5.94877475e-01, 1.56585356e-01],
[1.20000000e+01, 6.10068150e-01, 1.00017897e-01],
[1.30000000e+01, 6.07536514e-01, 8.10373343e-02],
[1.40000000e+01, 6.05068150e-01, 9.49300894e-02],
[1.50000000e+01, 5.97409932e-01, 7.67838312e-02],
[1.60000000e+01, 6.05037326e-01, 8.09688960e-02],
[1.70000000e+01, 5.94877475e-01, 1.03320077e-01],
[1.80000000e+01, 5.62379098e-01, 1.06720186e-01],
[1.90000000e+01, 6.04908309e-01, 1.11344069e-01]])

```

Figure 2.52: Depth Acc

1. Kodingan mendefinisikan pemanggilan dari library matplotlib.pyplot sebagai

salak sehingga nanti hasilnya akan berbentuk gambar grafik/gelombang. Untuk variabel fig dan ax akan mendefinisikan subplots dari salak. Setelah itu ketentuan dari parameter depth acc = 0, depth acc = 1 dan depth acc 2. Selanjutnya untuk menampilkan gelombang maka panggil variabel salak dengan perintah show.

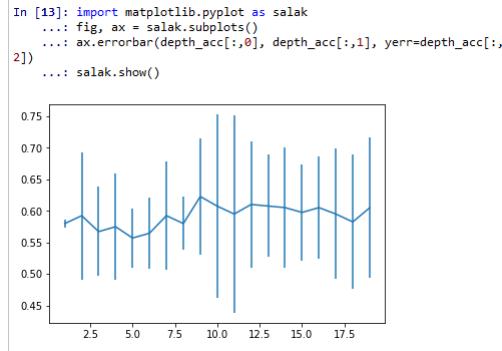


Figure 2.53: Import

2.8.9 Penanganan Eror

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

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

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

=====

ljjjjjj abaa009ed6a2bf07ff821b0fed9603a84340748c

Chapter 3

Methods

3.1 JESRON MARUDUT HATUAN/1164077

3.1.1 Teori

Penyelesaian Tugas Harian 5

1. Random Forest dan Ilustrasi Gambar

- Pengertian Random Forest:

Random forests merupakan sebuah metode dalam pembelajaran untuk klasifikasi, regresi dan tugas-tugas lain yang telah berperasi dengan membangun banyak pohon keputusan pada saat latihan hingga menciptakan kelas yang merupakan mode kelas atau klasifikasi atau beberapa prediksi dari setiap tree atau pohon.

- Ilustrasi Gambar Random Forest :

2. Langkah-langkah Membaca Dataset

Berikut adalah langkah-langkah membaca dataset :

- Pertama-tama buka aplikasi Anaconda Navigator lalu jalankan Syder, kemudian import libraries yang mau dibuat
- Selanjutnya masukkan kode python untuk membaca file csv, lalu run aplikasinya
- Maka pada jendela konsol akan menampilkan pesan berikut:
- Dan dari jendela explorer akan tampil dataset yang sudah diimport.
- Selanjutnya klik dataset cell, maka akan muncul seperti berikut :

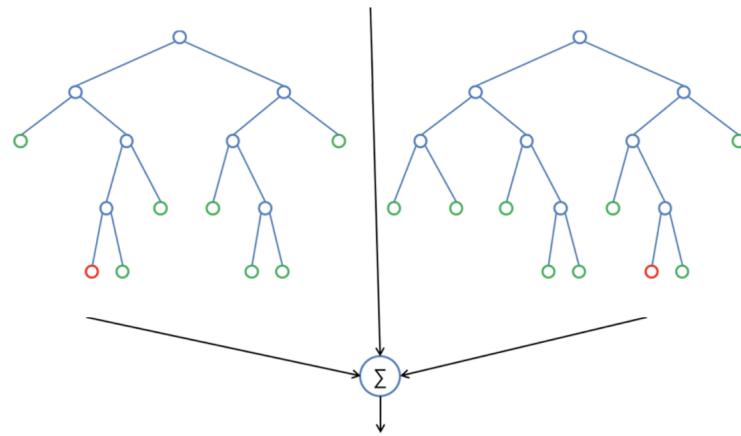


Figure 3.1: Gambar Random Forest

```
dataset = pd.read_csv('Data.csv')
```

Figure 3.2: Gambar Code Python

- Seperti yang terlihat pada gambar tersebut dataset ini memiliki Kolom Country, Age, dan Salary sebagai kolom
- Purchased sebagai dependent variable-nya.
- Selanjutnya buat 2 matrix of features yang berisi values dari independent variable dan dependent variable.
- Selanjutnya, masukan perintah berikut :
- Perintah yang telah dilakukan gunanya untuk menampilkan dataset.
- Lalu klick dataset tersebut maka muncul tabel berisi dataset.

3. Cross Validation

Cross Validation adalah sebuah teknik untuk memvalidasi model agar dapat menilai bagaimana hasil dari sebuah statistik analisis yang akan menggeneralisasi kumpulan beberapa data independen. Teknik ini lebih sering digunakan untuk melakukan prediksi model dan memperkirakan seberapa akurat sebuah model prediktif ketika sedang dijalankan dalam sebuah praktiknya. Dalam sebuah masalah prediksi, sebuah model biasanya diberikan kumpulan data (dataset) yang telah diketahui untuk digunakan dalam menjalankan pelatihan (dataset

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

Figure 3.3: Gambar Output

Name	Type	Size	Value
dataset	DataFrame	(10, 4)	Column names: Country, Age, Salary, Purchased

Figure 3.4: Gambar Import Dataset

pelatihan), serta kumpulan data yang tidak diketahui (atau data yang pertama kali dilihat) terhadap model yang diuji (pengujian dataset)

4. Maksud dari score 44% pada random forest, 27% pada decision tree dan 29% dari SVM.

Maksud dari score 27% pada decision tree adalah hasil dari presentasi dari perhitungan dataset, sedangkan maksud dari score 29% pada SVM adalah dengan pendekatan jaringan saraf. Hasilnya didapat dari validasi silang untuk memastikan bahwa membagi training test dengan cara yang berbeda. Sehingga outputnya 44% untuk hutan acak, 27% untuk pohon keputusan, dan 29% untuk SVM.

5. 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:

Selanjutnya data testingnya adalah

Pertama-pertama, kita lakukan adalah mencari 4 nilai yaitu a,b,c, dan d:

$$a= 5$$

$$b= 1$$

$$c= 1$$

$$d= 3$$

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

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

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

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	40	nan	Yes
5	France	35	58000	Yes
6	Spain	nan	52000	No
7	France	48	76000	Yes
8	Germany	58	83000	No
9	France	37	67000	Yes

Figure 3.5: Gambar Hasil Dataset Sel

```
dataset = read.csv('Data.csv')
```

Figure 3.6: Gambar Masukkan Perintah

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

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

6. Voting Random Forest Dan Ilustrasi Gambarnya.

- Pengertian Voting pada Random Forest Voting yaitu suara untuk setiap target yang diprediksi pada saat melakukan Random Forest. Pertimbangkan target prediksi dengan voting tertinggi sebagai prediksi akhir dari algoritma random forest.
- Gambar Voting Random Forest :

3.2 The data

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

3.3 Puad Hamdani/ 1164084

3.3.1 Teori

1. Random Forest

Merupakan classifier yang terdiri dari banyak pohon keputusan dan melakukan klasifikasi berdasarkan keluaran dari hasil klasifikasi setiap pohon keputusan anggota

2. Cara membaca dataset kasus

- Buka aplikasi spyder untuk membuka dan membaca kodingan dataset
- Kemudian buat variable imgatt untuk memasukkan atribut label

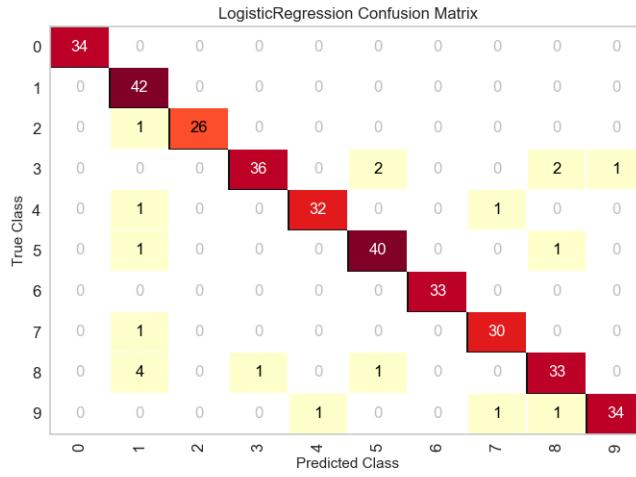


Figure 3.7: Pohon Keputusan

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

Figure 3.8: Gambar Data Testing

- Lalu uji coba kodingan untuk mengetahui apa hasil dari dataset tersebut
- imgatt.head() untuk melihat sebagian data awal
- .shape untuk melihat jumlah data
- .pivot untuk merubah atribut menjadi kolom

Dengan menguji coba kodingan yang ada pada spyder untuk membaca data set.

3. Cross Validation

Cross Validation adalah teknik validasi model untuk menilai bagaimana hasil analisis statistik (model) akan digeneralisasi ke kumpulan data independen. terutama digunakan dalam pengaturan di mana tujuannya adalah prediksi, dan orang ingin memperkirakan seberapa akurat model prediksi akan dilakukan dalam praktek

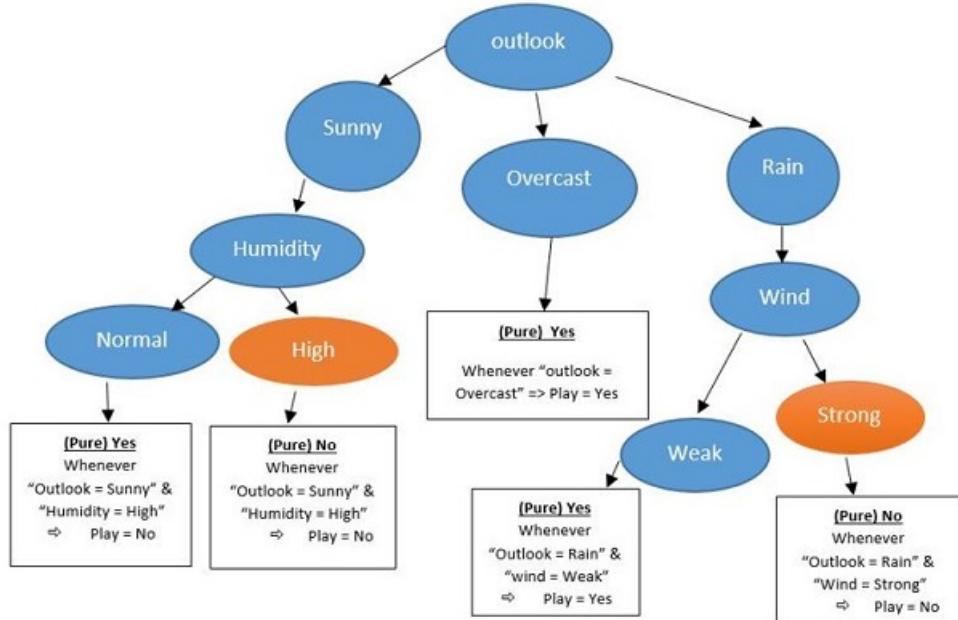


Figure 3.9: Gambar Voting Random Forest



Figure 3.10: Random Forest

4. Arti 44 persen pada RF, 27 persen pada Decission Tree, dan 29 persen pada SVM.

- Merupakan akurasi dari sebuah pohon keputusan untuk menunjukkan hasil keputusan dengan klasifikasi dari dataset yang ada.

5. Confusion Matrix

- Import confusion matrix
- Plot confusion matrix
- Lalu sesuaikan plotnya
- Setelah itu plot kembali

6. Voting

Voting Merupakan metode untuk menentukan keputusan dalam suatu pemilihan, berdasarkan pendapat per orang, dan keputusan ditentukan berdasarkan pemilih terbanyak

3.4 Jesron Marudut Hatuan / 1164077

3.4.1 Praktek

Tugas Harian ke-2

1. Aplikasi Sederhana Pandas dan Penjelasan Code (Perbaris)

- Padas:
 - Baris1 : Mengimport library pandas dari python dengan inisiasi pd.
 - Baris 2 : Parameter data ini diisi dengan data yang akan dibuat seriess.
 - Baris 3 : Mengubah data karakter menjadi series
 - Baris 4 : Mempulkan karakter
- Hasil:

```
2 import pandas as pd
3 data = np.array(['De Gea', 'Pogba', 'Rashford', 'Lindgard', 'Lukaku'])
4 karakter = pd.Series(data)
5 print(karakter)
```

Figure 3.11: Applikasi sederhana Pandas

• Aplikasi Sederhana Numpy dan Penjelasan Code (Perbaris)

- Code Numpy:
 - * Baris 1 : Mengimport library numpy dari python dengan inisiasi np
 - * Baris 2 : Parameter data ini diisi dengan data yang akan dibuat seriess.
 - * Baris 3 : Mengubah data karakter menjadi series
 - * Baris 4 : Mempulkan karakter

– Hasil:

– Aplikasi Sederhana Matplotlib dan Penjelasan Code (Perbaris)

* Code Matplotlib:

- Baris 1 : Mengimport library matplotlib dari python dengan inisiasi jes.

```

2 import numpy as np
3 data = np.array(['MANCHESTER UNITED', 'LIVERPOOL', 'CHELSEA', 'BARCELONA', 'PSG'])
4 karakter = pd.Series(data)
5 print(karakter)

```

Figure 3.12: Applikasi Numpy

- Baris 2 : Menampilkan variabel dari x dan variabel y
- Baris 3 : Menampilkan inisiasi jes dari variable X dan Y
- * Hasil:

```

19
20 import matplotlib.pyplot as jes
21 jes.plot([4,8,12,17,20],[50, 67, 98, 78, 45])
22 jes.show()

```

Figure 3.13: Applikasi Matplotlib

- Program Aplikasi Random Forest dan Penjelasan Keluarannya :
- * Kode Random Forest 1 :

```

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

```

Figure 3.14: Gambar ke-1

- Penjelasan : Untuk membaca dataset. Kodingan tersebut menghasilkan variabel baru yaitu imgatt dan terdapat 3 kolom dan 3677856 baris data.
- * Kode Random Forest 2 :
 - Penjelasan : Kodingan berikut berfungsi untuk melihat sebagian data awal dari dataset. Dan hasilnya akan terdapat pada gambar di atas setelah di eksekusi.
- * Kode Random Forest 3 :

```

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 3.15: Gambar ke-2

```

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

```

Figure 3.16: Gambar ke-3

- Penjelasan : Kodingan diatas merupakan gambaran untuk menampilkan hasil dari dataset yang telah di eksekusi. Dimana pada gambar di atas 3677856 merupakan baris dan 3 adalah kolom.
- * Kode Random Forest 4 :

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

Figure 3.17: Gambar ke-4

- Penjelasan : Gambar di atas menampilkan hasil dari variabel imgatt2. Dimana index nya 'imgid', kolom berisi 'attid' dan values atau nilainya berisi 'present'.
- * Kode Random Forest 5 :
 - Penjelasan :Gambar diatas menampilkan hasil dari variabel imgatt2.head, dimana dataset nya ada 5 baris dan 312 kolom.
- * Kode Random Forest 6 :
 - Penjelasan : Gambar diatas menampilkan jumlah dari baris dan kolom dari variabel imgatt2.
- * Kode Random Forest 7 :
 - Penjelasan : Gambar di atas menunjukkan hasil muat dari jawa-bannya 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.
- * Kode Random Forest 8 :

```

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

```

Figure 3.18: Gambar ke-5

```

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

```

Figure 3.19: Gambar ke-6

- Penjelasan : Gambar di atas menunjukkan hasil dari variabel imglabels. Dimana menampilkan dataset dari imgid dan label. Dan dapat dilihat hasilnya dari gambar di atas.
- * Kode Random Forest 9 :
 - Penjelasan : Gambar di atas menunjukkan jumlah baris dan kolom dari variabel imglabels. Dimana hasil dari kodingan tersebut dapat dilihat setelah di run.
- * Kode Random Forest 10 :
 - Penjelasan : Gambar diatas diakibatkan isi yang sama, maka dapat melakukan join antara dua data yang diesekusi (yaitu ada imgatt2 dan imglabels), sehingga pada hasilnya akan didapatkan data ciri dan data jawaban 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.
- * Kode Random Forest 11 :
 - Penjelasan : Gambar di atas menghasilkan pemisahan dan pemilihan tabel (memisahkan dan memilih tabel).
- * Kode Random Forest 12 :
 - Penjelasan : Gambar di atas menunjukkan hasil dari variabel dtatthead yang dimana data nya dapat dilihat pada gambar diatas. Dan dataset nya terdiri dari 5 baris dan 312 kolom.
- * Kode Random Forest 13 :

```
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 3.20: Gambar ke-7

The screenshot shows a Jupyter Notebook cell with the code 'In [50]: imglabels.head()'. Below it, the output shows the first five rows of the DataFrame:

imgid	label
1	1
2	1
3	1
4	1
5	1

Figure 3.21: Gambar ke-8

- Penjelasan : Gambar di atas menunjukkan hasil dari variabel dflabel.head. Dimana berisikan data dari imgid dan label. Dan hasilnya dapat dilihat pada gambar di atas.
- * Kode Random Forest 14 :
 - Penjelasan : Gambar di atas merupakan pembagian dari data training dan dataset
- * Kode Random Forest 15 :
 - Penjelasan : Gambar di atas merupakan pemanggilan kelas RandomForestClassifier. max features yang diartikan berapa banyak kolom pada setiap tree.
- * Kode Random Forest 16 :
 - Penjelasan : Gambar di atas merupakan perintah untuk melakukan fit untuk membangun random forest yang sudah ditentukan dengan maksimum fitur sebanyak 50.
- * Kode Random Forest 17 :
 - Penjelasan : Gambar di atas menunjukkan hasil dari cetakan variabel dftrainatt.head.
- * Kode Random Forest 18 :
 - Penjelasan : Gambar di atas merupakan hasil dari variabel dftesttatt da dftsetlabel. Dimana hasilnya dapat dilihat dari pada gambar di atas

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

Figure 3.22: Gambar ke-9

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

Figure 3.23: Gambar ke-10

– Program Aplikasi Confusion Matrix dan Penjelasan Keluarannya :

* Kode Confusion Matrix 1 :

- Penjelasan : Gambar di atas merupakan kodingan untuk import confusiion matrik dari random forest. untuk hasilnya dapat dilihat dari gambar.

* Code Confusion Matrix 2 :

- Penjelasan : Gambar di atas merupakan tampilan dari variabel cm.

* Kode Confusion Matrix 3 :

- Penjelasan : Gambar di atas merupakan perintah untuk plot. Dan untuk hasilnya terpadat pada gambar di atas.

* Kode Confusion Matrix 4 :

- Penjelasan : Gambar di atas merupakan kodingan untuk menyesuaikan sumbu dengan nama datanya makanya datset nya dilakukan dengan perintah di atas.

* Kode Confusion Matrix 5 :

- Penjelasan : Gambar di atas merupakan perintah plot dari gambar sebelumnya.

– Program Klasifikasi SVM dan Decision Tree Beserta Penjelasan Keluarannya :

* Kode SVM :

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

* Kode Decision Tree :

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

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

Figure 3.24: Gambar ke-11

```
In [44]: df_att.head()
Out[44]:
   1    2    3    4    5    6    7    ...   306   307   308   309   310   311   312
imgid
10779  0    0    0    0    0    0    1    ...
9334   0    0    0    0    0    0    1    ...
10372  0    0    0    0    0    0    1    ...
1554   1    0    0    0    0    0    0    ...
378    0    0    0    0    0    0    1    ...
[5 rows x 312 columns]
```

Figure 3.25: Gambar ke-12

– Program Cross Validation dan Penjelasan Keluarannya :

* Code Cross Validation 1 :

- Penjelasan : Pada gambar di atas merupakan Hasil dari cross validation random forest.

* Code Cross Validation 2 :

- Penjelasan : Pada gambar di atas merupakan hasil dari cross validation Decission tree.

* Code Cross Validation 3 :

- Penjelasan : Pada gambar di atas merupakan hasil dari cross validation SVM.

– Program Pengamatan Komponen Informasi dan Penjelasan Keluarannya :

* Code Pengamatan Komponen Informasi 1 :

- Penjelasan : Pada gambar di atas menunjukkan cara untuk mengetahui berapa banyak tree yang dibuat, berapa banyak atribut yang dipakai dan informasi lainnya menggunakan kode.

* Code Pengamatan Komponen Informasi 2 :

- Penjelasan : Pada gambar di atas merupakan cara untuk melakukan plot informasi ini dengan kode di atas.

2. Penanganan Error

- Skrinsut Error

```
In [45]: df_label.head()
Out[45]:
   label
  imgid
10779    183
9334     159
10372    177
1554      28
378       8
```

Figure 3.26: Gambar ke-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 3.27: Gambar ke-14

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

|||||| HEAD

3.5 Puad Hamdani / 1164084

3.5.1 Praktek

Penyelesaian Tugas minggu ke 3

1. Program Aplikasi Random Forest dan Penjelasan Keluarannya :

- Code 1 :
 - Penjelasan : Membaca dataset. Codingan di atas menghasilkan variabel baru yaitu imgatt. Terdapat 3 kolom dan 3677856 baris data.
- Code 2 :
 - Penjelasan : Codingan di atas berfungsi untuk melihat sebagian data awal dari dataset. Hasilnya terdapat pada gambar di atas setelah di eksekusi.
- Code 3 :

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

Figure 3.28: Gambar ke-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 3.29: Gambar ke-16

- 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.
- Code 4 :
 - Penjelasan : Pada gambar di atas menampilkan hasil dari variabel imgatt2. Dimana index nya 'imgid', kolom berisi 'attid' dan values atau nilainya berisi 'present'.
- Code 5 :
 - Penjelasan : Pada gambar di atas menampilkan hasil dari variabel imgatt2.head. Dimana dataset nya ada 5 baris dan 312 kolom.
- Code 6 :
 - Penjelasan : Pada gambar di atas menampilkan jumlah dari baris dan kolom dari variabel imgatt2. Dimana 11788 adalah baris dan 312 adalah kolom.
- Code 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.
- Code 8 :

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

Figure 3.30: Gambar ke-17

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

Figure 3.31: Gambar ke-18

- 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.

- Code 9 :

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

- Code 10 :

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

- Code 11 :

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

- Code 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.

- Code 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.

```
In [52]: from sklearn.metrics import confusion_matrix
...: pred_labels = clf.predict(df_test_att)
...: cm = confusion_matrix(df_test_label, pred_labels)
```

Figure 3.32: Gambar ke-19

```
In [53]: cm
Out[53]:
array([[ 3,  0,  1, ...,  0,  0,  0],
       [ 1, 11,  0, ...,  0,  0,  0],
       [ 2,  0,  5, ...,  0,  0,  0],
       ...,
       [ 0,  0,  0, ...,  5,  0,  0],
       [ 0,  0,  0, ...,  0,  9,  0],
       [ 0,  0,  0, ...,  0,  1, 19]], dtype=int64)
```

Figure 3.33: Gambar ke-20

- Code 14 :
 - Penjelasan : Pada gambar di atas merupakan pembagian dari data training dan dataset
- Code 15 :
 - Penjelasan : Pada gambar di atas merupakan pemanggilan kelas RandomForestClassifier. max features yang diartikan berapa banyak kolom pada setiap tree.
- Code 16 :
 - Penjelasan : Pada gambar di atas merupakan perintah untuk melakukan fit untuk membangun random forest yang sudah ditentukan dengan maksimum fitur sebanyak 50.
- Code 17 :
 - Penjelasan : Pada gambar di atas menunjukkan hasil dari cetakan variabel dftrainatt.head.
- Code 18 :
 - Penjelasan : Pada gambar di atas merupakan hasil dari variabel dftestatt da dftsetlabel. Dimana hasilnya dapat dilihat dari pada gambar di atas

2. Program Aplikasi Confusion Matrix dan Penjelasan Keluarannya :

- Code 1 :

```

In [54]: 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 3.34: Gambar ke-21

- Penjelasan : Pada gambar di atas merupakan kodingan untuk import confusision matrik dari random forest. untuk hasilnya dapat dilihat dari gambar.
- Code 2 :
 - Penjelasan : Pada gambar di atas merupakan tampilan dari variabel cm.
- Code 3 :
 - Penjelasan : Pada gambar di atas merupakan perintah untuk plot. Dan untuk hasilnya terpadat pada gambar di atas.
- Code 4 :
 - Penjelasan : Pada gambar di atas merupakan kodingan untuk menye-suaikan sumbu dengan nama datanya makanya datset nya di lakukan dengan perintah di atas.
- Code 5 :
 - Penjelasan : Pada gambar di atas merupakan perintah plot dari gam-bar sebelumnya.

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

- Code SVM :

```

In [56]: birds = pd.read_csv("CUB_200_2011/classes.txt",
...:                         sep='\t+', header=None, usecols=[1], names=['birdname'])
...: birds
...: birds
Out[56]:
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
19         020.Yellow_breasted_Chat
20         021.Eastern_Towhee
21         022.Chuck_will_Widow
22         023.Brandt_Cormorant

```

Figure 3.35: Gambar ke-22

```

In [37]: 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
[[0.27 0. 0.18 ... 0. 0. 0.]
 [0. 0.73 0. ... 0. 0. 0.]
 [0.08 0. 0.42 ... 0. 0. 0.]
 ...
 [0. 0. 0. ... 0.2 0. 0.]
 [0. 0. 0. ... 0. 0.3 0.]
 [0. 0. 0. ... 0. 0. 0.88]]

```

Figure 3.36: Gambar ke-23

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

4. Program Cross Validation dan Penjelasan Keluarannya :

- Code 1 :
 - Penjelasan : Pada gambar di atas merupakan Hasil dari cross validation random forest.
- Code 2 :

```

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 3.37: Gambar 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 3.38: Decission Tree

- Penjelasan : Pada gambar di atas merupakan hasil dari cross validation Decission tree.
- Code 3 :
 - Penjelasan : Pada gambar di atas merupakan hasil dari cross validation SVM.

5. Program Pengamatan Komponen Informasi dan Penjelasan Keluarannya :

- Code 1 :
- Penjelasan : Pada gambar di atas menunjukkan cara untuk mengetahui berapa banyak tree yang dibuat, berapa banyak atribut yang dipakai dan informasi lainnya menggunakan kode.
- Code 2 :
- Penjelasan : Pada gambar di atas merupakan cara untuk melakukan plot informasi ini dengan kode di atas.

6. 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 [46]: from sklearn.model_selection import cross_val_score
...: scores = cross_val_score(clf, df_train_att, df_train_label, cv=5)
...: # show average score and +/- two standard deviations away (covering 95% of
...: scores)
...: print("Accuracy: %0.2f (+/- %0.2f)" % (scores.mean(), scores.std() * 2))
Accuracy: 0.44 (+/- 0.02)
```

Figure 3.39: Cross Validation 1

```
In [47]: scorestree = cross_val_score(clftree, df_train_att, df_train_label, cv=5)
...: print("Accuracy: %0.2f (+/- %0.2f)" % (scorestree.mean(), scorestree.std()
...: * 2))
Accuracy: 0.26 (+/- 0.03)
```

Figure 3.40: Cross Validation 2

3.6 Mhd Zulfikar Akram Nasution / 1164081

3.6.1 Teori

1. Random Forest

Ramdom Forest adalah hutan yang acak, dimana maksudnya yaitu terdapat banyak pohon-pohon yang mana disetiap pohon tersebut memiliki atribut yang berbeda-beda, random forest disebut juga kumpulan pohon-pohon keputusan. contoh random forest seperti gambar 3.1

2. Cara membaca dataset kasus
3. Buka aplikasi spyder untuk membuka dan membaca kodingan dataset
4. Kemudian buat variable imgatt untuk memasukkan atribut label
5. Lalu uji coba kodingan untuk mengetahui apa hasil dari dataset tersebut
6. imgatt.head() untuk melihat sebagian data awal
7. .shape untuk melihat jumlah data
8. .pivot untuk merubah atribut menjadi kolom

Dengan menguji coba kodingan yang ada pada spyder kita akan dapat membaca dataset yang ada.

9. Cross Validation

Cross Validation adalah sebuah metode statistik yang digunakan untuk mengevaluasi kinerja model, dimana data dipisah menjadi dua subset yaitu data proses pembelajaran dan data evaluasi atau validasi.

```

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)
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)
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)
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)
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)
Accuracy: 0.27 (+/- 0.02)

```

Figure 3.41: Cross Validation 3

10. Arti 44 persen pada RF, 27 persen pada Decission Tree, dan 29 persen pada SVM.
11. 44 persen pada Random Forest adalah menunjukkan hasil yang sempurna pada keputusan yang diambil, biasanya hasil keputusan yang dicapai sekitar 42-44 persen.
12. 27 persen pada Decission Tree adalah menunjukkan hasil keputusan pada tiap-tiap tree dari dataset yang ada.
13. 29 persen pada SVM menunjukkan hasil keputusan dengan klasifikasi dari dataset yang ada.
14. Confusion Matrix
15. Pertama import confusion matrixnya
16. Kemudian Plot confusion matrix
17. Lalu sesuaikan plotnya dengan nama data yang ada
18. Setelah itu plot kembali
- Contoh hasil connfusion matrix pada gambar 3.2
19. Voting

```

In [49]: 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: 0.26 (+/- 0.02)
Max features: 5, num estimators: 30, accuracy: 0.35 (+/- 0.03)
Max features: 5, num estimators: 50, accuracy: 0.39 (+/- 0.03)
Max features: 5, num estimators: 70, accuracy: 0.41 (+/- 0.02)
Max features: 5, num estimators: 90, accuracy: 0.42 (+/- 0.03)
Max features: 5, num estimators: 110, accuracy: 0.43 (+/- 0.03)
Max features: 5, num estimators: 130, accuracy: 0.43 (+/- 0.03)
Max features: 5, num estimators: 150, accuracy: 0.44 (+/- 0.03)

```

Figure 3.42: Program Pengamatan Komponen Informasi 1

Voting adalah hasil akhir dari keputusan yang ada pada setiap pohon di random forest, maksudnya ialah setiap keputusan yang telah dikumpulkan maka akan di voting bahwa hasil tersebut adalah hasil yang benar. Misalnya kita dapat lihat pada gambar 3.3 yaitu dari beberapa ciri-ciri yang ada dapat di voting atau disimpulkan hasil yang paling banyak dimiliki oleh burung belibis, sehingga dapat disimpulkan bahwa dari ciri- tersebut ialah merupakan ciri-ciri dari burung belibis.

3.6.2 Praktek

1. Aplikasi sederhana menggunakan Pandas seperti pada gambar 3.14

Penjelasan kodingan :

- Memanggil library.
- Membuat variable dengan data frame.
- Menampilkan hasil

Sehingga menghasilkan gambar 3.15 :

2. Aplikasi sederhana menggunakan Numpy pada gambar 3.16

Penjelasan kodingan :

- Memanggil library

```
In [9]: 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()
```

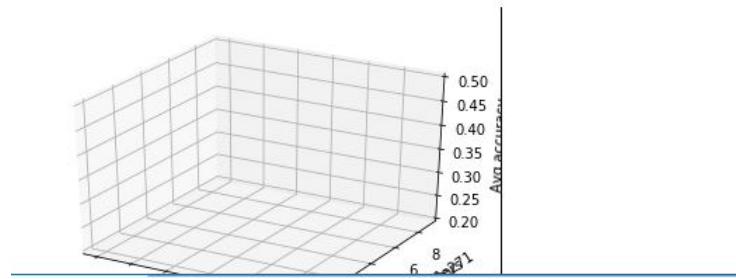


Figure 3.43: Program Pengamatan Komponen Informasi 2

- Membuat variable dengan value random dan size 3
- Menampilkan hasil value

Sehingga menghasilkan gambar 3.17:

3. Aplikasi sederhana menggunakan Matplotlib pada gambar 3.18

Penjelasan kodingan :

- Memanggil library
- Membuat variable yang berisi bahasa pemrograman
- Membuat variable yang berisi popularitas
- Membuat variable untuk explode
- Membuat diagram pie atau yang berbentuk lingkaran
- Membuat garis koordinat
- Menampilkan hasil

Sehingga menghasilkan gambar 3.19:

4. Program klasifikasi random forest

- Pertama baca dataset terlebih dahulu seperti pada gambar 3.20

```

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

```

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

```

Figure 3.45: Gambar1

- Kemudian lihat sebagian data awal dengan listing seperti pada gambar 3.21
- Kemudian lihat jumlah datal dengan listing seperti pada gambar 3.22
- Ubah atribut menjadi kolom seperti pada gambar 3.23
- Membaca dataset label seperti pada gambar 3.24
- Menggabungkan field dari dua file terpisah seperti pada gambar 3.25
- Memidahkan dan memilih label seperti pada gambar 3.26
- Melihat isi masing-masing data frame seperti pada gambar 3.27
- Pembagian data training dan data tes seperti pada gambar 3.28

```
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 3.46: Gambar2

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

Figure 3.47: Gambar3

- Memanggil kelas random forest seperti pada gambar 3.29
- Lakukan fit untuk emmbangun random forest seperti pada gambar 3.30
- Kemudian lihat hasil dengan predict seperti pada gambar 3.31
- Lalu akan terlihat hasil score dari klasifikasi seperti pada gambar 3.32

5. Program Klasifikasi Confusion Matrix

- Setelah melakukan random forest kemudian dipetakan ke dalam confusion matrix seperti pada gambar 3.33
- Kemudian lihat hasilnya seperti pada gambar 3.34
- Lalu lakukan perintah plot seperti pada gambar 3.35
- Selanjutnya nama data akan di set agar plot sumbunya sesuai seperti pada gambar 3.36
- Setelah label berubah, maka dilakukan perintah plot seperti pada gambar 3.37

6. Program Klasifikasi SVM dan Decision Tree

- Program Decision Tree seperti pada gambar 3.38
Mengklasifikasikan dataset yang sama menggunakan decision tree.
- Program Klasifikasi SVM seperti pada gambar 3.39
Mengklasifikasikan dataset yang sama menggunakan SVM.

7. Program Cross Validation

- Lakukan pengecekan cross validation untuk random forest seperti pada gambar 3.40

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

Figure 3.48: Gambar 4

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

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 3.49: Gambar 5

- Lakukan pengecekan cross validation untuk decision tree seperti pada gambar 3.41
- Lakukan pengecekan cross validation untuk SVM seperti 2 pada gambar 3.4

8. Program Pengamatan Komponen Informasi

- Melakukan pengamatan komponen informasi untuk mengetahui berapa banyak tree yang dibuat, atribut yang dipakai, dan informasi lainnya seperti pada gambar 3.43
- Melakukan plot informasi agar bisa dibaca seperti pada gambar 3.44

lliilli c781ac4472a25ed2af92e92a48bac83484ece61c

3.7 Method 1

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

3.8 Method 2

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

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

Figure 3.50: 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 3.51: Gambar 7

```
In [39]: imglabels.head()  
Out[39]:  
imgid  
1  
2  
3  
4  
5
```

Figure 3.52: Gambar 8

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

Figure 3.53: Gambar 9

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

Figure 3.54: Gambar 10

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

Figure 3.55: Gambar 11

```
In [44]: df_att.head()  
Out[44]:  
1 2 3 4 5 6 7 ... 306 307 308 309 310 311 312  
imgid  
10779 0 0 0 0 0 0 1 ... 1 0 0 0 0 0 0 1  
9334 0 0 0 0 0 0 1 ... 1 0 0 0 0 0 1 0  
10372 0 0 0 0 0 0 1 ... 0 0 0 1 0 0 0 0  
1554 1 0 0 0 0 0 0 ... 1 0 0 0 0 1 0 0  
378 0 0 0 0 0 0 1 ... 0 0 0 0 1 0 0 0  
[5 rows x 312 columns]
```

Figure 3.56: Gambar 12

```
In [45]: df_label.head()
Out[45]:
   label
  imgid
0 10779    183
1 9334     159
2 10372     177
3 1554      28
4 378       8
```

Figure 3.57: 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 3.58: Gambar 14

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

Figure 3.59: 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 3.60: Gambar 16

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

Figure 3.61: Gambar 17

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

Figure 3.62: Gambar 18

```
In [52]: from sklearn.metrics import confusion_matrix
...: pred_labels = clf.predict(df_test_att)
...: cm = confusion_matrix(df_test_label, pred_labels)
```

Figure 3.63: Gambar 19

```
In [53]: cm
Out[53]:
array([[ 3,  0,  1, ...,  0,  0,  0],
       [ 1, 11,  0, ...,  0,  0,  0],
       [ 2,  0,  5, ...,  0,  0,  0],
       ...,
       [ 0,  0,  0, ...,  5,  0,  0],
       [ 0,  0,  0, ...,  0,  9,  0],
       [ 0,  0,  0, ...,  0,  1, 19]], dtype=int64)
```

Figure 3.64: Gambar 20

```
In [54]: 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 3.65: Gambar 21

```
In [56]: birds = pd.read_csv("CUB_200_2011/classes.txt",
...:                         sep='\s+', header=None, usecols=[1], names=['birdname'])
...: birds = birds[['birdname']]
...: birds
Out[56]:
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
19     020.Yellow_breasted_Chat
20     021.Eastern_Towhee
21     022.Chuck_will_Widow
22     023.Brandt_Cormorant
```

Figure 3.66: Gambar 22

```
In [37]: 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
[[0.27 0. 0.18 ... 0. 0. 0. ]
 [0. 0.73 0. ... 0. 0. 0. ]
 [0.08 0. 0.42 ... 0. 0. 0. ]
 ...
 [0. 0. 0. ... 0.2 0. 0. ]
 [0. 0. 0. ... 0. 0.3 0. ]
 [0. 0. 0. ... 0. 0. 0.88]]
```

Figure 3.67: Gambar 23

```
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 3.68: 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 3.69: Decission Tree

```
In [46]: from sklearn.model_selection import cross_val_score
...: scores = cross_val_score(clf, df_train_att, df_train_label, cv=5)
...: # show average score and +/- two standard deviations away (covering 95% of
scores)
...: print("Accuracy: %0.2f (+/- %0.2f)" % (scores.mean(), scores.std() * 2))
Accuracy: 0.44 (+/- 0.02)
```

Figure 3.70: Cross Validation 1

```
In [47]: scorestree = cross_val_score(clftree, df_train_att, df_train_label, cv=5)
...: print("Accuracy: %0.2f (+/- %0.2f)" % (scorestree.mean(), scorestree.std()
* 2))
Accuracy: 0.26 (+/- 0.03)
```

Figure 3.71: Cross Validation 2

```

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)
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)
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)
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)
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)
Accuracy: 0.27 (+/- 0.02)

```

Figure 3.72: Cross Validation 3

```

In [49]: 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: 0.26 (+/- 0.02)
Max features: 5, num estimators: 30, accuracy: 0.35 (+/- 0.03)
Max features: 5, num estimators: 50, accuracy: 0.39 (+/- 0.03)
Max features: 5, num estimators: 70, accuracy: 0.41 (+/- 0.02)
Max features: 5, num estimators: 90, accuracy: 0.42 (+/- 0.03)
Max features: 5, num estimators: 110, accuracy: 0.43 (+/- 0.03)
Max features: 5, num estimators: 130, accuracy: 0.43 (+/- 0.03)
Max features: 5, num estimators: 150, accuracy: 0.44 (+/- 0.03)

```

Figure 3.73: Program Pengamatan Komponen Informasi 1

```
In [9]: 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()
```

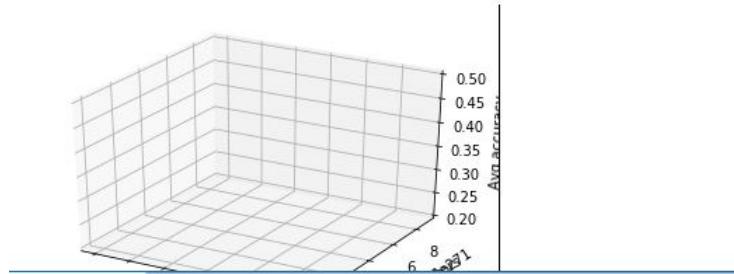


Figure 3.74: Program Pengamatan Komponen Informasi 2

```
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 3.75: Error

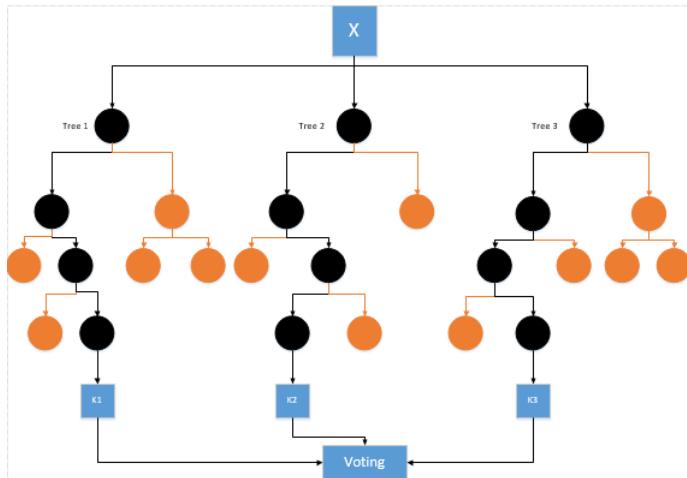


Figure 3.76: Random Forest

Ciri-ciri Burung	Merpati	Garuda	Belibis	Perkutut
1	1	0	0	0
2	0	0	1	0
3	1	0	0	1
4	0	1	0	0
5	0	0	0	1
6	0	0	1	0
7	0	0	1	1
8	1	0	1	0

Figure 3.77: Confusion Matrix

Ciri-ciri Burung	Merpati	Garuda	Belibis	Perkutut
1	1	0	0	0
2	0	0	1	0
3	1	0	0	1
4	0	1	0	0
5	0	0	0	1
6	0	0	1	0
7	0	0	1	1
8	1	0	1	0

Figure 3.78: Voting

```

In [14]: import pandas as musik
....: mb = {'Nama Alat' : ['Snare', 'Quint Tom', 'Bass', 'Cymbal', 'Trumpet'],
....:       'Type Alat' : ['Percussion Line', 'Percussion Line', 'Percussion
Line', 'Percussion Line', 'Bruss Line']}
....: dh = musik.DataFrame(mb)
....: print(dh)
  
```

Figure 3.79: Aplikasi pandas

Index	Nama Alat	Type Alat
0	Snare	Percussion Line
1	Quint Tom	Percussion Line
2	Bass	Percussion Line
3	Cymbal	Percussion Line
4	Trumpet	Bruss Line

Figure 3.80: Hasil Pandas

```
import numpy as a
x=a.random.normal(size=3)
print(x)
```

Figure 3.81: Aplikasi Numpy

```
import numpy as a
x=a.random.normal(size=3)
print(x)
```

Figure 3.82: Hasil Numpy

```
import matplotlib.pyplot as plt

# Pie chart, where the slices will be ordered and plotted counter-clockwise:
labels = 'Futsal', 'Badminton', 'Renang', 'Lari'
sizes = [30, 15, 35, 20]
explode = (0, 0.1, 0, 0) # only "explode" the 2nd slice (i.e. 'Hogs')

fig1, ax1 = plt.subplots()
ax1.pie(sizes, explode=explode, labels=labels, autopct='%1.1f%%',
         shadow=True, startangle=90)
ax1.axis('equal') # Equal aspect ratio ensures that pie is drawn as a circle.

plt.show()
```

Figure 3.83: Aplikasi Matplotlib

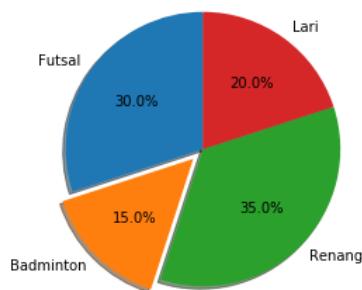


Figure 3.84: Hasil Matplotlib

```

import pandas as pd

# some lines have too many fields (?), so skip bad lines
imgatt = pd.read_csv("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 3.85: Membaca Data File

```
imgatt.head()
```

Figure 3.86: Melihat sebagian data

```
imgatt.shape
```

Figure 3.87: Melihat jumlah data

```
imgatt2 = imgatt.pivot(index='imgid', columns='attid', values='present')
```

Figure 3.88: Ubah atribut jadi kolom

```

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
# in the file image_class_labels.txt, with each line corresponding to one

```

Figure 3.89: Membaca dataset label

```

df = imgatt2.join(imglabels)
df = df.sample(frac=1)

```

Figure 3.90: Menggabungkan field

```

df_att = df.iloc[:, :312]
df_label = df.iloc[:, 312:]

```

Figure 3.91: Memisahkan dan memilih label

```
df_att.head()
```

```
# In[56]:
```

```
df_label.head()
```

Figure 3.92: Melihat isi data

```
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 3.93: Pembagian data

```
from sklearn.ensemble import RandomForestClassifier
clf = RandomForestClassifier(max_features=50, random_state=0, n_estimators=100)
```

Figure 3.94: Kelas random forest

```
In [46]: clf.fit(df_train_att, df_train_label)
Out[46]:
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 3.95: Membuat fitting

```
In [47]: print(clf.predict(df_train_att.head()))
[ 2 178 128  70 170]
```

Figure 3.96: Melihat hasil

```
In [48]: clf.score(df_test_att, df_test_label)
Out[48]: 0.44350580781414994
```

Figure 3.97: Hasil score

```
In [49]: from sklearn.metrics import confusion_matrix
...: pred_labels = clf.predict(df_test_att)
...: cm = confusion_matrix(df_test_label, pred_labels)
```

Figure 3.98: Memetakan ke confusion matrix

```
In [50]: cm
Out[50]:
array([[ 8,  3,  3, ...,  0,  0,  0],
       [ 0, 12,  0, ...,  0,  0,  0],
       [ 4,  1,  5, ...,  0,  1,  0],
       ...,
       [ 0,  0,  0, ...,  4,  0,  0],
       [ 0,  0,  0, ...,  0, 11,  0],
       [ 0,  0,  0, ...,  0,  0, 11]], dtype=int64)
```

Figure 3.99: Melihat hasil

```
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])):
        plt.text(j, i, format(cm[i, j], fmt),
                 horizontalalignment="center",
                 color="white" if cm[i, j] > thresh else "black")

    plt.tight_layout()
    plt.ylabel('True label')
    plt.xlabel('Predicted label')
```

Figure 3.100: Melakukan Plot

```
In [53]: birds = pd.read_csv("classes.txt",
...:                         sep='\t+', header=None, usecols=[1],
names=['birdname'])
...: birds = birds['birdname']
...: birds
Out[53]:
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
```

Figure 3.101: Plotting nama data

```
In [54]: 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
[[0.38 0.14 0.14 ... 0. 0. 0.]
 [0. 0.8 0. ... 0. 0. 0.]
 [0.24 0.06 0.29 ... 0. 0.06 0.]
 ...
 [0. 0. 0. ... 0.24 0. 0.]
 [0. 0. 0. ... 0. 0.61 0.]
 [0. 0. 0. ... 0. 0. 0.73]]
```

Figure 3.102: Melakukan perintah plot

```
In [55]: from sklearn import tree
...: clftree = tree.DecisionTreeClassifier()
...: clftree.fit(df_train_att, df_train_label)
...: clftree.score(df_test_att, df_test_label)
Out[55]: 0.2608236536430834
```

Figure 3.103: Klasifikasi menggunakan decision tree

```
In [56]: from sklearn import svm
...: clfsvm = svm.SVC()
...: clfsvm.fit(df_train_att, df_train_label)
...: clfsvm.score(df_test_att, df_test_label)
C:\Users\user\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[56]: 0.2914466737064414
```

Figure 3.104: Klasifikasi menggunakan SVM

```
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 3.105: 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 3.106: Pengecekan cross validation decision tree

```

...:     ...: print("Accuracy: %0.2f (+/- %0.2f)" % (scoressvm.mean(), scoressvm.std()
...: * 2))
C:\Users\lsvapr\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)
C:\Users\lsvapr\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)
C:\Users\lsvapr\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)
C:\Users\lsvapr\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)
C:\Users\lsvapr\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)
C:\Users\lsvapr\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)
Accuracy: 0.26 (+/- 0.02)

```

Figure 3.107: Pengecekan cross validation SVM

```

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 3.108: 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()
```

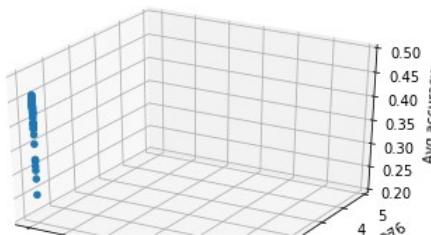


Figure 3.109: Plot informasi

Chapter 4

Klasifikasi Teks

4.1 Jesron Marudut Hatuan/1164077

4.1.1 Teori

1. Pengertian Klasifikasi teks

Klasifikasi merupakan sebuah kata serapan yang berasal dari bahasa Belanda, yaitu classificatie, yang sendirinya berasal dari bahasa Prancis classification. Istilah ini menunjuk pada metode untuk menyusun data secara sistematis atau menurut beberapa aturan atau kaidah yang telah ditetapkan. Di dalam kamus besar bahasa Indonesia, klasifikasi adalah penyusunan bersistem dalam kelompok atau golongan menurut kaidah atau standar yang ditetapkan. Secara harafiah bisa pula dikatakan bahwa klasifikasi adalah pembagian sesuatu menurut kelas-kelas. Menurut Ilmu Pengetahuan, Klasifikasi adalah Proses pengelompokan benda berdasarkan ciri-ciri dari persamaan dan perbedaan.

No.	Tier	Keterangan
1	Bronze	Rank pertama sekaligus terendah dalam PUBG
2	Silver	Rank ke-2
3	Gold	Rank ke-3
4	Platinum	Rank ke-4
5	Diamond	Rank ke-5
6	Crown	Rank ke-6
7	Ace	Rank ke-7
8	Conqueror	Rank ke-8 sekaligus tertinggi dalam Pubg

Figure 4.1: Klasifikasi teks

2. Alasan Klasifikasi Bunga tidak bisa menggunakan machine learning

Machine Learning tidak dapat mengklasifikasikan bunga, dikarena data yang diberikan pada mesin itu akan di algoritmakan untuk mencari sesuatu yang

menarik dalam data yang kita berikan, hingga akhirnya sistem AI akan membangun pengetahuan berdasarkan data tersebut. Dengan maksud pembelajaran mesin data beradaptasi terhadap suatu masalah dengan mempelajari pola-pola yang ditemukan dalam data. Sebagai contoh data pada spesies bunga dari genus Iris dengan melihat ukuran sepal (kelopak) dan mahkota pada algoritma data bunga tersebut akan melatih proses pembelajaran pada mesin dalam menganalisa spesies bunga Iris. Dan algoritma pembelajaran mesin akan mempelajari karakteristik dari masing-masing spesies bunga Iris berdasarkan ukuran sepal dan petal yang diberikan.

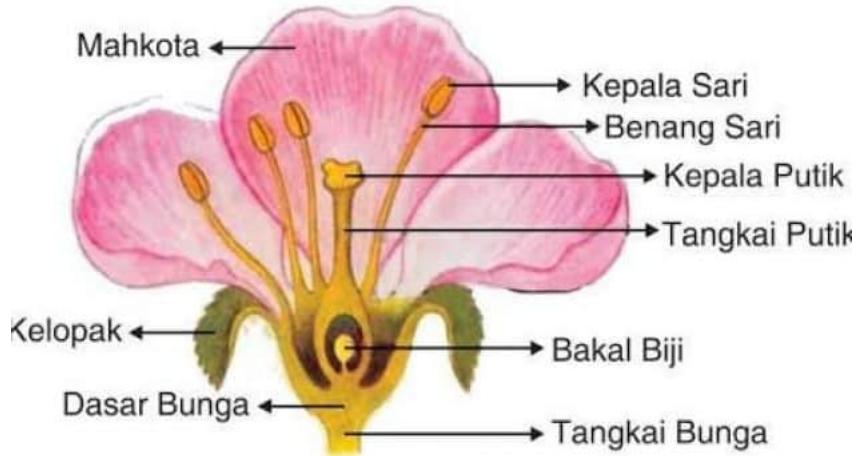


Figure 4.2: Klasifikasi bunga

3. Youtube memungkinkan agar mendapatkan video yang direkomendasikan, karena Machine Learning pada Youtube pasti akan melibatkan data yang sering di lihat oleh penggunanya. Youtube juga akan memberitahukan si pengguna apabila ada video baru yang telah di upload pada chanel yang direkomendasikan untuk si pengguna. Dan apabila menonton video pada youtube maka youtube dapat mengingat dan menggunakan kata tersebut sebagai referensi.

4. Vectorisasi Data

- proses vektorisasi ini menghasilkan suatu wujud peta yang menggambarkan keadaan permukaan bumi atau bentang alam. Sifat data yang geometris menunjukkan ukuran dimensi yang sesungguhnya.

5. Bag of word

Bag of word merupakan konsep yang diambil dari analisis, kemudian merepresentasikan dokumen berupa kumpulan informasi penting tanpa mengurutkan setiap katanya.

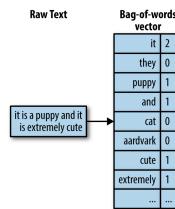


Figure 4.3: Bag of Word

6. TF-IDF

TF-IDF dimaksudkan untuk mencerminkan seberapa relevan suatu istilah dalam dokumen yang diberikan. Intuisi di baliknya adalah bahwa jika sebuah kata muncul beberapa kali dalam sebuah dokumen, kita harus meningkatkan relevansinya karena itu harus lebih bermakna daripada kata-kata lain yang muncul lebih sedikit kali (TF). Pada saat yang sama, jika sebuah kata muncul berkali-kali dalam suatu dokumen tetapi juga di sepanjang banyak dokumen lain, mungkin itu karena kata ini hanya kata yang sering; bukan karena itu relevan atau bermakna (IDF).

4.2 Jesron Marudut Hatuan / 1164077

4.2.1 Praktek

1. Kali ini saya akan membuat data dummy dimana saya menggunakan dengan format csv, dan data-datanya saya ambil dari sebuah web yang bernama UCI Machine Learning

Repository dengan nama file Youtube03-LMFAO.csv

Penjelasan pada gambar 1 yaitu mengimport library pandas dimana kita mengalaskan praktek sebagai pandas. Pandas berguna untuk mengelola dataframe = matrix = tabel kemudian untuk memanggil nama alias dan membaca format csvnya.

2. Dari dataframe yang sudah ada sebelumnya kita akan membagi menjadi 2 yaitu 450 row pertama dan 50 row

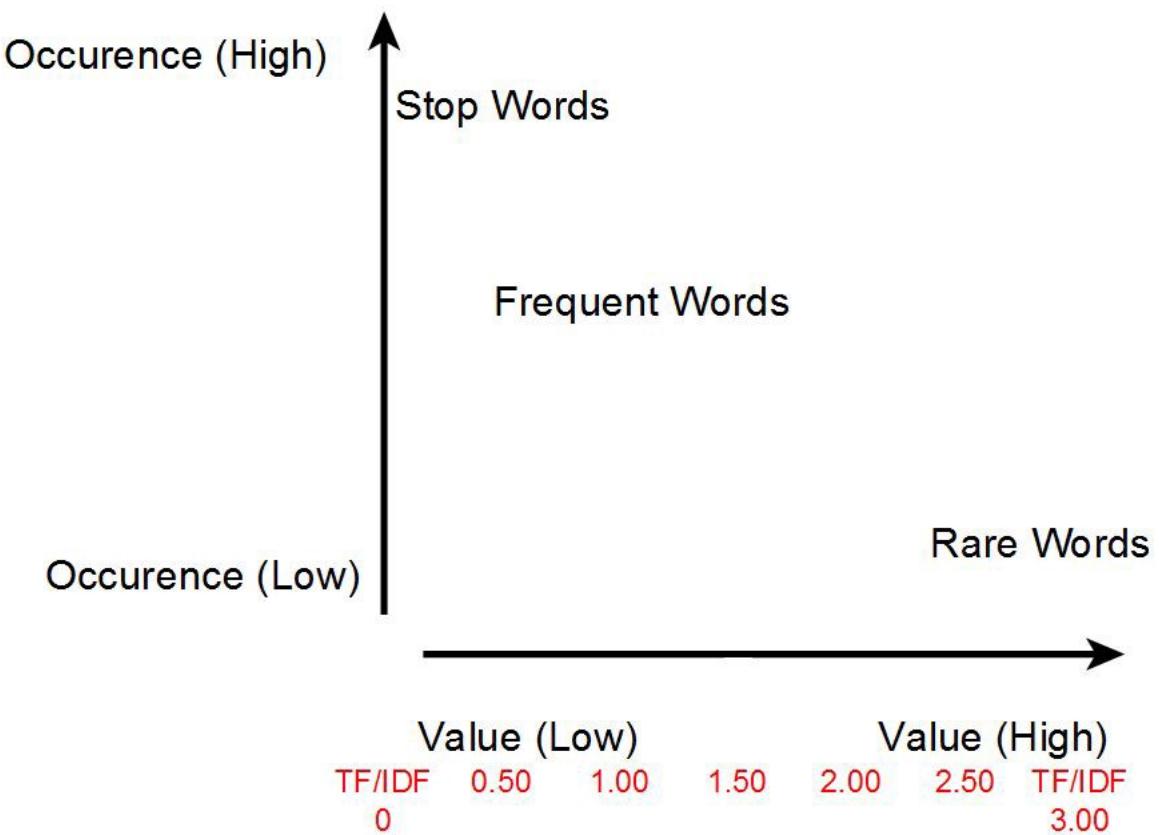


Figure 4.4: TF-IDF

```
# In[1]: melakukan import pandas dan membaca file csv
import pandas as pd
andi=pd.read_csv('D:\Tugas Kuliah\Semester 6\KECERDASAN BUATAN\Python-Artificial-Intelligence-Projects-for-Beginners\Chapter03\spam.csv')
```

Figure 4.5: Data Dummy 500 Data

Dapat dilihat pada gambar 5.5

Penjelasan pada gambar 2 yaitu baris pertama dimana `d_train` untuk membagi data training sebanyak 450 row dan pada

baris kedua dimana `d_test` untuk data sisa atau data yang baru sebanyak 50 row.

3. Melakukan Vektorisasi dan Klasifikasi Data pada data LMFAO pada gambar 3 merupakan dataframe keseluruhan dari file csv yang sudah dimasukkan dengan jumlah 448 baris dan 5 kolom. Nospam merupakan dataframe yang isinya hanya data yang bukan termasuk spam dengan inisial angka 0. Sedangkan spam merupakan dataframe yang isinya hanya data spam dengan inisial angka 1.

The screenshot shows a Jupyter Notebook interface. At the top, there are three dataframes: nospam, spam, and yeay, each with 175 rows and 5 columns. Below them is a navigation bar with 'Variable explorer', 'File explorer', and 'Help'. Underneath is an 'IPython console' tab, which is active. A 'Console 1/A' tab is open, showing the following code:

```
In [118]: yeay = andi.sample(frac=1)
In [119]: andi_train=yeay[:300]
...: andi_test=yeay[300:]
In [120]:
```

Figure 4.6: Membagi 2 Dataframe

nospam	DataFrame	(175, 5)	Column names: COMMENT_ID, AUTHOR, DATE, CONTENT, CLASS
spam	DataFrame	(175, 5)	Column names: COMMENT_ID, AUTHOR, DATE, CONTENT, CLASS
yeay	DataFrame	(350, 5)	Column names: COMMENT_ID, AUTHOR, DATE, CONTENT, CLASS

Figure 4.7: Vektorisasi dan Klasifikasi Data

Pada gambar 3 merupakan hasil output content dimana terdapat 448 baris/-data yang mempunyai 1602 kata-kata yang digunakan pada komentar yang ada di content tersebut.

```
In [85]: dvec
Out[85]:
<350x1738 sparse matrix of type '<class 'numpy.int64'>'>
with 5275 stored elements in Compressed Sparse Row format>
```

Figure 4.8: Data Content

Pada gambar 4 maksud dari outputannya merupakan dataframe kata-kata tadi yang berjumlah 1602 kata orang yang komen pada data LMFAO.

4. Mengklasifikasikan dari data vektorisasi dengan menggunakan klasifikasi svm(support vectorisasi machine) dapat dilihat pada gambar 6.

Jadi pada gambar 6 merupakan hasil dari memprediksi data score vektorisasi dengan svm menggunakan metode fit dimana digunakan untuk data

dk	list	1738	['00', '000', '002', '018', '04', '053012', '0cb8qfjaa', '0d878a889c',...]
----	------	------	--

Figure 4.9: DataFrame Kata-kata Pada Content

```
In [141]: from sklearn import svm
....: clfsvm = svm.SVC()
....: clfsvm.fit(andi_train_att, andi_train_label)
....: clfsvm.score(andi_test_att, andi_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[141]: 0.4
```

Figure 4.10: Klasifikasi SVM Dari Data Vektorisasi

training atau data pelatihannya.

5. Mengklasifikasikan dari data vektorisasi dengan menggunakan klasifikasi Decision Tree dapat dilihat pada gambar 7.

```
In [140]: from sklearn import tree
....: clftree = tree.DecisionTreeClassifier()
....: clftree.fit(andi_train_att, andi_train_label)
....: clftree.score(andi_test_att, andi_test_label)
Out[140]: 0.96
```

Figure 4.11: Klasifikasi Decision Tree Dari Data Vektorisasi

Jadi pada gambar 7 merupakan hasil dari memprediksi data score vektorisasi dengan Decision Tree menggunakan metode fit dimana digunakan untuk data training atau data pelatihannya saja.

6. Mengeplot confusion matrix menggunakan matplotlib dapat dilihat pada gambar 8.

Pada gambar 8 sebelumnya kita harus mengimport matplotlibnya terlebih dahulu setelah itu di sini saya menggunakan numpy untuk mengeluarkan hasil plot confusion matrix pada matplotlibnya nantinya akan keluar normalisasi dari confusion matrix berupa data baris dan kolom.

```
In [144]: import numpy as np
....: np.set_printoptions(precision=2)
....: plot_confusion_matrix(cm, classes=andi, normalize=True)
....: plt.show()
Normalized confusion matrix
[[1.  0. ]
 [0.05 0.95]]
```

Figure 4.12: Plot Confusion Matrix Menggunakan Matplotlib

7. Menjalankan program cross validation pada data vektorisasi dapat dilihat pada gambar 9.

```
In [145]: from sklearn.model_selection import cross_val_score
....: scores = cross_val_score(clf, andi_train_att, andi_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: 0.93 (+/- 0.03)
```

Figure 4.13: Program Cross Validation Pada Data Vektorisasi

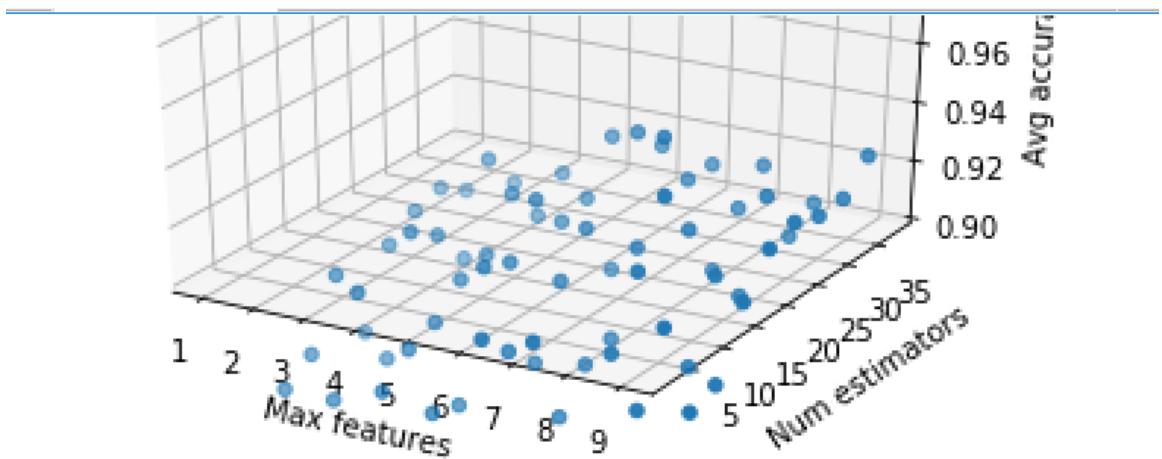
Pada gambar 9 yang pertama yaitu memunculkan akurasi cross validation dari random forest pada data yang sudah divektorisasi, yang kedua yaitu memunculkan akurasi cross validation dari decision tree pada data yang sudah divektorisasi, dan yang ketiga yaitu memunculkan akurasi cross validation dari svm pada data yang sudah divektorisasi.

8. Membuat program pengamatan komponen informasi pada data yang sudah divektorisasi dapat dilihat pada gambar 10

Pada gambar 10 merupakan hasil outputan yang mana max features di sana terdapat 9 data dari 10 yang sudah kita masukkan ke dalam codingan sebelumnya dan penomoran estimatornya merupakan data per 5 dari angka 40 sedangkan rata-rata akurasinya kita tuliskan datanya mulai dari 0.9 sampai dengan 1. Titik-titik yang didalam tersebut merupakan data vektorisasi dari pengamatan komponen informasinya.

4.2.2 Penanganan Eror

1. Pada gambar 11 merupakan ScreeShootan dari data yang eror.



In [180]: |

Figure 4.14: Program Pengamatan Komponen Informasi

```

File "C:\ProgramData\Anaconda3\lib\site-packages\matplotlib\figure.py",
line 1221, in add_subplot
    self, *args, **kwargs)

File "C:\ProgramData\Anaconda3\lib\site-packages\matplotlib\projections
\__init__.py", line 91, in process_projection_requirements
    projection_class = get_projection_class(projection)

File "C:\ProgramData\Anaconda3\lib\site-packages\matplotlib\projections
\__init__.py", line 65, in get_projection_class
    raise ValueError("Unknown projection '%s'" % projection)

ValueError: Unknown projection '3d'

```

Figure 4.15: Eror matplotlib.pyplot

2. matplotlib.pyplot

Traceback (most recent call last):

```

File "<ipython-input-151-8094433808cc>", line 6, in <module>
    ax = fig.gca(projection='3d')

File "C:\ProgramData\Anaconda3\lib\site-packages\matplotlib\figure.py", line
    return self.add_subplot(1, 1, 1, **kwargs)

File "C:\ProgramData\Anaconda3\lib\site-packages\matplotlib\figure.py", line

```

```
    self, *args, **kwargs)

File "C:\ProgramData\Anaconda3\lib\site-packages\matplotlib\projections\_in
projection_class = get_projection_class(projection)

File "C:\ProgramData\Anaconda3\lib\site-packages\matplotlib\projections\_in
raise ValueError("Unknown projection '%s'" % projection)

ValueError: Unknown projection '3d'

<Figure size 432x288 with 0 Axes>
```

3. Solusi eror tersebut dengan menambahkan menambahkan codingan tersebut seperti berikut.

```
from mpl_toolkits.mplot3d import Axes3D
from matplotlib import cm
fig = plt.figure()
fig.clf()
```

Chapter 5

Conclusion

brief of conclusion

5.1 Jesron Marudut Hatuan /1164077

5.1.1 Teori

1. Alasan mengapa Kata-Kata harus dilakukan vektorisasi lengkapi dengan ilustrasi gambar.

Kata kata harus dilakukan vektorisasi dikarenakan untuk mengukur nilai kemunculan suatu kata yang sama dari sebuah kalimat sehingga kata-kata tersebut dapat di prediksi berapa kemunculannya. Atau juga di buatkan vektorisasi data yang digunakan untuk memprediksi bobot dari suatu kata misalkan mobil dan motor sama-sama kendaraan maka akan dibuat prediksi apakah kata tersebut akan muncul pada kalimat yang kira-kira memiliki bobot yang sama.

Untuk ilustrasinya dapat dilihat pada gambar 5.1



Figure 5.1: Ilustrasi Soal No. 1

2. Alasan Mengapa dimensi dari vektor dataset google bisa mencapai 300 lengkapi dengan ilustrasi gambar.

Dimensi dari vektor dataset google bisa mencapai 300 karena dimensi dari vektor digunakan untuk membandingkan bobot dari setiap kata, misalkan terdapat kata mobil dan motor pada dataset google tersebut setiap kata tersebut di buat dimensi vektor 300 untuk kata mobil dan 300 dimensi vektor juga untuk kata motor kemudian kata tersebut di bandingkan bobot kesamaan katanya maka akan muncul akurasi sekitar 70an persen kesamaan bobot dikarenakan kata mobil dan motor sama sama di gunakan sebagai kendaraan.

Untuk ilustrasinya dapat dilihat pada gambar 5.2

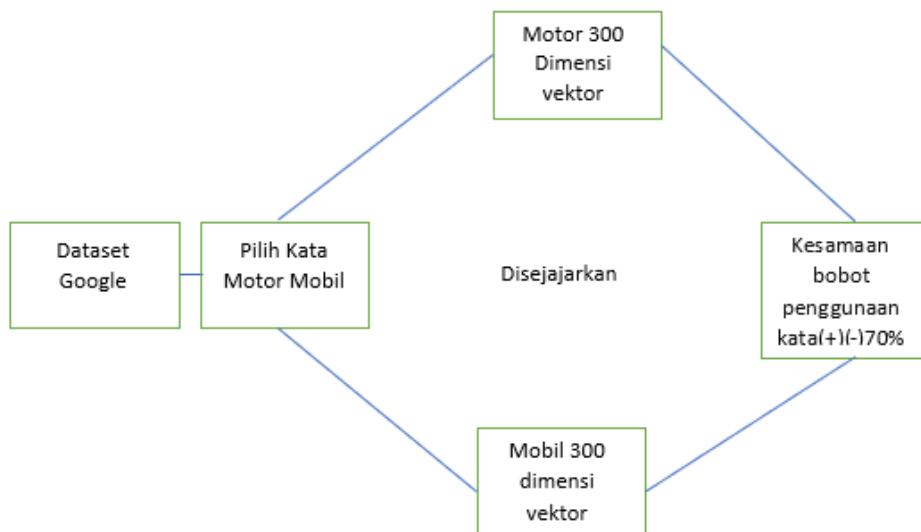


Figure 5.2: Ilustrasi Soal No. 2

3. Penjelasan tentang konsep vektorisasi untuk kata dan dilengkapi dengan ilustrasi atau gambar.

Konsep vektorisasi untuk kata yaitu mengetahui kata tengah dari suatu kalimat utama dengan suatu kalimat contoh (Jangan lupa like dan comment yah makasih) kata tengah tersebut merupakan (dan) yang memiliki bobot sebagai kata tengah dari suatu kalimat atau bobot sebagai objek dari suatu kalimat. hal ini sangat berkaitan dengan dimensi vektor pada dataset google yang 300 tadi karena untuk mendapatkan nilai atau bobot dari kata tengah tersebut di dapatkan dari proses dimensiasi dari kata tersebut.

Untuk ilustrasinya dapat dilihat pada gambar 5.3



Figure 5.3: Ilustrasi Soal No. 3

4. Penjelasan konsep vektorisasi untuk dokumen dan dilengkapi dengan ilustrasi atau gambar.

Konsep vektorisasi untuk dokumen hampir sama seperti vektorisasi untuk kata hanya saja pemilihan kata utama atau kata tengah terdapat pada satu dokumen jadi mesin akan membuat dimensi vektor 300 untuk dokumen dan nanti kata tengahnya akan di sandingkan pada dokumen yang terdapat pada dokumen tersebut.

Untuk ilustrasinya dapat dilihat pada gambar 5.4



Figure 5.4: Ilustrasi Soal No. 4

5. Penjelasan dari mean dan standar deviasi,beserta ilustrasi atau gambar.

Mean adalah nilai rata-rata dari suatu data. Mean disini merupakan petunjuk terhadap kata-kata yang diolah jika kata-kata itu akurasinya tinggi berarti kata tersebut sering muncul begitu juga sebaliknya. Standar deviasi merupakan standar untuk menimbang kesalahan.

Untuk ilustrasinya dapat dilihat pada gambar 5.5

6. Penjelasan Skip-Gram beserta contoh ilustrasi.

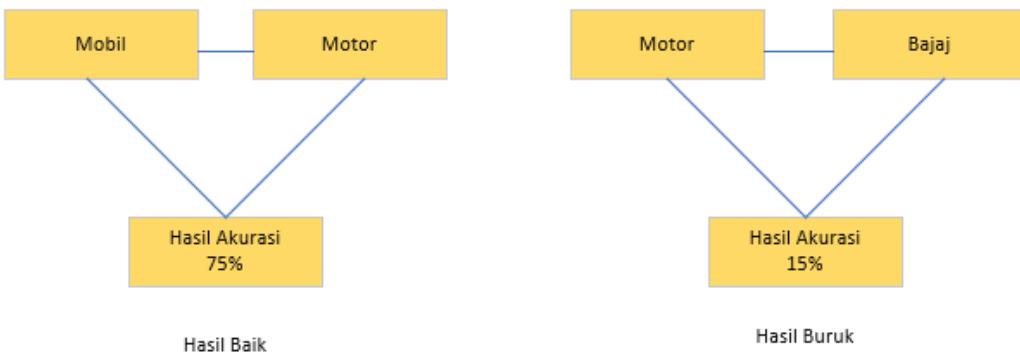


Figure 5.5: Ilustrasi Soal No. 5

Skip-Gram yaitu dimana kata tengah menjadi acuan terhadap kata-kata pelengkap dalam suatu kalimat.

Untuk ilustrasinya dapat dilihat pada gambar 5.6

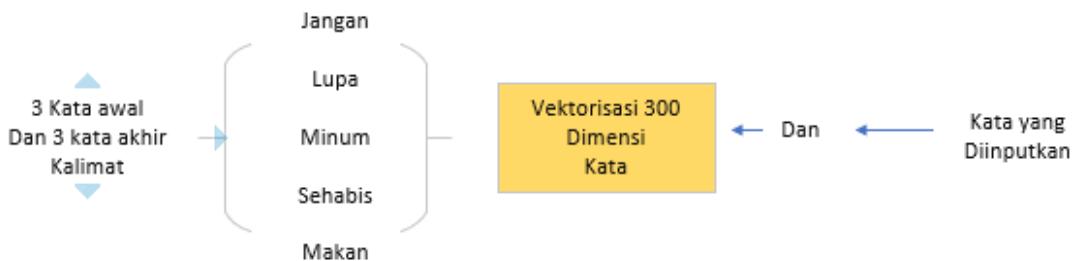


Figure 5.6: Ilustrasi Soal No. 6

5.1.2 Praktek Program

1. Mencoba dataset google, dan penjelasan vektor dari kata love, faith, fall, sick, clear, shine, bag, car, wash, motor, cycle dan mencoba untuk melakukan perbandingan similaritas dari masing-masing kata tersebut. Jelaskan arti dari outputan similaritas.

Output source code dibawah akan memunculkan data vektor untuk kata love. bahwa vektor memiliki array sebanyak 300 dimensi. Hasil pada source code tersebut dapat dilihat pada gambar 5.7.

```
gmodel['love']
```

```
In [34]: gmodel['love']
Out[34]:
array([ 0.10302734, -0.15234375,  0.02587891,  0.16503906, -0.16503906,
```

Figure 5.7: Love

Output source code dibawah akan memunculkan data vektor untuk kata faith. Bahwa vektor memiliki array sebanyak 300 dimensi. Hasil pada source code tersebut dapat dilihat pada gambar 5.8.

```
gmodel['faith']
```

```
In [35]: gmodel['faith']
Out[35]:
array([ 0.26367188, -0.04150391,  0.1953125 ,  0.13476562, -0.14648438,
```

Figure 5.8: Faith

Output source code dibawah akan memunculkan data vektor untuk kata fall. Bahwa vektor memiliki array sebanyak 300 dimensi. Hasil pada source code tersebut dapat dilihat pada gambar 5.9.

```
gmodel['fall']
```

```
In [36]: gmodel['fall']
Out[36]:
array([-0.04272461,  0.10742188, -0.09277344,  0.16894531, -0.1328125 ,
```

Figure 5.9: Fall

Output source code dibawah akan memunculkan data vektor untuk kata sick. Bahwa vektor memiliki array sebanyak 300 dimensi. Hasil pada source code tersebut dapat dilihat pada gambar 5.10.

```
gmodel['sick']
```

Output source code dibawah akan memunculkan data vektor untuk kata clear. Bahwa vektor memiliki array sebanyak 300 dimensi. Hasil pada source code tersebut dapat dilihat pada gambar 5.11.

```
In [37]: gmodel['sick']
Out[37]:
array([ 1.82617188e-01,  1.49414062e-01, -4.05273438e-02,  1.64062500e-01,
```

Figure 5.10: Sick

```
In [38]: gmodel['clear']
Out[38]:
array([-2.44140625e-04, -1.02050781e-01, -1.49414062e-01, -4.24804688e-02,
```

Figure 5.11: Clear

```
gmodel['clear']
```

Output source code dibawah akan memunculkan data vektor untuk kata shine. Bahwa vektor memiliki array sebanyak 300 dimensi. Hasil pada source code tersebut dapat dilihat pada gambar 5.12.

```
gmodel['shine']
```

```
In [39]: gmodel['shine']
Out[39]:
array([-0.12402344,  0.25976562, -0.15917969, -0.27734375,  0.30273438,
```

Figure 5.12: Shine

Output source code dibawah akan memunculkan data vektor untuk kata bag. Bahwa vektor memiliki array sebanyak 300 dimensi. Hasil pada source code tersebut dapat dilihat pada gambar 5.13.

```
gmodel['bag']
```

```
In [40]: gmodel['bag']
Out[40]:
array([-0.03515625,  0.15234375, -0.12402344,  0.13378906, -0.11328125,
```

Figure 5.13: Bag

Output source code dibawah akan memunculkan data vektor untuk kata car. Bahwa vektor memiliki array sebanyak 300 dimensi. Hasil pada source code tersebut dapat dilihat pada gambar 5.14.

```
In [41]: gmodel['car']
Out[41]:
array([ 0.13085938,  0.00842285,  0.03344727, -0.05883789,  0.04003906,
```

Figure 5.14: Car

```
gmodel['car']
```

Output source code dibawah akan memunculkan data vektor untuk kata wash. Bahwa vektor memiliki array sebanyak 300 dimensi. Hasil pada source code tersebut dapat dilihat pada gambar 5.15.

```
gmodel['wash']
```

```
In [42]: gmodel['wash']
Out[42]:
array([ 9.46044922e-03,  1.41601562e-01, -5.46875000e-02,  1.34765625e-01,
```

Figure 5.15: Wash

Output source code dibawah akan memunculkan data vektor untuk kata motor. Bahwa vektor memiliki array sebanyak 300 dimensi. Hasil pada source code tersebut dapat dilihat pada gambar 5.16.

```
gmodel['motor']
```

```
In [43]: gmodel['motor']
Out[43]:
array([ 5.73730469e-02,  1.50390625e-01, -4.61425781e-02, -1.32812500e-01,
```

Figure 5.16: Motor

Output source code dibawah akan memunculkan data vektor untuk kata cycle. Bahwa vektor memiliki array sebanyak 300 dimensi. Hasil pada source code tersebut dapat dilihat pada gambar 5.17.

```
gmodel['cycle']
```

Pada source code dibawah menunjukkan hasil score perbandingan kata apakah kata motor dan cycle memiliki ke samaan atau tidak. Hasil pada source code tersebut dapat dilihat pada gambar 5.18.

```
In [44]: gmodel['cycle']
Out[44]:
array([ 0.04541016,  0.21679688, -0.02709961,  0.12353516, -0.20703125,
```

Figure 5.17: Cycle

```
In [45]: gmodel.similarity('motor','cycle')
Out[45]: 0.1794929675764453
```

Figure 5.18: Similariti Pada Kata Motor dan Cycle

```
gmodel.similarity('motor','cycle')
```

Pada source code dibawah menunjukkan hasil score perbandingan kata apakah kata wash dan motor memiliki ke samaan atau tidak. Hasil pada source code tersebut dapat dilihat pada gambar 5.19.

```
gmodel.similarity('wash','motor')
```

```
In [46]: gmodel.similarity('wash','motor')
Out[46]: 0.10280077965607967
```

Figure 5.19: Similariti Pada Kata Wash dan Motor

Untuk Motor dan Cycle hasilnya adalah 17%

Untuk Wash dan Motor hasilnya adalah 10%

Artinya Motor dan Cyle memang dalam kategori yang sama misalnya dalam kategori kata-kata yang disatukan/berpasangan. Mesin sudah mengetahui bahwa keduanya dapat dikategorikan sebagai sepasang kata.

2. Penjelasan kata dan ilustrasi fungsi dari extract_words dan PermuteSentences.

Extract_Words merupakan function untuk menambahkan, menghilangkan atau menghapuskan, hal hal yang tidak penting atau tidak perlu di dalam teks. Pada gambar 5.20 berikut ini menggunakan function extract_words untuk menghapus komen dengan python style , mencari data yang diinginkan, dan memberikan spasi pada teks.

```
In [48]: import re
.... def extract_words(luarbiasa):
....     luarbiasa = luarbiasa.lower()
....     luarbiasa = re.sub(r'<[^>]+>', ' ', luarbiasa) #hapus tag html
....     luarbiasa = re.sub(r'(\w)\\'(\w)', ' ', luarbiasa) #hapus petik satu
....     luarbiasa = re.sub(r'\W', ' ', luarbiasa) #hapus tanda baca
....     luarbiasa = re.sub(r'\s+', ' ', luarbiasa) #hapus spasi yang berurutan
....     return luarbiasa.split()
```

Figure 5.20: Extract_Words

PermuteSentences berfungsi untuk melakukan pengacakan data supaya memperoleh data yang teratur. Ini merupakan class yang digunakan untuk melakukan pengocokan secara acak pada data yang ada. Digunakan cara ini agar tidak terjadi kelebihan memori pada saat dijalankan. Hasilnya dapat dilihat pada gambar 5.21.

```
In [49]: import random
.... class PermuteSentences(object):
....     def __init__(self, luarbiasas):
....         self.luarbiasas = luarbiasas
....     def __iter__(self):
....         shuffled = list(self.luarbiasas)
....         random.shuffle(shuffled)
....         for luarbiasa in shuffled:
....             yield luarbiasa
```

Figure 5.21: Permute Sentences

3. Fungsi dari librari gensim TaggedDocument dan Doc2Vec disertai praktek pemakaianya.

Fungsi dari library gensim yaitu sebagai pemodelan topik tanpa pengawasan dan pemrosesan bahasa alami, atau bisa kita sebut dengan unsupervised. tagged document itu sendiri untuk memasukan kata-kata pada setiap dokumennya yang akan di vektorisasi. Fungsi dari doc2vec itu sendiri ialah untuk membandingkan bobot data yang terdapat pada dokumen lainnya, apakah kata-kata didalamnya ada yang sama atau tidak. Ketika di running maka tidak terjadi apa-apa, seperti pada gambar 5.22

Name	Type	Size	Value

Variable explorer File explorer Help

IPython console

Console 1/A X

```
In [3]: from gensim.models.doc2vec import TaggedDocument
...: from gensim.models import Doc2Vec
```

Figure 5.22: Gensim TaggedDocument dan Doc2vec

4. Penjelasan kata dan praktik cara menambahkan data training dari file yang dimasukkan kepada variabel dalam rangka melatih model doc2vec.

Untuk menambahkan data training dengan cara melakukan import library os, library os sebelumnya berfungsi untuk melakukan interaksi antara python dengan os laptop masing-masing, setelah itu kita buat variabel unsup_sentences. Kemudian pilih direktori tempat data kita disimpan. Lalu menyortir data yang terdapat pada folder aclImdb dan membaca file tersebut dengan ekstensi .txt.

```
import os
unsup_sentences = []
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]))
```

Hasil dari code yang diatas ialah terdapatnya data training dengan jumlah 10000 data dari variabel unsup_sentences hasil running dari folder aclImdb dapat dilihat pada gambar 5.23

```
for dirname in ["review_polarity/txt_sentoken/pos", "review_polarity/txt_sentoken/neg"]:
```

Name	Type	Size	Value
dirname	str	1	test/neg
fname	str	1	9_4.txt
sent	str	1	David Bryce's comments nearby are exceptionally well written and infor ...
unsup_sentences	list	100000	[TaggedDocument, TaggedDocument, TaggedDocument, TaggedDocument, Tagge ...]
words	list	391	['david', 'bryc', 'comments', 'nearby', 'are', 'exceptionally', 'well' ...]

Figure 5.23: Aclimbdb

```

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)]))

```

Untuk code berikutnya akan menambahkan data training pada variabel unsup_sentences sekitar 64.720 data, seperti pada gambar 5.24

Name	Type	Size	Value
dirname	str	1	txt_sentoken/neg
fname	str	1	cv999_14636.txt
i	int	1	24
sent	str	1	after watching _a_night_at_the_roxbury_ , you'll be left with exactly ...
unsup_sentences	list	164720	[TaggedDocument, TaggedDocument, TaggedDocument, TaggedDocument, Tagge ...]
words	list	11	['after', 'watching', '_a_night_at_the_roxbury_', 'yo', 'l', 'be', 'le ...']

Figure 5.24: Aclimbdb

```

with open("stanfordSentimentTreebank/original_rt_snippets.txt",encoding='UTF-8'):
    for i, sent in enumerate(f):
        words = extract_words(sent)
        unsup_sentences.append(TaggedDocument(words, ["rt-%d" % i]))

```

Untuk code berikutnya akan menambahkan data training pada variabel unsup_sentences sekitar 10.605 data, seperti pada gambar 5.25

Name	Type	Size	Value
dirname	str	1	txt_sentoken/neg
fname	str	1	cv999_14636.txt
i	int	1	10604
sent	str	1	Her fans walked out muttering words like ``horrible'' and ``terrible,' ...
unsup_sentences	list	175325	[TaggedDocument, TaggedDocument, TaggedDocument, TaggedDocument, Tagge ...]
words	list	29	['her', 'fans', 'walked', 'out', 'muttering', 'words', 'like', 'horrib ...']

Figure 5.25: Aclimbdb

5. Penjelasan kata dan praktek mengapa harus dilakukan pengocokan dan pembersihan data.

Pengocokan data itu berguna untuk mengacak data supaya pada saat data di running bisa berjalan lebih baik dan hasil presentase akhirnya bisa lebih bagus. Sedangkan pembersihan data untuk memberikan ruang bagi ram komputer kita setelah melakukan running data sebanyak 3 juta lebih, agar lebih ringan saat proses selanjutnya. Hasil dari pengacakan data tidak ditampilkan pada spyder. Dan sebelumnya memori yang terpakai itu sekitar 87% , setelah dikosongkan jadi 71%. Untuk source codenya dapat dilihat sebagai berikut:

```
# Pengocokan data
mute = PermuteSentences(unsup_sentences)
# Pembersihan data
model.delete_temporary_training_data(keep_inference=True)
```

6. Penjelasan kata dan praktek kenapa model harus di save dan kenapa temporari training harus dihapus.

Save data adalah proses menyimpan file hasil dari proses training data sebelumnya, ini dilakukan untuk memberikan keringanan pada RAM agar pada saat kita akan melakukan training lagi, model tersebut tinggal di muat tanpa harus melakukan traning dari awal dan bisa menghemat waktu. Sedangkan untuk delete temporary training data untuk menghapus data training yang sebelumnya sudah dilakukan dan disimpan, tujuannya memberikan keringanan pada RAM. Karena setelah melakukan proses training, RAM biasanya jadi berat untuk dijalankan bahkan komputer menjadi lemot. Untuk source codenya dapat dilihat sebagai berikut:

```

# Save data
model.save('ocean.d2v')
# Delete temporary data
model.delete_temporary_training_data(keep_inference=True)

```

7. Penjelasan dengan kata dan praktek maksud dari infer code.

Infer code yaitu membandingkan kata yang tercantum dengan code pada dokumen yang sudah di muat sebelumnya. Selain itu infer_code juga menghitung code dari kata yang dicantumkan pada model yang telah kita buat. Dan seharusnya kata yang dicantumkan lebih panjang agar hasilnya bisa lebih baik.

```
model.infer_vector(extract_words("I will go home"))
```

Pada source code tersebut menghasilkan keluaran seperti pada gambar 5.26.

```

In [14]: model.infer_vector(extract_words("I will go home"))
Out[14]:
array([-0.13122962, -0.2173184 , -0.14073701,  0.47829896,  0.10263892,
       -0.12006506, -0.02554635, -0.06321128,  0.01674332,  0.21312888,
      -0.03759272, -0.21068032,  0.08137176, -0.04404473,  0.23163871,
     -0.11791784, -0.02098055, -0.44984344, -0.21420991,  0.10826829,
    -0.24330835, -0.13386108,  0.05421483, -0.10329439, -0.16347748,
     0.03265174, -0.18829556,  0.14504528, -0.03915659, -0.39758494,
     0.05695166,  0.02784907, -0.15537408,  0.14412752, -0.0224928 ,
    -0.40351996, -0.11055487,  0.18106677,  0.09285883, -0.2614472 ,
     0.24014267,  0.15855208, -0.1287663 ,  0.53696984,  0.08572944,
     0.2602722 ,  0.3341717 ,  0.12752089,  0.15490048,  0.25898734,
    -0.23501003, -0.03285267], dtype=float32)

```

Figure 5.26: Infer Code

8. Penjelasan dengan praktek dan kata maksud dari cosine similarity.

Cosine similarity merupakan proses membandingkan vektorisasi data diantara kedua kata yang di masukkan, apabila hasil presentase dari kedua kata tersebut lebih dari 50 persen kemungkinan kata tersebut terdapat dalam 1 file. Tapi jika kurang dari 50 persen kata tersebut tidak terdapat dalam 1 file. Hasil yang didapatkan pada code tersebut hanya 0.4 persen itu dikarenakan kata pertama dan kedua tidak memiliki kesamaan vektorisasi dan tidak terdapat pada salah satu dokumen.

```

from sklearn.metrics.pairwise import cosine_similarity
cosine_similarity(
    [model.infer_vector(extract_words("she going to school, after wash hand"))],
    [model.infer_vector(extract_words("Services sucks."))])

```

Pada source code tersebut menghasilkan keluaran seperti pada gambar 5.27.

```

In [15]: from sklearn.metrics.pairwise import cosine_similarity
.... cosine_similarity(
....     [model.infer_vector(extract_words("she going to school, after wash hand"))],
....     [model.infer_vector(extract_words("Services sucks."))])
Out[15]: array([[0.46642035]], dtype=float32)

```

Figure 5.27: Cosine Similarity

9. Penjelasan dengan praktik score dari cross validation masing-masing metode.

```

scores = cross_val_score(clf, sentvecs, sentiments, cv=5)
np.mean(scores), np.std(scores)

```

Pada source code tersebut melakukan perhitungan persentase dengan menggunakan cross validation yang mana mengecek data score menggunakan metode kneighborsClassifier untuk menghasilkan typedata float64 dimana terdapat 5 data dari score yang typedatanya float64 tersebut. menghasilkan keluaran seperti pada gambar 5.28.

scores	float64	(5,)	[0.75666667 0.79 0.76666667 0.74 0.77166667]	
Variable explorer	File explorer	Help		
Python console				
Console 1/A				

```

In [20]: scores = cross_val_score(clf, sentvecs, sentiments, cv=5)
.... np.mean(scores), np.std(scores)
Out[20]: (0.765, 0.016532795690182997)

```

Figure 5.28: Cross Validation Metode 1

```

scores = cross_val_score(clfrf, sentvecs, sentiments, cv=5)
np.mean(scores), np.std(scores)

```

```
Out[21]: (0.7150000000000001, 0.02378141197564929)
```

Figure 5.29: Cross Validation Metode 2

Pada source code tersebut melakukan perhitungan presentase dengan menggunakan cross validation yang mana mengecek data score menggunakan metode Random forest menghasilkan keluaran seperti pada gambar 5.29.

```
from sklearn.pipeline import make_pipeline
from sklearn.feature_extraction.text import CountVectorizer, TfidfTransformer
pipeline = make_pipeline(CountVectorizer(), TfidfTransformer(), RandomForestClassifier())
scores = cross_val_score(pipeline, sentences, sentiments, cv=5)
np.mean(scores), np.std(scores)
```

Pada source code tersebut melakukan skoring dari vektorisasi, tfidf, dan rf lalu dibuat perbandingan yang menghasilkan keluaran seperti pada gambar 5.30.

```
Out[22]: (0.749, 0.01271918935222596)
```

Figure 5.30: Cross Validation Metode 3

5.1.3 Penanganan Eror

1. Screenshot Error dapat dilihat pada gambar 5.31

```
In [1]: import gensim, logging
F:\anaconda\lib\site-packages\gensim\utils.py:1197: UserWarning: detected Windows; aliasing
chunkize to chunkize_serial
    warnings.warn("detected Windows; aliasing chunkize to chunkize_serial")
```

Figure 5.31: eror

2. Tuliskan kode eror dan jenis erornya.

Kode eror

```
import gensim, logging
```

Jenis eronya.

```
F:\anaconda\lib\site-packages\gensim\utils.py:1197: UserWarning: detected Windows
warnings.warn("detected Windows; aliasing chunkize to chunkize_serial")
```

3. Solusi Pemecahan Error Tersebut.

yaitu dengan cara memberikan source code seperti dibawah ini:

```
import warnings
warnings.filterwarnings(action='ignore', category=UserWarning, module='gensim')
```

Kenapa demikian itu karena gensim memberi tahu bahwa ia akan melakukan chunkize ke fungsi yang berbeda karena kita menggunakan os tertentu.

5.2 Jesron Marudut Hatuan/1164077

5.2.1 Teori

1. Mengapa file suara harus di lakukan MFCC.

Alasan mengapa File suara atau audio harus dilakukan MFCC karena MFCC dapat mengubah file suara/frekuensi suara ke dalam sebuah bentuk data, yaitu menjadi data vektor yang nantinya dapat diolah menjadi sebuah output dimana telah dilakukan ekstraksi oleh MFCC selanjutnya direalisasikan sebagai data matrix. Contoh ilustrasi dapat dilihat pada gambar 5.32.

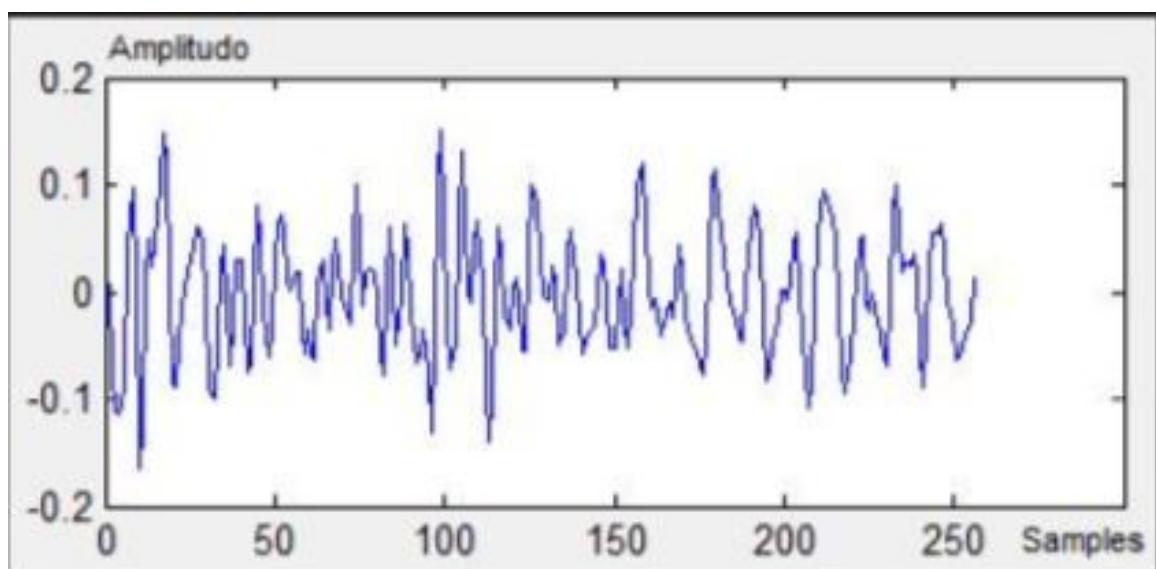


Figure 5.32: Suara ke MFCC

2. Penjelasan tentang konsep dasar neural network.

Neural Network merupakan cara untuk menkomputasi yang efisien dimana tema utamanya dipinjam dari analogi jaringan saraf biologis. Neural Network juga biasa disebut dengan jaringan saraf tiruan dimana Neural Network sebenarnya dapat mengadopsi dari kemampuan otak manusia yang dapat memberikan rangsangan, melakukan proses, dan memberikan output. Contoh ilustrasi dapat dilihat pada gambar 5.33.

3. Penjelasan tentang konsep dari pembobotan dalam neural network.

Neural Network dapat dikonfigurasi untuk aplikasi tertentu, seperti pengenalan pola atau klasifikasi data, contohnya saat Neural Network melakukan

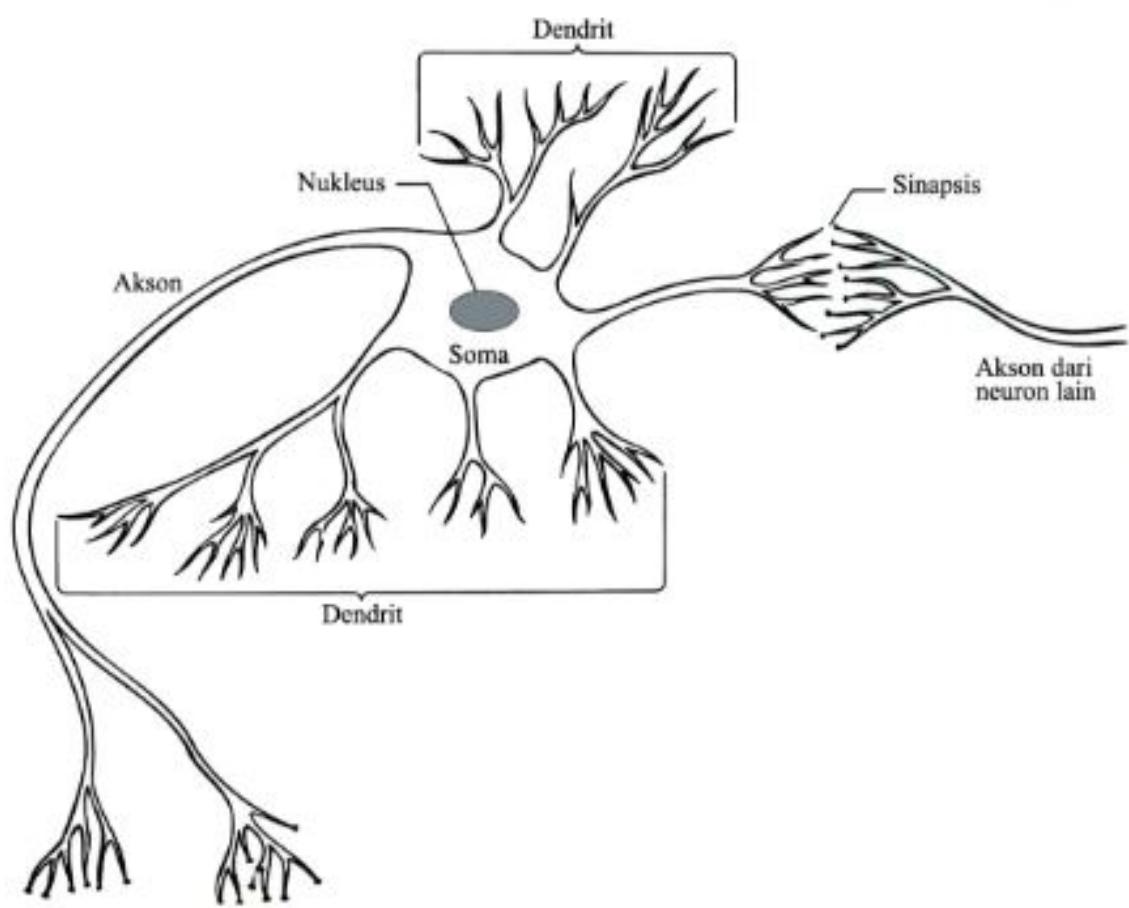


Figure 5.33: Neural Network

penyesuaian koneksi sinaptik antara Neuron, maka dilakukan dengan menyeuaikan nilai bobot yang ada pada setiap koneksi baik dari input, Neuron maupun output disinkronkan dengan penyesuaian koneksi sinaptik antar neuron itu sendiri. Contoh ilustrasi dapat dilihat pada gambar 5.34.

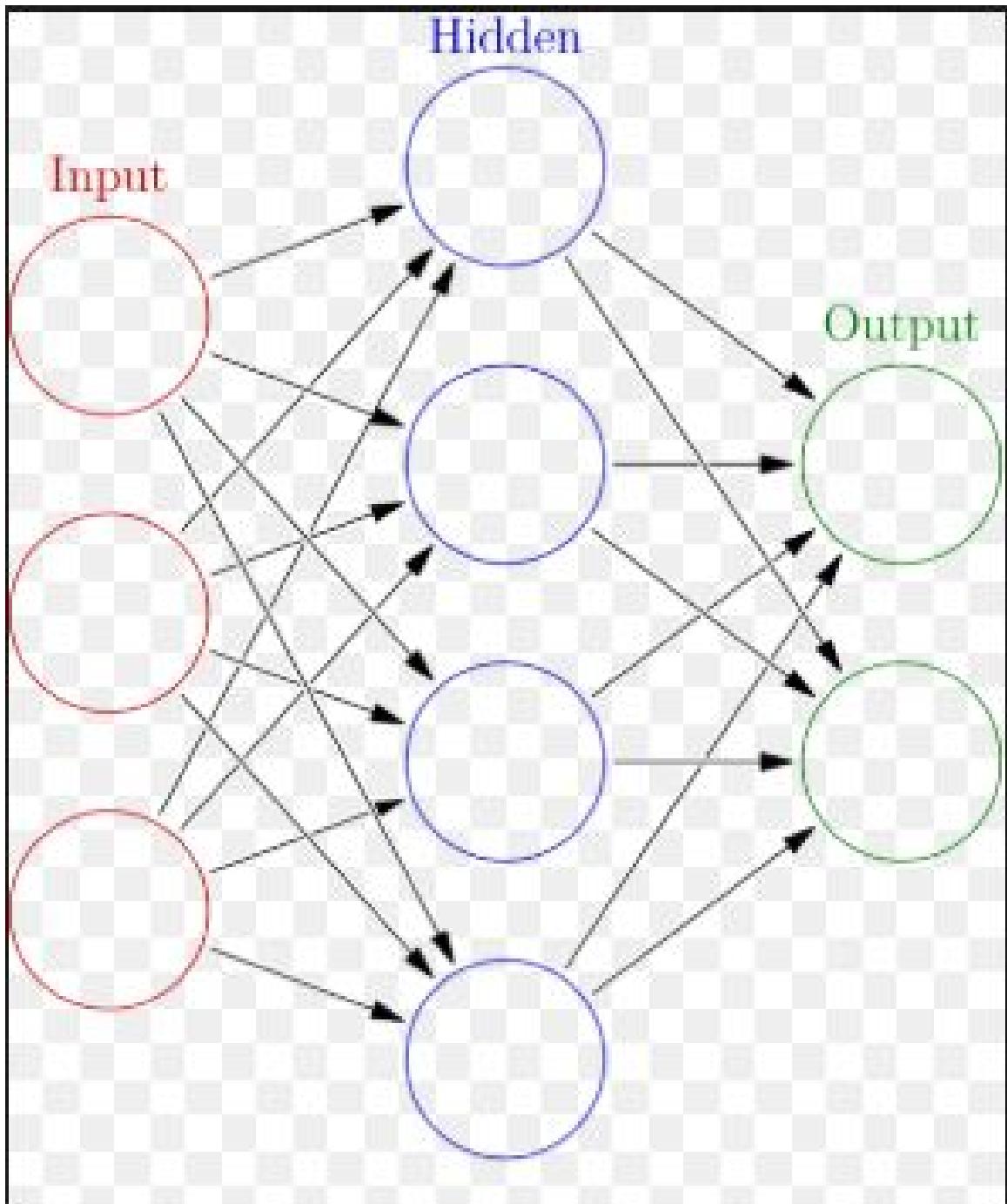


Figure 5.34: Konsep Pembobotan pada Neural Network

4. Penjelasan tentang konsep fungsi aktifasi dalam Neural Network.

Dalam Neural Network, fungsi aktivas adalah setiap Neuron dapat mempunyai tingkat aktivasi yang merupakan fungsi dari input yang masuk padanya. Aktivasi yang dikirim suatu Neuron ke Neuron lain berupa sinyal dan hanya dapat mengirim sekali dalam satu waktu, meskipun sinyal tersebut disebarluaskan pada beberapa neuron yang lain. Contoh ilustrasi dapat dilihat pada gambar 5.35.



Figure 5.35: Fungsi Aktivasi pada Neural Network

5. Penjelasan cara membaca hasil plot dari MFCC.

Cara membaca dari MFCC yaitu nanti akan ada outputan berbentuk grafik setelah melakukan plot dari MFCC. Kemudian terdapat frekuensi/Hz pada suara frekuensi biasanya vertikal atau biasa disimbolkan dengan sumbu y, Lalu terdapat waktu yang mana waktu diartikan dalam simbol sumbu x. Sedangkan pada bagian dalam dapat disimbolkan dengan sumbu z yang merupakan power atau kekuatan dari lagu atau suara atau audio yang dihasilkan. Untuk warna biru itu merupakan suara rendah, yang merah merupakan tinggi apabila daya frekuensi nya misalkan suara siul berarti dominan warna merah karena siul biasanya pada nada yang tinggi sedangkan jika bass dominan biru karena bass merupakan nada rendah. Contoh ilustrasi dapat dilihat pada gambar 5.36.

6. Penjelasan tentang apa itu one-hot encoding.

One-hot encoding yaitu sebuah cara untuk representasi data dari variabel kategori sebagai vektor biner. Yaitu nilai kategorika harus dipetakan ke nilai integer. Selanjutnya, setiap nilai integer direpresentasikan sebagai vektor biner yang semuanya bernilai nol kecuali indeks integer, yang dapat ditandai dengan 1. Contoh ilustrasi dapat dilihat pada gambar 5.37.

7. Penjelasan fungsi dari np.unique dan to_categorical dalam kode program.

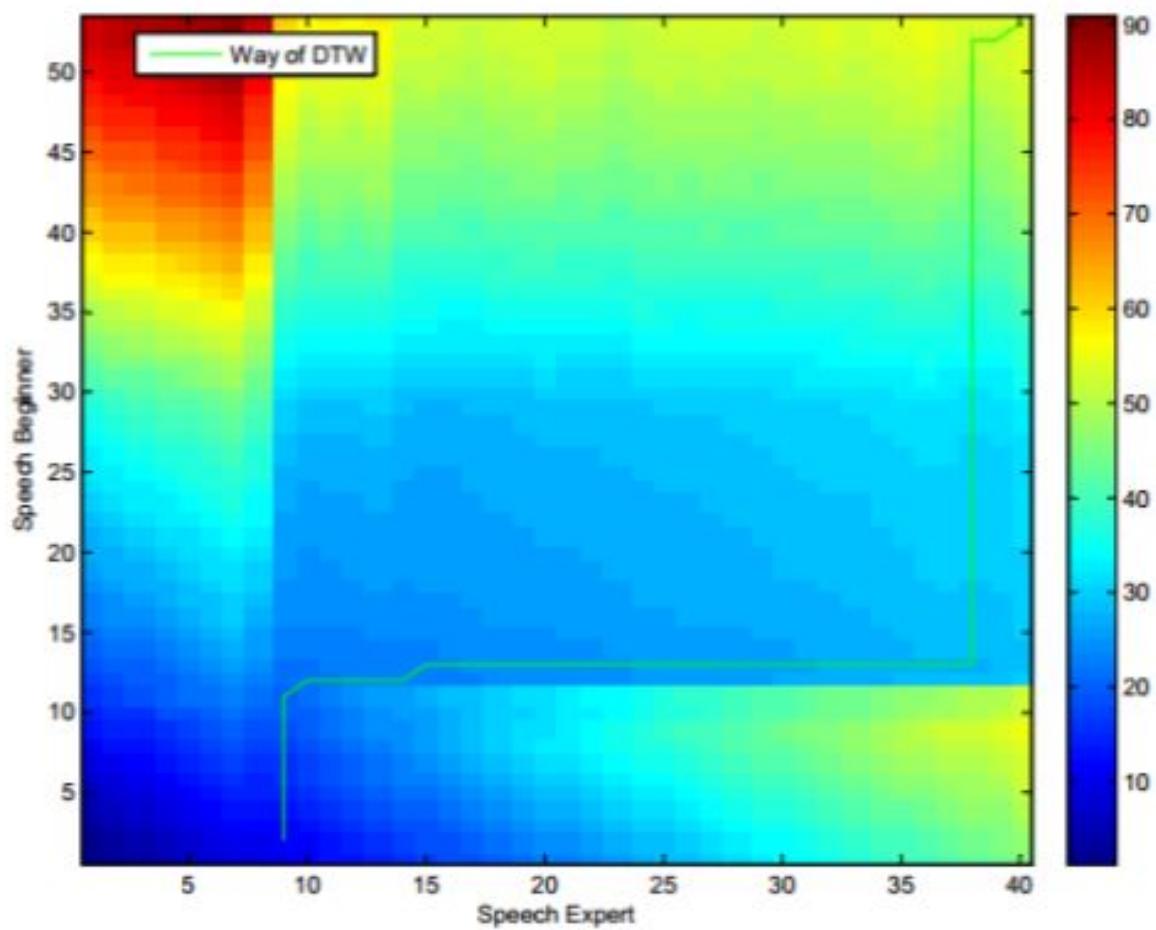


Figure 5.36: Membaca Hasil Plot dari MFCC

Original data:		One-hot encoding format:						
id	Color		id	White	Red	Black	Purple	Gold
1	White		1	1	0	0	0	0
2	Red		2	0	1	0	0	0
3	Black		3	0	0	1	0	0
4	Purple		4	0	0	0	1	0
5	Gold		5	0	0	0	0	1

Figure 5.37: One-Hot Encoding

- np.unique adalah cara untuk mengekstraksi elemen-elemen unik tertentu yang terdapat dalam array.
- sedangkan to_categorical adalah cara untuk mengubah vektor kelas yang berupa integer menjadi matriks kelas biner.

```
>>> a = np.array([1,1,1,2,2,3,4,4,4,4,5,5,5,5,5], float)
>>> np.unique(a)
array([ 1.,  2.,  3.,  4.,  5.])
```

Figure 5.38: np.unique

```
> labels
array([0, 2, 1, 2, 0])
> to_categorical(labels)
array([[ 1.,  0.,  0.],
       [ 0.,  0.,  1.],
       [ 0.,  1.,  0.],
       [ 0.,  0.,  1.],
       [ 1.,  0.,  0.]], dtype=float32)
```

Figure 5.39: to_categorical

8. Penjelasan fungsi dari Sequential dalam kode program.

Fungsi dari Sequential dalam kode program yaitu sebuah jenis model yang dapat digunakan dalam perhitungan ataupun code program yang dapat direalisasikan. Neural Networks Sequential dapat membangun fitur tingkat tinggi melalui lapisan yang berurutan. Sequential juga merupakan proses dimana membandingkan setiap elemen larik satu per satu secara beruntun, mulai dari elemen pertama, sampai dengan elemen terakhir atau elemen yang dicari sudah ditemukan. Contoh ilustrasi dapat dilihat pada gambar 5.40.

```

Pencarian Sequential
Jumlah data : 5
Indeks [1] : 1
Indeks [2] : 8
Indeks [3] : 4
Indeks [4] : 6
Indeks [5] : 9
Masukkan Data Yang Ingin Dicari : 6
6 Ditemukan pada indeks ke-4

```

Figure 5.40: One-Hot Encoding

5.2.2 Praktek Program

1. Penjelasan dari data GTZAN Genre Collection dan data dari freesound dan Berupa kode program untuk meload data tersebut untuk digunakan pada MFCC.

Isi dari data GTZAN Genre Collection itu isinya berupa data musik yang sudah di folderkan berdasarkan genre lagunya (dikelompokkan) dengan ekstensi .au yang akan kita lakukan proses MFCC. Sedangkan freesound hanya untuk 1 lagu saja dengan ekstensi .wav.

```
filename_jazz = 'genres/jazz/jazz.00062.au'
x_jazz, sr_jazz = librosa.load(filename_jazz, duration=10)
```

- Code pada baris pertama yaitu Filename jazz merupakan sebuah variabel yang nantinya berisikan direktori dari file yang dituju, pada code ini digunakan file audio dari genre jazz.
- Code pada baris kedua yaitu membuat variabel X jazz dan sr jazz yang digunakan untuk meload file dari variabel filename jazz menggunakan librari Librosa dengan durasi 10, yang nantinya akan digunakan pada Mfcc.

Hasilnya dapat dilihat pada gambar 5.41.

2. Penjelasan perbaris kode program dengan kata-kata dan dilengkapi ilustrasi gambar fungsi dari display mfcc() .

Name	Type	Size	Value
filename_jazz	str	1	genres/jazz/jazz.00062.au
sr_jazz	int	1	22050
x_jazz	float32	(220500,)	[-0.28155518 -0.3609314 -0.4019165 ... -0.04586792 -0.052... -0.0 ...]

Figure 5.41: GTZAN

```
display_mfcc('genres/hiphop/hiphop.00028.au')
```

Pada kode tersebut menjelaskan display_mfcc untuk menampilkan vektorisasi dari sebuah suara yang akan menampilkan Plot dan Bar dari inputan code dan file suara hiphop.00028.au. Hasilnya dapat dilihat pada gambar 5.42.

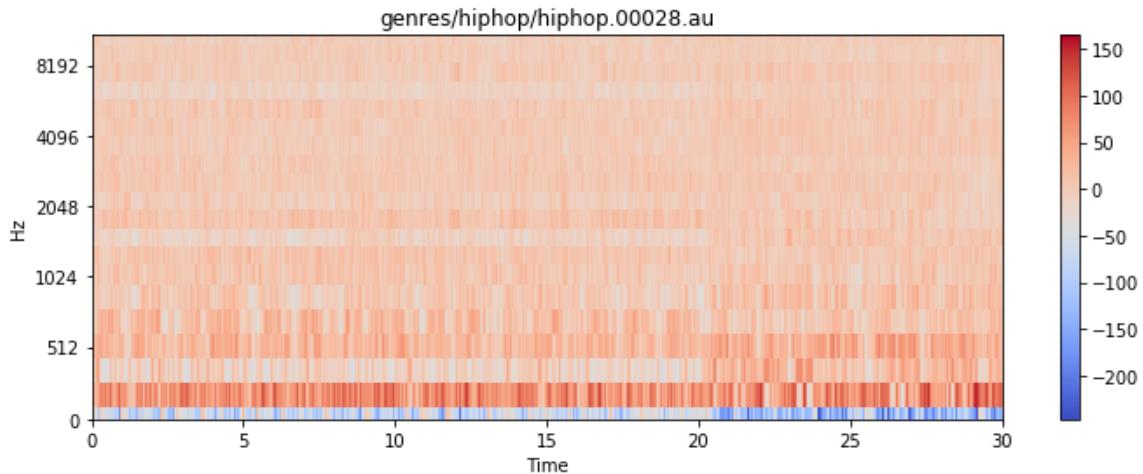


Figure 5.42: Display MFCC

3. Penjelasan perbaris kode program dengan kata-kata dan dilengkapi ilustrasi gambar fungsi dari extract features song(). Jelaskan juga mengapa yang diambil 25.000 baris pertama?

```
display_mfcc('genres/hiphop/hiphop.00028.au')
```

- Pada kode baris pertama berfungsi untuk memuat inputan.
- Pada kode baris kedua itu akan memuat data inputan dengan menggunakan librosa.
- Kemudian pada parameter inputan yaitu y untuk membuat sebuah fitur pada mfcc.

- Selanjutnya me-return menjadi array dan akan mengambil 25000 data saja dari hasil vektorisasi dalam 1 lagu yang sudah di display_mfcc.
 - Mengapa yang dimabil 25.000 baris pertama karena nemuat nilai-nilai MFCC untuk lagu, tetapi karena nilai-nilai ini mungkin antara negatif 250 hingga positif 150, mereka tidak baik untuk jaringan saraf.
4. Penjelasan perbaris kode program dengan kata-kata dan dilengkapi ilustrasi gambar fungsi dari generate features and labels().
- ```
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('Processing %d songs in %s genre...' % (len(sound_files), 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 cth blues : 1000000000 classical : 0000000001
 label_uniq_ids, label_row_ids = np.unique(all_labels, return_index=True)
 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
```

Pada kode diatas berfungsi melakukan fungsi yang sebelumnya kita telah jalankan. Lalu pada bagian genres itu disesuaikan dengan nama folder dataset. Untuk baris selanjutnya itu akan melakukan looping dari folder genres dengan ekstensi .au. Lalu akan memanggil fungsi ekstrak lagu. Pada setiap file yang terdapat pada folder itu akan di ekstrak menjadi vektor dan akan dimasukan kedalam fitur. Dan fungsi append itu menumpuk file-file yang telah di vektorisasi.

5. Penjelasan dengan kata dan praktek kenapa penggunaan fungsi generate features and labels() sangat lama ketika meload dataset genre.

```
features, labels = generate_features_and_labels()
```

Pada kode diatas menjelaskan dikarenakan terdapat 10 folder dengan genre berbeda, dan didalamnya terdapat 100 audio maka itu akan lama untuk meload

dataset genre. Dari setiap folder itu akan dilakukan features dan perubahan label dengan ekstrasi data menggunakan mfcc. Karena banyaknya jumlah file maka proses loadnya pun lama. Hasil dapat dilihat pada gambar 5.43.

| Name     | Type    | Size          | Value                                                             |
|----------|---------|---------------|-------------------------------------------------------------------|
| features | float64 | (1000, 25000) | [-0.81999071 -0.81014303 -0.75184401 ... -0.01818394...<br>0 ...] |
| labels   | float32 | (1000, 10)    | [1. 0. 0. ... 0. 0. 0.]<br>[1. 0. 0. ... 0. 0. 0.]                |

Figure 5.43: Generate Features and Labels

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

```
training_split = 0.8
```

Pada kode diatas berfungsi untuk memisahkan data training dan dataset sebesar 80% digunakan untuk memudahkan dalam melakukan pengacakan atau pengocokan nantinya. Dimana 80% merupakan data training dan sisanya 20% merupakan datatestnya. data training perlu lebih banyak agar saat dilakukan pengocokan tidak teracak dalam urutan yang berbeda. Hasilnya dapat dilihat pada gambar 5.44.

```
training_split float 1 0.8
```

Figure 5.44: Training Data Sebesar 80%

7. Mempraktekkan dan jelaskan masing-masing parameter dari fungsi Sequential().

```
model = Sequential([
 Dense(100, input_dim=np.shape(train_input)[1]),
 Activation('relu'),
 Dense(10),
 Activation('softmax'),
])
```

Pada kode diatas dijelaskan bahwa:

- Layer pertama dense dari 100 neuron untuk inputan

- Activationnya menggunakan fungsi relu yaitu jika ada inputan dengan nilai maksimum maka inputan itu yang akan terpilih.
  - Dense 10 mengkategorikan 10 neuron untuk jenis genrenya untuk output nya.
  - Untuk dense diatas aktivasinya menggunakan fungsi Softmax
8. Praktekkan dan jelaskan masing-masing parameter dari fungsi compile() dan tunjukkan keluarannya dengan fungsi summary.
- ```
model.compile(optimizer='adam',
              loss='categorical_crossentropy',
              metrics=['accuracy'])
print(model.summary())
```
- Pada kode diatas dijelaskan bahwa:
- Menggunakan algortima adam sebagai optimizer. Adam yaitu algoritme pengoptimalan yang dapat digunakan sebagai ganti dari prosedur penu-runan gradien stokastik klasik untuk memperbarui bobot jaringan yang berulang berdasarkan data training.
 - Loss nya menggunakan categorical_crossentropy untuk fungsi optimasi skor
- Hasilnya dapat dilihat pada gambar 5.45.

Layer (type)	Output Shape	Param #
<hr/>		
dense_1 (Dense)	(None, 100)	2500100
activation_1 (Activation)	(None, 100)	0
dense_2 (Dense)	(None, 10)	1010
activation_2 (Activation)	(None, 10)	0
<hr/>		
Total params: 2,501,110		
Trainable params: 2,501,110		
Non-trainable params: 0		
<hr/>		
None		

Figure 5.45: Fungsi Compile

9. Penjelasan dan jelaskan masing-masing parameter dari fungsi fit().

```
model.fit(train_input, train_labels, epochs=10, batch_size=32,
          validation_split=0.2)
```

Pada kode tersebut dapat dilihat bahwa pada model fit digunakan untuk melatih mesin dengan data training input dan training label. Epochs ini merupakan iterasi atau pengulangan berapa kali data tersebut akan dilakukan. Batch_size ini adalah jumlah file yang akan dilakukan pelatihan pada setiap 1 kali pengulangan. Sedangkan validation_split itu untuk menentukan persentase dari cross validation atau k-fold sebanyak 20% dari masing-masing data pengulangan. Hasilnya dapat dilihat pada gambar 5.46.

```
640/640 [=====] - 2s 3ms/step - loss: 0.9898 - acc: 0.6703 -
val_loss: 1.6596 - val_acc: 0.4500
Epoch 4/10
640/640 [=====] - 2s 3ms/step - loss: 0.7642 - acc: 0.7688 -
val_loss: 1.5874 - val_acc: 0.4625
Epoch 5/10
640/640 [=====] - 2s 3ms/step - loss: 0.6307 - acc: 0.8141 -
val_loss: 1.5605 - val_acc: 0.4437
Epoch 6/10
640/640 [=====] - 2s 3ms/step - loss: 0.4229 - acc: 0.9109 -
val_loss: 1.5566 - val_acc: 0.5000
Epoch 7/10
640/640 [=====] - 2s 3ms/step - loss: 0.3339 - acc: 0.9516 -
val_loss: 1.6867 - val_acc: 0.4750
Epoch 8/10
640/640 [=====] - 2s 3ms/step - loss: 0.2515 - acc: 0.9719 -
val_loss: 1.6458 - val_acc: 0.4813
Epoch 9/10
640/640 [=====] - 2s 3ms/step - loss: 0.1881 - acc: 0.9828 -
val_loss: 1.5771 - val_acc: 0.4813
Epoch 10/10
640/640 [=====] - 2s 3ms/step - loss: 0.1470 - acc: 0.9984 -
val_loss: 1.5798 - val_acc: 0.4813
Out[18]: <keras.callbacks.History at 0x25a2985f390>
```

Figure 5.46: Fungsi Fit

10. Mempraktekkan dan jelaskan masing-masing parameter dari fungsi evaluate().

```
loss, acc = model.evaluate(test_input, test_labels, batch_size=32)
```

Pada kode diatas maka dapat dilihat bahwa dengan menggunakan test input dan test label dilakukan evaluasi atau proses menemukan model terbaik yang mewakili data dan seberapa baik model yang dipilih akan dijalankan kedepannya. Hasilnya dapat dilihat pada gambar 5.47.

11. Praktekkan dan jelaskan masing-masing parameter dari fungsi predict().

Name	Type	Size	Value
acc	float64	1	0.48
loaded	float32	(1000, 25010)	[-0.40208789 -0.42075999 -0.51786171 ... 0. ... 0. ...]
model	float32	(1000, 25000)	[-0.81999071 -0.81014303 -0.75184401 ... -0.0181839 ... 0 ...]
pred	float32	(1000, 1)	genres/jazz/jazz.00062.au
test_input	float32	(1000, 10)	[1. 0. 0. ... 0. 0. 0.] [1. 0. 0. ... 0. 0. 0.]
loss	float64	1	1.5005969190597535

Figure 5.47: Fungsi Evaluate

```
model.predict(test_input[:1])
```

Pada kode diatas akan melakukan testing satu data lagu. file yang di jalankan tersebut termasuk ke dalam genre apa, hasilnya bisa dilihat pada gambar tersebut presentase yang paling besar yakni genre hiphop. Maka lagu tersebut termasuk ke dalam genre hiphop dengan perbandingan presentase hasil prediksi. Hasilnya dapat dilihat pada gambar 5.48.

```
In [21]: model.predict(test_input[:1])
Out[21]:
array([[1.4394021e-02, 1.3317568e-06, 1.1783892e-02, 8.4673949e-03,
       9.1628902e-02, 1.5559867e-02, 8.3826762e-06, 3.7754746e-03,
       8.2906008e-01, 2.5320653e-02]], dtype=float32)
```

Figure 5.48: Fungsi Predict

5.2.3 Penanganan Eror

1. ScreenShoot Error dapat dilihat pada gambar 5.49.

2. Tuliskan kode eror dan jenis errornya.

```
ModuleNotFoundError: No Module named 'keras.models'
```

3. Solusi pemecahan masalah error tersebut.

```
# Cara penanganannya adalah install modul keras
# dengan perintah seperti berikut
```

```
#dengan pip
pip install keras
```

```
In [1]: import librosa
...: import librosa.feature
...: import librosa.display
...: import glob
...: import numpy as np
...: import matplotlib.pyplot as plt
...: from keras.models import Sequential
...: from keras.layers import Dense, Activation
...: from keras.utils.np_utils import to_categorical
Traceback (most recent call last):

File "<ipython-input-1-736e292567e0>", line 7, in <module>
    from keras.models import Sequential

ModuleNotFoundError: No module named 'keras.models'
```

Figure 5.49: Eror

```
#dengan conda
conda install keras
```

5.3 Jesron Marudut Hatuan/1164077

5.3.1 Teori

1. Penjelasan mengapa file teks harus di lakukan tokenizer.

Tokenizer merupakan cara untuk membuat vektor dari teks. Dan mengapa harus dilakukan Tokenizer pada file teks? itu karena dengan memfungsikan Tokenizer, teks dapat divektorkan. Sehingga teks yang telah telah divektorkan tersebut dapat terbaca pada Machine Learning. Proses Tokenizer itu hanya memenggal kata-kata yang ada didalam suatu frasa atau kalimat yang terdapat didalam suatu text dataset. Ilustrasi gambar dapat dilihat pada gambar 5.50.

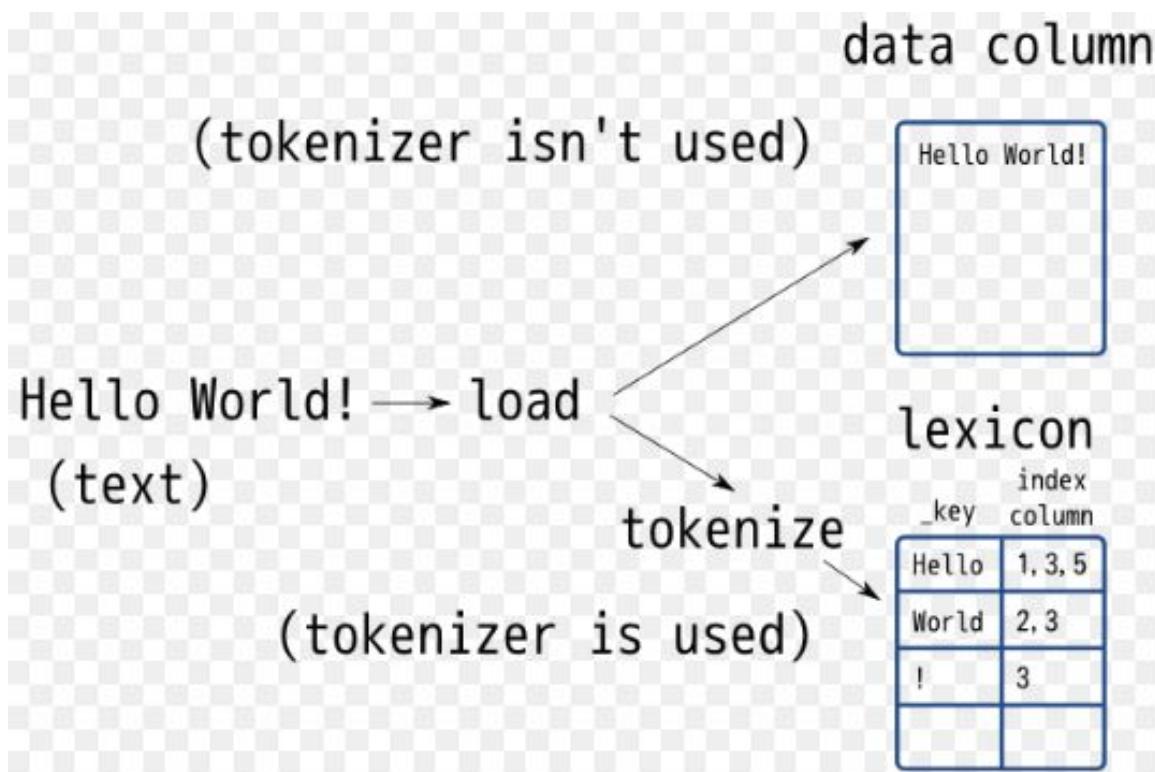


Figure 5.50: Tokenizer

2. Penjelasan konsep dasar K Fold Cross Validation pada dataset komentar Youtube pada kode listing 5.1.

Listing 5.1: K Fold Cross Validation

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

StartifiedKFold berisikan presentasi sampel untuk setiap kelas, dimana dalam ilustrasi ini sampel dibagi menjadi 5 dalam setiap class nya. Kemudian sampel tadi akan dimasukan kedalam class dari dataset youtube tadi. Contoh dapat dilihat pada gambar 5.51.

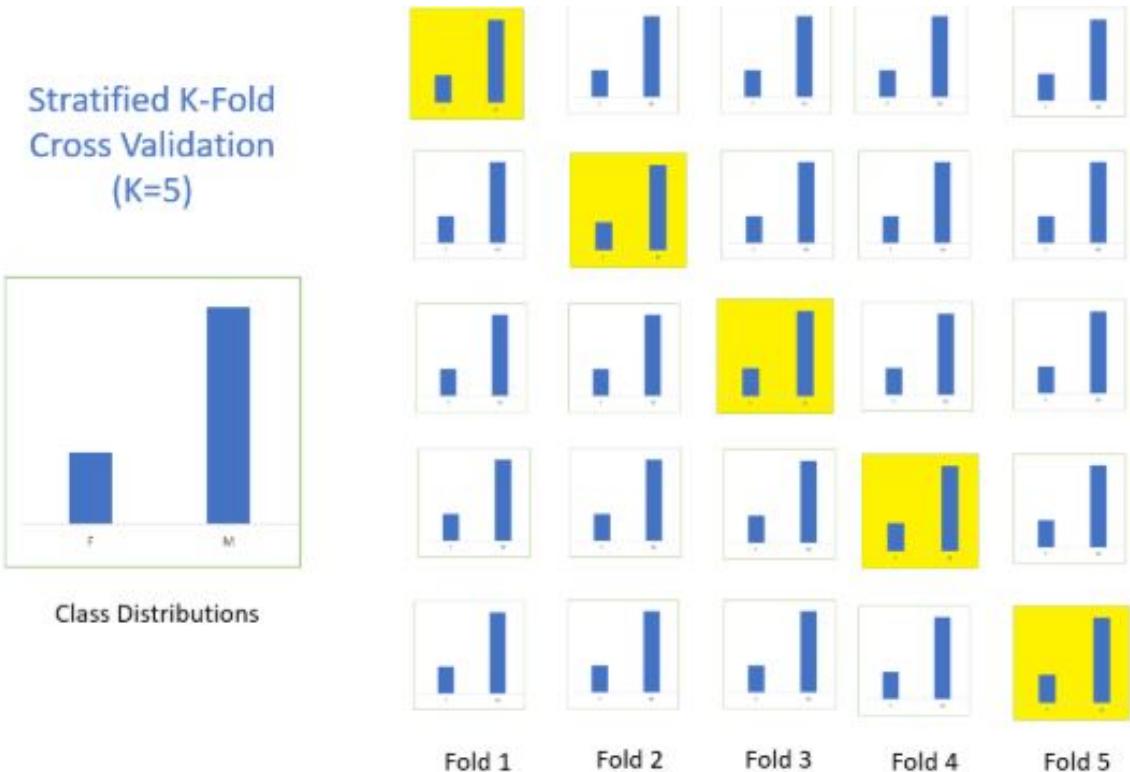


Figure 5.51: StartifiedKFold

3. Penjelasan maksud kode program for train, test in splits.

Maksudnya adalah 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 4 baju dengan model yang berbeda. Kemudian kita bagikan kedua anak, tentunya setiap anak yang menerima baju tidak memiliki baju yang sama modelnya.

4. Penjeasan maksud dari kode program $train_content = d['CONTENT'].iloc[train_idx]$ dan $test_content = d['CONTENT'].iloc[test_idx]$.

Maksudnya yaitu mengambil data pada kolom atau index CONTENT yang merupakan bagian dari train_idx dan test_idx. Ilustrasinya, ketika data telah diubah menjadi train dan test maka kita dapat memilihnya untuk ditampilkan pada kolom yang diinginkan.

5. Penjelasan maksud dari fungsi `tokenizer = Tokenizer(num_words=2000)` dan `tokenizer.fit_on_texts(train_content)`.

Dimana variabel tokenizer akan melakukan vektorisasi kata menggunakan fungsi Tokenizer yang dimana jumlah kata yang ingin diubah kedalam bentuk token adalah 2000 kata. Dan untuk `tokenizer.fit_on_texts(train_content)` maksudnya kita akan melakukan fit tokenizer hanya untuk dat trainnya saja tidak dengan data test nya untuk kolom CONTENT. Ilustrasinya, Jadi, jika Anda memberikannya sesuatu seperti, "Kucing itu duduk di atas tikar." Ini akan membuat kamus s.t. `word_index ["the"] = 0`; `word_index ["cat"] = 1` itu adalah kata -*i* kamus indeks sehingga setiap kata mendapat nilai integer yang unik.

6. Penjelasan 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')`.

Maksudnya yaitu untuk variabel `d_train_inputs` akan melakukan tokenizer dari bentuk teks ke matrix dari data `train_content` dengan mode matriksnya yaitu tfidf begitu juga dengan variabel `d_test_inputs` untuk data test. Contoh dapat dilihat pada gambar 5.52.

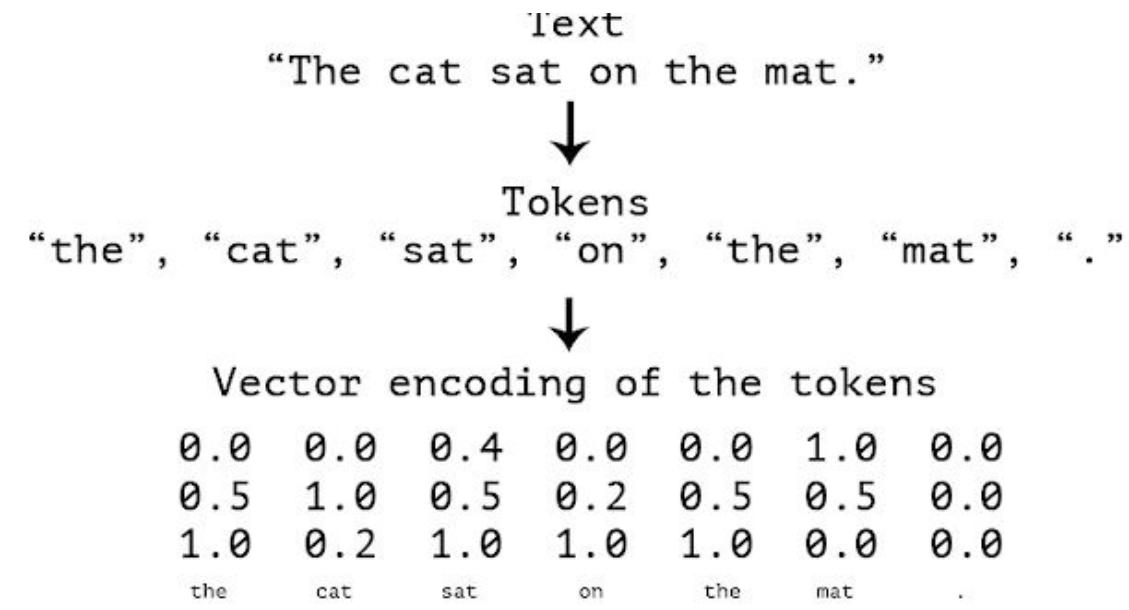


Figure 5.52: Fungsi Tokenizer

7. Penjelasan maksud dari fungsi `d_train_inputs = d_train_inputs/np.amax(np.absolute(d_train_inputs))` dan `d_test_inputs = d_test_inputs/np.amax(np.absolute(d_test_inputs))`.

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. Contoh dapat dilihat pada gambar 5.53.

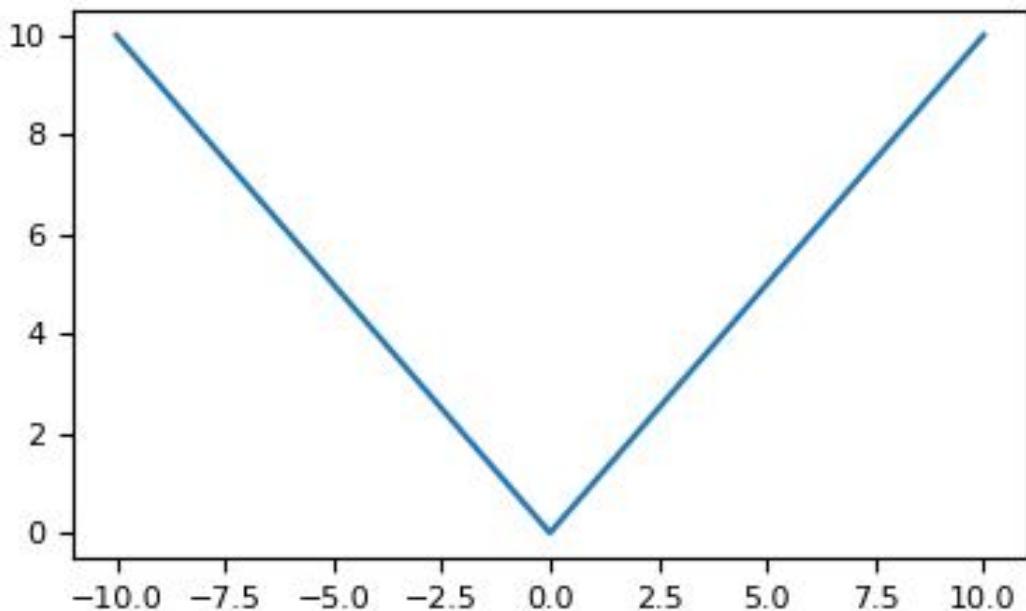


Figure 5.53: Matrix TFID

8. Penjelasan maksud fungsi dari `d_train_outputs = np_utils.to_categorical(d['CLASS'].iloc[train]` dan `d_test_outputs = np_utils.to_categorical(d['CLASS'].iloc[test_idx])` dalam kode program.

maksud dari fungsi tersebut yaitu untuk merubah nilai vektor yang ada pada atribut class menjadi bentuk matrix dengan pengurutan berdasarkan data index training dan testing. Contoh dapat dilihat pada gambar 5.54

9. Penjelasan maksud dari fungsi di listing 5.2. Gambarkan ilustrasi Neural Network nya dari model kode tersebut.

Listing 5.2: Neural Network

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

```
# Consider an array of 5 labels out of a set of 3 classes {0, 1, 2}:
> labels
array([0, 2, 1, 2, 0])
# `to_categorical` converts this into a matrix with as many
# columns as there are classes. The number of rows
# stays the same.
> to_categorical(labels)
array([[ 1.,  0.,  0.],
       [ 0.,  0.,  1.],
       [ 0.,  1.,  0.],
       [ 0.,  0.,  1.],
       [ 1.,  0.,  0.]], dtype=float32)

>>> type(df.iloc[0])
<class 'pandas.core.series.Series'>
>>> df.iloc[0]
a    1
b    2
c    3
d    4
Name: 0, dtype: int64
```

Figure 5.54: Matrix Training dan Testing

```
model.add(Dense(2))
model.add(Activation('softmax'))
```

10. Penjelasan maksud dari fungsi di listing 5.3 dengan parameter tersebut.

Listing 5.3: Model Compile Metric

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

11. Penjelasan mengenai apa itu Deep Learning.

Deep Learning adalah salah satu cabang dari Machine Learning atau dapat dikatakan bagian keluarga yang lebih luas dari method machine learning berdasarkan pada representasi data pembelajaran dan memiliki konsep serupa, tapi dilakukan dengan metode yang lebih cerdas. Deep Learning menggunakan Deep Neural Network dalam melakukan suatu penyelesaian suatu masalah yang terjadi pada Machine Learning.

12. Penjelasan apa itu Deep Neural dan bedanya dengan Deep Learning.

- Deep Neural Network atau DNN adalah sebuah algoritma yang berbasis neural network yang bisa digunakan untuk mengambil sebuah keputusan.
- Dan 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. Penjelasan dengan ilustrasi gambar buatan sendiri(langkah per langkah) bagaimana perhitungan algoritma konvolusi dengan ukuran stride (NPM mod3+1) x (NPM mod3+1) yang terdapat max pooling.

Konvolusi pada gambar dilakukan dalam image processing untuk menerapkan operator yang memiliki nilai output dari piksel gambar yang berasal dari kombinasi linear nilai input piksel, semakin nilai piksel tersebut maka kualitas gambar bisa semakin bagus. Contoh ilustrasi gambar dapat dilihat pada gambar 5.55 dan 5.56.

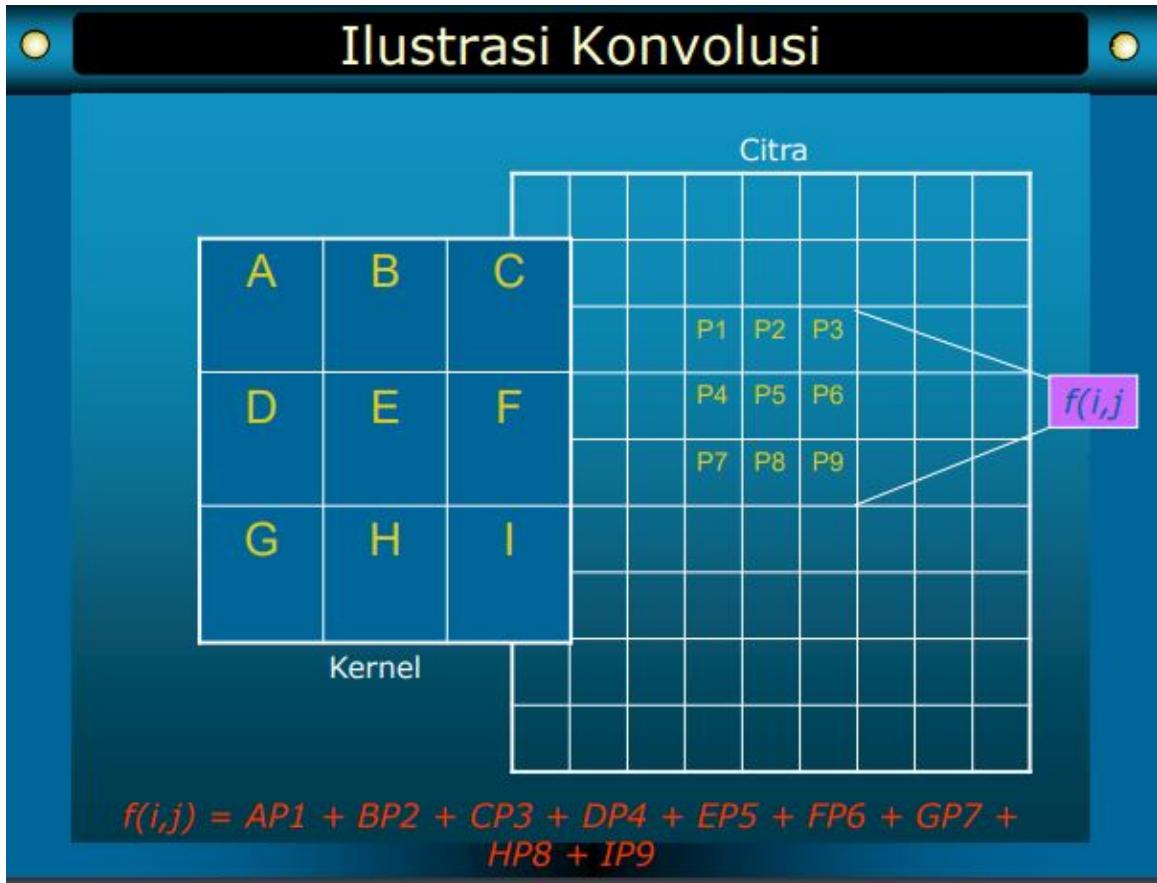


Figure 5.55: Konvolusi

Npm saya adalah 1164062 dan hasil dari $(NPM \bmod 3+1)=3$, maka saya menggunakan matrik kernel berukuran 3×3 . Misal $f(x,y)$ yang digunakan berukuran 4×4 dan kernel atau mask berukuran 3×3 dengan data masing-masing sebagai berikut:

$$f(x,y) = \begin{matrix} 2 & 4 & 6 & 8 \\ 1 & 3 & 5 & 7 \\ 1 & 2 & 3 & 4 \\ 4 & 3 & 2 & 1 \end{matrix} \quad g(f,y) = \begin{matrix} 1 & 2 & 1 \\ 2 & 1 & 2 \\ 3 & 2 & 3 \end{matrix}$$

Penyelesaian pada algoritma perhitungan konvolusi antara $f(x,y)$ dan $g(x,y)$ yaitu $f(x,y)xg(x,y)$. cara menghitungnya dari ujung kiri $f(x,y)$ ke $g(x,y)$ jika sudah geser pada kernel yang berbeda selanjutnya seperti berikut ini:
 1. $(2 \times 1) + (4 \times 2) + (6 \times 1) + (1 \times 2) + (3 \times 1) + (5 \times 2) + (1 \times 3) + (2 \times 2) + (3 \times 3) = 47$
 2. $(4 \times 1) + (6 \times 2) + (8 \times 1) + (3 \times 2) + (5 \times 1) + (7 \times 2) + (2 \times 3) + (3 \times 2) + (4 \times 3) = 73$
 3. $(1 \times 1) + (3 \times 2) + (5 \times 1) + (1 \times 2) + (2 \times 1) + (3 \times 2) + (4 \times 3) + (3 \times 2) + (2 \times 3) = 46$
 4. $(3 \times 1) + (5 \times 2) + (7 \times 1) + (2 \times 2) + (3 \times 1) + (4 \times 2) + (3 \times 3) + (2 \times 2) + (1 \times 3) = 51$

Hasil dari perhitungan diatas menghasilkan matrik kernel dengan 2×2 seperti berikut ini:

$$\begin{vmatrix} | \\ 47 & 73 \\ 46 & 51 \end{vmatrix}$$

Figure 5.56: Algoritma Perhitungan Konvolusi

5.3.2 Praktek Program

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

Berikut ini kode program yang digunakan :

Listing 5.4: 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

Sehingga dari kode program tersebut bila dijalankan, maka menghasilkan seperti pada gambar 5.57.

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

Figure 5.57: In[1]

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

Berikut ini kode program yang digunakan :

Listing 5.5: Kode Program 2

```
imgs = []
classes = []
with open('Chapter04/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 values
# so we rescale each pixel value to be in the range 0.0 - 1.0
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.

Sehingga dari kode program tersebut bila dijalankan, maka menghasilkan seperti pada gambar 5.58.

Name	Type	Size	Value
classes	list	168233	['A', 'A', ...]
i	int	1	168234
img	float32	(32, 32, 3)	[[[1. 1. 1.] [1. 1. 1.]
imgs	list	168233	[('hasy-data/v2-00000.png', 'A', Numpy array), ('hasy-data/v2-00001.pn ...]
row	list	4	['hasy-data/v2-168232.png', '1400', '\guillemotleft', '16925']

Figure 5.58: In[2]

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

Berikut adalah kode program yang digunakan :

Listing 5.6: 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

Sehingga dari kode program tersebut bila dijalankan, maka menghasilkan seperti pada gambar 5.59.

split_idx	int	1	134586
test	list	33647	[('hasy-data/v2-58330.png', '\int', Numpy array), ('hasy-data/v2-11911 ...
train	list	134586	[('hasy-data/v2-44778.png', '\xi', Numpy array), ('hasy-data/v2-67449. ...

Figure 5.59: In[3]

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

Berikut adalah kode program yang digunakan :

Listing 5.7: 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 mengoleksi 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.

Sehingga dari kode program tersebut bila dijalankan, maka menghasilkan seperti pada gambar 5.60.

test_input	float32	(33647, 32, 32, 3)	[[[1. 1. 1.] [1. 1. 1.]
test_output	str608	(33647,)	ndarray object of numpy module
train_input	float32	(134586, 32, 32, 3)	('hasy-data/v2-44778.png', '\xi', Numpy array), ('hasy-data/v2-67449. ...
train_output	str608	(134586,)	[[[1. 1. 1.] [1. 1. 1.]
			ndarray object of numpy module

Figure 5.60: In[4]

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

Berikut ini kode program yang digunakan :

Listing 5.8: 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].

Sehingga dari kode program tersebut bila dijalankan, maka menghasilkan seperti pada gambar 5.61.

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

Figure 5.61: In[5]

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

Berikut ini kode program yang digunakan :

Listing 5.9: 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.

Sehingga dari kode program tersebut bila dijalankan, maka menghasilkan seperti pada gambar 5.62.

integer_encoded	int64	(168233,)	[15 15 15 ... 143 143 143]
-----------------	-------	-----------	-----------------------------

Figure 5.62: In[6]

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

Berikut ini kode program yang digunakan :

Listing 5.10: 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 :

- Variabel onehot_encoder akan memanggil fungsi OneHotEncoder dimana tidak berisikan matriks sparse.
- Pada variabel integer_encoded akan diubah bentuknya dimana setiap nilai integer akan direpresentasikan sebagai vektor binari dengan nilai 0 kecuali index dari integer tersebut ditandai dengan 1.
- Melakukan fit untuk one hot encoder kedalam integer_encoder.

Sehingga dari kode program tersebut bila dijalankan, maka menghasilkan seperti pada gambar 5.63.

```
In [8]: onehot_encoder = OneHotEncoder(sparse=False)
....: integer_encoded = integer_encoded.reshape(len(integer_encoded), 1)
....: onehot_encoder.fit(integer_encoded)
C:\Users\HUDA\Anaconda3\lib\site-packages\sklearn\preprocessing\_encoders.py:368:
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[8]:
OneHotEncoder(categorical_features=None, categories=None,
               dtype=<class 'numpy.float64'>, handle_unknown='error',
               n_values=None, sparse=False)
```

Figure 5.63: In[7]

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

Berikut ini kode program yang digunakan :

Listing 5.11: Kode Program 8

```
train_output_int = label_encoder.transform(train_output)
train_output = onehot_encoder.transform(train_output_int.reshape(len(
test_output_int = label_encoder.transform(test_output)
test_output = onehot_encoder.transform(test_output_int.reshape(len(
num_classes = len(label_encoder.classes_))
print("Number of classes: %d" % num_classes)
```

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

- Variabel train_output_int akan mengubah data dari train_output menjadi LabelEncoder
- Dimana pada train_output setelah diubah labelnya menjadi integer dilakukan one hot encoding diambil dari train_output_int dan menggunakan .reshape untuk memberikan bentuk baru ke array tanpa mengubah datanya dengan keterangan jika index dari integer tersebut ditandai dengan 1 dan sisanya yang bukan nol.
- Variabel test_output_int akan mengubah data dari test_output menjadi LabelEncoder
- Dimana pada train_output setelah diubah labelnya menjadi integer dilakukan one hot encoding diambil dari test_output_int dan menggunakan .reshape untuk memberikan bentuk baru ke array tanpa mengubah datanya dengan keterangan jika index dari integer tersebut ditandai dengan 1 dan sisanya yang bukan nol.
- Variabel num_classes akan menampilkan jumlah data dari classes yang telah dilakukan label encoder
- Menampilkan tulisan "Number of classes : %d" dimana mengembalikan nilai integer dari num_classes.

Sehingga dari kode program tersebut bila dijalankan, maka menghasilkan seperti pada gambar 5.64.



Figure 5.64: In[8]

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

Berikut ini kode program yang digunakan :

Listing 5.12: 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 :

- Impor Sequential dari model pada librari Keras.
- Impor Dense, Dropout, Flatten dari modul Layers pada librari Keras.
- Impor Conv2D, MaxPooling2D dari modul Layers pada librari Keras.

Sehingga dari kode program tersebut bila dijalankan, maka menghasilkan seperti pada gambar 5.65.

```
In [10]: from keras.models import Sequential
....: from keras.layers import Dense, Dropout, Flatten
....: from keras.layers import Conv2D, MaxPooling2D
```

Figure 5.65: In[9]

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

Berikut ini kode program yang digunakan :

Listing 5.13: 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 :

- Melakukan pemodelan Sequential.
- Menambahkan Konvolusi 2D dengan 32 filter konvolusi masing-masing berukuran 3x3 dengan algoritam activation relu dengan data dari train_input mulai dari baris nol.
- Menambahkan Max Pooling dengan matriks 2x2.

- Dilakukan lagi penambahan Konvolusi 2D dengan 32 filter konvolusi masing-masing berukuran 3x3 dengan algoritam activation relu.
- Menambahkan lagi Max Pooling dengan matriks 2x2.
- Mendefinisikan inputan dengan 1024 neuron dan menggunakan algoritma tanh untuk activationnya.
- Dropout terdiri dari pengaturan secara acak tingkat pecahan unit input ke 0 pada setiap pembaruan selama waktu pelatihan, yang membantu mencegah overfitting sebesar 50% .
- Untuk output layer menggunakan data dari variabel num_classes dengan fungsi activationnya softmax.
- Mengonfigurasi proses pembelajaran, yang dilakukan melalui metode compile, sebelum melatih suatu model.
- Menampilkan atau mencetak representasi ringkasan model yang telah dibuat.

Sehingga dari kode program tersebut bila dijalankan, maka menghasilkan seperti pada gambar 5.66.

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

Berikut ini kode program yang digunakan :

Listing 5.14: Kode Program 11

```
import keras.callbacks
tensorboard = keras.callbacks.TensorBoard(log_dir='./logs/mnist-sty
```

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

- Impor Modul Callbacks dari Librari Keras.
- Variabel callback mendefinisikan Callback ini untuk menulis log untuk TensorBoard, yang memungkinkan Anda untuk memvisualisasikan grafik dinamis dari pelatihan dan metrik pengujian Anda, serta histogram aktivasi untuk berbagai lapisan dalam model Anda.

Sehingga dari kode program tersebut bila dijalankan, maka menghasilkan seperti pada gambar 5.67.

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

Berikut adalah kode program yang digunakan :

Layer (type)	Output Shape	Param #
<hr/>		
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
<hr/>		
Total params: 1,569,041		
Trainable params: 1,569,041		
Non-trainable params: 0		
<hr/>		
None		

Figure 5.66: In[10]

```
In [12]: import keras.callbacks
...: tensorboard = keras.callbacks.TensorBoard(log_dir='./logs/mnist-style')
```

Figure 5.67: In[11]

Listing 5.15: 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 :

- Melakukan fit model dengan 32 ukuran subset dari sampel pelatihan Anda
- Epoch sebanyak 10 kali
- Verbose=2 maksudnya menampilkan nomor dari epoch yang sedang berjalan atau yang sudah dijalankan.
- Validasi split sebanyak 20% sebagai fraksi data pelatihan untuk digunakan sebagai data validasi.
- Menggunakan TensorBoard sebagai callback untuk diterapkan selama pelatihan dan validasi.
- Variabel score mengembalikan nilai evaluate untuk menampilkan data loss dan data accuracy dari test
- Menampilkan data loss dengan menghitung jumlah kemunculan nol .
- Menampilkan data accuracy dengan menghitung jumlah kemunculan 1.

Sehingga dari kode program tersebut bila dijalankan, maka menghasilkan seperti pada gambar 5.68.

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

Berikut ini kode program yang digunakan :

Listing 5.16: Kode Program 13

```
import time

results = []
for conv2d_count in [1, 2]:
    for dense_size in [128, 256, 512, 1024, 2048]:
```

```

Train on 107668 samples, validate on 26918 samples
Epoch 1/10
- 693s - loss: 1.5615 - acc: 0.6233 - val_loss: 0.9945 - val_acc: 0.7260
Epoch 2/10
- 360s - loss: 0.9810 - acc: 0.7283 - val_loss: 0.8854 - val_acc: 0.7521
Epoch 3/10
- 355s - loss: 0.8676 - acc: 0.7516 - val_loss: 0.8852 - val_acc: 0.7550
Epoch 4/10
- 351s - loss: 0.7921 - acc: 0.7670 - val_loss: 0.8286 - val_acc: 0.7717
Epoch 5/10
- 350s - loss: 0.7472 - acc: 0.7770 - val_loss: 0.8424 - val_acc: 0.7687
Epoch 6/10
- 352s - loss: 0.7054 - acc: 0.7854 - val_loss: 0.8333 - val_acc: 0.7687
Epoch 7/10
- 351s - loss: 0.6716 - acc: 0.7929 - val_loss: 0.8297 - val_acc: 0.7679
Epoch 8/10
- 352s - loss: 0.6432 - acc: 0.7987 - val_loss: 0.8535 - val_acc: 0.7706
Epoch 9/10
- 368s - loss: 0.6197 - acc: 0.8047 - val_loss: 0.8778 - val_acc: 0.7635
Epoch 10/10
- 374s - loss: 0.5991 - acc: 0.8094 - val_loss: 0.8634 - val_acc: 0.7684
Test loss: 0.8603214174495762
Test accuracy: 0.767527565614718

```

Figure 5.68: In[12]

```

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'))

model.compile(loss='categorical_crossentropy', optimizer='adam')

log_dir = './logs/conv2d_%d-dense_%d-dropout_%.2f' % (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=[t
score = model.evaluate(test_input, test_output, verbose
end = time.time()
elapsed = end - start
print("Conv2D count: %d, Dense size: %d, Dropout: %.2f "
results.append((conv2d_count, dense_size, dropout, scor

```

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

- impor modul time dari python anaconda
- Variabel result berisikan array kosong.
- Menggunakan convolution 2D yang dimana akan memiliki 1 atau 2 layer.
- Mendefinisikan dense_size dengan ukuran 128, 256, 512, 1024, 2048
- Mendefinsikan drop_out dengan 0, 25%, 50%, dan 75%
- Melakukan pemodelan Sequential
- Jika ini adalah layer pertama, kita perlu memasukkan bentuk input.
- Kalau tidak kita hanya akan menambahkan layer.
- Kemudian, setelah menambahkan layer konvolusi, kita akan melakukan hal yang sama dengan max pooling.
- Lalu, kita akan meratakan atau flatten dan menambahkan dense size ukuran apa pun yang berasal dari dense_size. Dimana akan selalu menggunakan algoritma tanh
- Jika dropout digunakan, kita akan menambahkan layer dropout. Menyebut dropout ini berarti, katakanlah 50%, bahwa setiap kali ia memperbarui bobot setelah setiap batch, ada peluang 50% untuk setiap bobot yang tidak akan diperbarui
- menempatkan ini di antara dua lapisan padat untuk dihidupkan dari melindunginya dari overfitting.
- Lapisan terakhir akan selalu menjadi jumlah kelas karena itu harus, dan menggunakan softmax. Itu dikompilasi dengan cara yang sama.
- Atur direktori log yang berbeda untuk TensorBoard sehingga dapat membedakan konfigurasi yang berbeda.
- Variabel start akan memanggil modul time atau waktu
- Melakukan fit atau compile

- MELakukan scoring dengan .evaluate yang akan menampilkan data loss dan accuracy dari model
- end merupakan variabel untuk melihat waktu akhir pada saat pemodelan berhasil dilakukan.
- Menampilkan hasil dari run skrip diatas

Sehingga dari kode program tersebut bila dijalankan, maka menghasilkan seperti pada gambar 5.69.

```
2f, Accuracy: %.2f, Time: %d 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: 1.13, Accuracy: 0.74, Time: 1572
sec
Conv2D count: 1, Dense size: 128, Dropout: 0.25 - Loss: 0.91, Accuracy: 0.76, Time: 1632
sec
Conv2D count: 1, Dense size: 128, Dropout: 0.50 - Loss: 0.80, Accuracy: 0.78, Time: 1662
sec
Conv2D count: 1, Dense size: 128, Dropout: 0.75 - Loss: 0.80, Accuracy: 0.78, Time: 1735
sec
Conv2D count: 1, Dense size: 256, Dropout: 0.00 - Loss: 1.28, Accuracy: 0.74, Time: 2153
sec
Conv2D count: 1, Dense size: 256, Dropout: 0.25 - Loss: 1.08, Accuracy: 0.76, Time: 2212
sec
Conv2D count: 1, Dense size: 256, Dropout: 0.50 - Loss: 0.91, Accuracy: 0.78, Time: 2184
sec
Conv2D count: 1, Dense size: 256, Dropout: 0.75 - Loss: 0.78, Accuracy: 0.78, Time: 2184
sec
```

Figure 5.69: In[13]

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

Berikut ini kode program yang digunakan :

Listing 5.17: 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 :

- Melakukan pemodelan Sequential
- Untuk layer pertama, Menambahkan Convolutio 2D dengan dmensi 32, dan ukuran matriks 3x3 dengan function aktivasi yang digunakan yaitu relu dan menampilkan input_shape
- Dilakukan Max Pooling 2D dengan ukuran matriks 2x2
- Untuk layer kedua, melakukan Convolusi lagi dengan kriteria yang sama tanpa menambahkan input, ini dilakukan untuk mendapatkan data yang terbaik
- Flatten digunakan ntuk meratakan inputan
- Menambahkan dense input sebanyak 128 neuron dengan menggunakan function aktivasi tanh.
- Dropout sebanyak 50% untuk menghindari overfitting
- Menambahkan dense pada model untuk output dimana layer ini akan menjadi jumlah dari class yang ada.
- Mengcompile model yang didefinisikan diatas
- Menampilkan ringkasan dari pemodelan yang dilakukan

Sehingga dari kode program tersebut bila dijalankan, maka menghasilkan seperti pada gambar 5.70.

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

Berikut ini kode program yang digunakan :

Listing 5.18: 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 :

- Melakukan fit dengan join data train dan test agar dapat dilakukan pelatihan untuk jaringan pada semua data yang dimiliki.

Sehingga dari kode program tersebut bila dijalankan, maka menghasilkan seperti pada gambar 5.71.

Layer (type)	Output Shape	Param #
<hr/>		
conv2d_12 (Conv2D)	(None, 30, 30, 32)	896
max_pooling2d_12 (MaxPooling)	(None, 15, 15, 32)	0
conv2d_13 (Conv2D)	(None, 13, 13, 32)	9248
max_pooling2d_13 (MaxPooling)	(None, 6, 6, 32)	0
flatten_11 (Flatten)	(None, 1152)	0
dense_21 (Dense)	(None, 128)	147584
dropout_8 (Dropout)	(None, 128)	0
dense_22 (Dense)	(None, 369)	47601
<hr/>		
Total params: 205,329		
Trainable params: 205,329		
Non-trainable params: 0		
<hr/>		
None		

Figure 5.70: In[14]

```
In [15]: model.fit(np.concatenate((train_input, test_input)),
...:                 np.concatenate((train_output, test_output)),
...:                 batch_size=32, epochs=10, verbose=2)
Epoch 1/10
- 267s - loss: 1.7763 - acc: 0.5881
Epoch 2/10
- 249s - loss: 1.0767 - acc: 0.7061
Epoch 3/10
- 248s - loss: 0.9666 - acc: 0.7297
Epoch 4/10
- 249s - loss: 0.9096 - acc: 0.7411
Epoch 5/10
- 248s - loss: 0.8699 - acc: 0.7518
Epoch 6/10
- 253s - loss: 0.8428 - acc: 0.7551
Epoch 7/10
- 250s - loss: 0.8211 - acc: 0.7608
Epoch 8/10
- 247s - loss: 0.8050 - acc: 0.7639
Epoch 9/10
- 252s - loss: 0.7880 - acc: 0.7679
Epoch 10/10
- 251s - loss: 0.7751 - acc: 0.7697
Out[15]: <keras.callbacks.History at 0x22f13d79a20>
```

Figure 5.71: In[15]

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

Berikut ini kode program yang digunakan :

Listing 5.19: Kode Program 16

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

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

- Menyimpan atau save model yang telah di latih dengan nama mathsymbols.model

Sehingga dari kode program tersebut bila dijalankan, maka menghasilkan seperti pada gambar 5.72.



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

└── mathsymbols.model 13/04/2019 03.45 MODEL File 2.443 KB

A screenshot of a Jupyter Notebook interface. The code cell In [16] contains the command `model.save("mathsymbols.model")`. Below the cell, a file named "mathsymbols.model" is shown with a download icon, indicating it has been saved. Metadata for the file includes the date (13/04/2019 03.45), type (MODEL File), and size (2.443 KB).

Figure 5.72: In[16]

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

Berikut ini kode program yang digunakan :

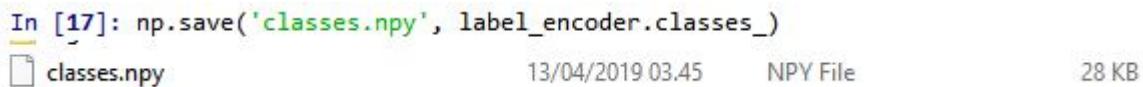
Listing 5.20: Kode Program 17

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

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

- Simpan label enkoder (untuk membalikkan one-hot encoder) dengan nama classes.npy

Sehingga dari kode program tersebut bila dijalankan, maka menghasilkan seperti pada gambar 5.73.



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

└── classes.npy 13/04/2019 03.45 NPY File 28 KB

A screenshot of a Jupyter Notebook interface. The code cell In [17] contains the command `np.save('classes.npy', label_encoder.classes_)`. Below the cell, a file named "classes.npy" is shown with a download icon, indicating it has been saved. Metadata for the file includes the date (13/04/2019 03.45), type (NPY File), and size (28 KB).

Figure 5.73: In[17]

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

Berikut ini kode program yang digunakan :

Listing 5.21: 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 :

- Impor models dari librari Keras
- Variabel model2 akan memanggil model yang telah disave tadi
- Menampilkan ringkasan dari hasil pemodelan

Sehingga dari kode program tersebut bila dijalankan, maka menghasilkan seperti pada gambar 5.74.

```
....: print(model2.summary())  


| Layer (type)                  | Output Shape       | Param # |
|-------------------------------|--------------------|---------|
| conv2d_12 (Conv2D)            | (None, 30, 30, 32) | 896     |
| max_pooling2d_12 (MaxPooling) | (None, 15, 15, 32) | 0       |
| conv2d_13 (Conv2D)            | (None, 13, 13, 32) | 9248    |
| max_pooling2d_13 (MaxPooling) | (None, 6, 6, 32)   | 0       |
| flatten_11 (Flatten)          | (None, 1152)       | 0       |
| dense_21 (Dense)              | (None, 128)        | 147584  |
| dropout_8 (Dropout)           | (None, 128)        | 0       |
| dense_22 (Dense)              | (None, 369)        | 47601   |
| Total params:                 | 205,329            |         |
| Trainable params:             | 205,329            |         |
| Non-trainable params:         | 0                  |         |
| None                          |                    |         |


```

Figure 5.74: In[18]

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

Berikut ini kode program yang digunakan :

Listing 5.22: 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 :

- Memanggil fungsi LabelEncoder
- Variabel label_encoder akan memanggil class yang disave sebelumnya.
- Function Predict akan mengubah gambar kedalam bentuk array
- Variabel prediction akan melakukan prediksi untuk model2 dengan reshape variabel newimg dengan bentukarray 4D.
- Variabel inverted akan mencari nilai tertinggi output dari hasil prediksi tadi
- Menampilkan hasil dari variabel prediction dan inverted

Sehingga dari kode program tersebut bila dijalankan, maka menghasilkan seperti pada gambar 5.75.

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

Berikut ini kode program yang digunakan :

Listing 5.23: Kode Program 20

```
predict("Chapter04/hasy-data/v2-00010.png")
predict("Chapter04/hasy-data/v2-00500.png")
predict("Chapter04/hasy-data/v2-00700.png")
```

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

```
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 5.75: In[19]

- Melakukan prediksi dari pelatihan dari gambar v2-00010.png
- Melakukan prediksi dari pelatihan dari gambar v2-00500.png
- Melakukan prediksi dari pelatihan dari gambar v2-00700.png

Sehingga dari kode program tersebut bila dijalankan, maka menghasilkan seperti pada gambar 5.76.

```
In [20]: predict("HASYv2/hasy-data/v2-00010.png")
....:
....: predict("HASYv2/hasy-data/v2-00500.png")
....:
....: predict("HASYv2/hasy-data/v2-00700.png")
Prediction: A, confidence: 0.85
Prediction: \pi, confidence: 0.81
Prediction: \alpha, confidence: 0.90
```

Figure 5.76: In[20]

5.3.3 Penanganan Eror

1. skrinsut error
2. Kode eror dan jenis errornya

Eror tersebut merupakan eror yang terjadi dan membuat kita tidak dapat mengakses dan menggunakan kernel atau konsol pada spyder.

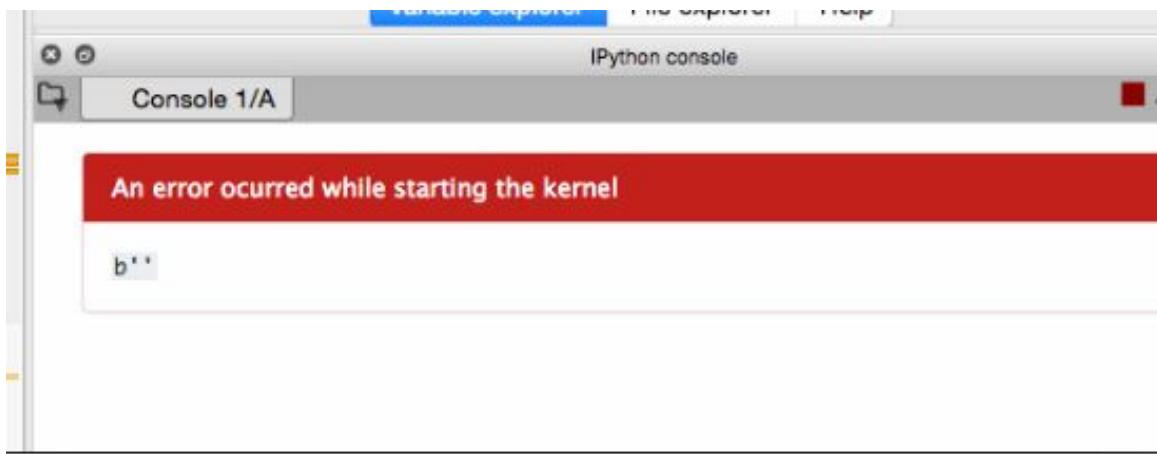


Figure 5.77: Eror

3. Solusi pemecahan masalah error tersebut

- Tutup spyder yang sedang dijalankan
- Kemudian buka kembali spyder
- maka eror pada kernel tersebut akan hilang dan kembali stabil agar bisa mengakses konsol tersebut.

Chapter 6

Discussion

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

Chapter 7

Discussion

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

Chapter 8

Discussion

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

Chapter 9

Discussion

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

Chapter 10

Discussion

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

Chapter 11

Discussion

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

Chapter 12

Discussion

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

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)
TOTAL			36	
Catatan : Nilai minimal untuk diterima 25				

Figure A.2: form nilai bagian 2.

Appendix B

FAQ

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

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

M : Pa saya sudah cari jurnal terindeks scopus tapi ga nemu. D : Kamu punya mata de? coba dicolok dulu. Kamu udah lakuin apa aja? tolong di list laporan 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.