

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 andi muh aslam/1164064	2
1.4.1 sejarah dan perkembangan kecerdasan buatan	2
1.5 Instalasi	4
1.5.1 instalasi Library Scikit dari Anaconda	4
1.5.1.1 Mencoba Loading an example dataset	4
1.6 Aip Suprapto Munari/1164063	5
1.6.1 Teori	5
1.6.2 Instalasi	7
1.6.2.1 Instalasi Library Scikit dari Anaconda	7
1.6.2.2 Mencoba Loading an example Dataset	8
1.6.2.3 Learning and Predicting	8
1.6.3 Mencoba Model Persistance, menjelaskan maksud dari tulisan tersebut dan mengartikan per baris	9
1.6.4 Mencoba Conventions, menjelaskan maksud dari tulisan tersebut dan mengartikan per baris	10
1.7 Penanganan Error	15
1.7.0.1 Model Presistence	20
1.7.0.2 Conventions	21
1.7.1 Penanganan eror	24
1.7.1.1 ScreenShoot Eror	24
1.7.1.2 Tuliskan Kode Eror dan Jenis Erornya	24
1.7.1.3 Solusi Pemecahan Masalah Error	25

2	Related Works	26
2.1	Aip Suprapto Munari/1164063	26
2.1.1	Teori	26
2.1.2	Binary Classification	26
2.1.3	Supervised Learning, Unsupervised Learning, Dan Classtering	26
2.1.4	Evaluasi Dan Akurasi	29
2.1.5	Confusion Matrix	29
2.1.6	Cara Kerja K-Fold Cross Validation	30
2.1.7	Decision Tree	31
2.1.8	Gain Dan Entropi	31
2.2	Aip Suprapto Munari/1164063	33
2.2.1	Scikit-learn	33
2.2.2	Penanganan Error	38
2.3	Andi Aslam/1164064	38
2.3.1	Binary Clasification beserta gambar	38
2.3.2	supervised learning dan unsupervised learning dan clustering dengan ilustrasi gambar	38
2.3.3	evaluasi dan akurasi dari buku dan disertai ilustrasi contoh dengan gambar	40
2.3.4	bagaimana cara membuat dan membaca confusion matrix, buat confusion matrix	40
2.3.5	bagaimana K-fold cross validation bekerja dengan gambar ilustrasi	41
2.3.6	decision tree dengan gambar ilustrasi	42
2.3.7	Information Gain dan entropi dengan gambar ilustrasi	42
2.4	Andiaslam/1164064	43
2.4.1	Scikit-learn	43
2.4.2	Praktek Penanganan Error	48
2.5	Same Topics	48
2.5.1	Topic 1	48
2.5.2	Topic 2	48
2.6	Same Method	49
2.6.1	Method 1	49
2.6.2	Method 2	49

3 Methods	50
3.1 The data	50
3.2 Method 1	50
3.3 Method 2	50
3.4 Aip Suprapto Munari/1164063	50
3.4.1 Teori	50
3.4.2 Praktek	56
3.4.3 Penanganan Error	65
3.5 Andi Muhammad Aslam/1164064	67
3.5.1 Praktek	70
4 Experiment and Result	83
4.1 Experiment	83
4.2 Result	83
4.3 Aip Suprapto Munari/1164063	83
4.3.1 Teori	83
4.4 BAGIAN PRAKTEK	85
4.5 Aip Suprapto Munari /1164063	85
4.5.1 Aplikasi Sederhana Menggunakan Pandas	85
4.5.2 Memecah DataFrame Menjadi 2 Dataframe	87
4.5.3 Vektorisasi Dan Klasifikasi Dari Data Youtube05-Shakira Dengan Decision Tree	87
4.5.4 Vektorisasi Dan Klasifikasi Dari Data Youtube05-Shakira Dengan SVM	88
4.5.5 Vektorisasi Dan Klasifikasi Dari Data Youtube05-Shakira Dengan Decision Tree 2	88
4.5.6 Plotting Confusion Matrix	89
4.5.7 Menjalankan Program Cross Validation	90
4.5.8 Program Pengamatan Komponen Informasi	91
4.6 Penanganan Error	91
4.6.1 Error Index	92
4.7 Andi Muhammad Aslam/1164064	92
4.7.1 Teori	92
4.8 Andi Muhammad Aslam/1164064	94
4.8.1 Praktek Program	94
4.8.2 Penanganan Eror	96

5 Conclusion	103
5.1 Conclusion of Problems	103
5.2 Conclusion of Method	103
5.3 Conclusion of Experiment	103
5.4 Conclusion of Result	103
5.5 Andi Muh Aslam/1164064	103
5.5.1 Teori	103
5.5.2 Praktek Program	106
5.6 Aip Suprapto Munari/1164063	112
5.6.1 Teori	112
5.7 PRAKTEK PROGRAM	116
5.8 Aip Suprapto Munari/1164063	116
5.8.1 Mencoba Dataset	116
5.8.1.1 Vektor	116
5.8.1.2 Similariti	118
5.8.2 Extract Words dan PermuteSentences	119
5.8.2.1 Extract Words	119
5.8.2.2 PermuteSentences	119
5.8.2.3 Library Gensim TaggedDocument Dan Doc2Vec . . .	119
5.8.2.4 Menambahkan Data Training	120
5.8.2.5 Pengocokan Dan Pembersihan Data	120
5.8.2.6 Mengapa Model Harus Di Save Dan Temporari Training Harus Dihapus	121
5.8.2.7 Infercode	121
5.8.2.8 Cosinesimilarity	121
5.8.2.9 Praktek Score Dari Cross Validation	122
5.9 Penanganan Error	122
5.9.1 Error	122
6 Discussion	124
6.1 Aip Suprapto Munari/1164063	124
6.1.1 Teori	124
6.1.2 Praktek	129
6.1.3 Penanganan Error	140
6.2 Andi Muh Aslam	141
6.2.1 Teori	141

6.2.2	Praktek	145
6.2.3	Penanganan Eror	148
7	Discussion	149
7.1	Andi Muhammad Aslam/1164064	149
7.1.1	Teori	149
7.1.2	Praktek Program	156
7.1.3	Penanganan Eror	178
7.2	Aip Suprapto Munari-1164063	179
7.2.1	Teori	179
7.2.2	Praktek	185
7.2.3	Penanganan Error	204
8	Discussion	206
9	Discussion	207
10	Discussion	208
11	Discussion	209
12	Discussion	210
13	Discussion	211
14	Discussion	212
A	Form Penilaian Jurnal	213
B	FAQ	216
	Bibliography	218

List of Figures

1.1	Langkah 1 instalasi anaconda	4
1.2	Langkah 2 instalasi anaconda	4
1.3	Langkah 3 instalasi anaconda	5
1.4	Langkah 4 instalasi anaconda.	5
1.5	Langkah 1 dataset.	6
1.6	Langkah 2 dataset.	6
1.7	Langkah 3 dataset.	7
1.8	Langkah 4 dataset.	7
1.9	Langkah 5 dataset.	8
1.10	Hasil Pengujian Classifier	9
1.11	Hasil Pengujian Classifier	10
1.12	Pickle Pada Python	10
1.13	Pengujian Classifier Pickle	10
1.14	Penggunaan Joblib	11
1.15	Deklarasi Numpy	11
1.16	Contoh Type Casting	11
1.17	Menggunakan FitTransform	12
1.18	Regresi Yang Dilempar	12
1.19	Refitting dan Memperbarui Parameter	13
1.20	MultiClass Classifier	14
1.21	MultiClass Classifier biner 2D	14
1.22	MultiLabel Classifier	14
1.23	Eror Import	15
1.24	Instal Library Joblib	15
1.25	Berhasil Import Library Joblib	16
1.26	Download Anaconda.	16
1.27	Langkah pertama instalasi anaconda.	16
1.28	Langkah kedua instalasi anaconda.	17

1.29 Langkah ketiga instalasi anaconda.	17
1.30 Langkah terakhir instalasi anaconda.	18
1.31 Langkah pertama instalasi scikit pada CMD.	18
1.32 Langkah ketiga instalasi conda scikit pada CMD.	18
1.33 Langkah kedua pilih y.	19
1.34 Langkah cek version yang diinstall.	19
1.35 Hasil Tampilan 1.	19
1.36 Hasil Tampilan 2.	19
1.37 Hasil Tampilan 3.	19
1.38 Hasil Tampilan Error.	24
1.39 Hasil Tampilan Uji coba perintah joblib.	25
 2.1 Binary Classification.	27
2.2 Supervised Learning.	28
2.3 Unsupervised Learning.	28
2.4 Clustering.	29
2.5 Evaluasi Dan Akurasi.	30
2.6 K-Fold Cross Validation.	31
2.7 Decision Tree.	32
2.8 Gain Dan Entropi.	32
2.9 Gambar pertama	33
2.10 Gambar kedua	33
2.11 Gambar Ketiga	34
2.12 Gambar Keempat	34
2.13 Gambar Kelima	34
2.14 Gambar Keenam	35
2.15 Gambar Ketujuh	35
2.16 Gambar Kedelapan	35
2.17 Gambar Kesembilan	36
2.18 Gambar Kesepuluh	36
2.19 Gambar Kesebelas	37
2.20 Gambar Keduabelas	37
2.21 Gambar Ketigabelas	38
2.22 Binary Classification	39
2.23 Supervised Learning	40
2.24 Evaluasi dan Akurasi	41

2.25	K-fold cross validation	42
2.26	Decision Tree	43
2.27	Entropi	43
2.28	Hasil Code 1	44
2.29	Hasil Code 2	44
2.30	Hasil Code 3	44
2.31	Hasil Code 4	45
2.32	Hasil Code 5	45
2.33	Hasil Code 6	45
2.34	Hasil Code 7	46
2.35	Hasil Code 8	46
2.36	Hasil Code 9	46
2.37	Hasil Code 10	47
2.38	Hasil Code 11	47
2.39	Hasil Code 12	48
2.40	HASIL YANG MASIH ERROR	48
3.1	Random Forest	51
3.2	(b)	51
3.3	(c)	52
3.4	(d)	52
3.5	(e)	52
3.6	(h)	53
3.7	Confussion Matrik	55
3.8	Voting Random forest	56
3.9	Aplikasi Pandas	56
3.10	Hasil Pandas	57
3.11	Aplikasi Numpy	57
3.12	Hasil Numpy	57
3.13	Aplikasi Matplotlib	57
3.14	Hasil Matplotlib	58
3.15	Membaca Data File	58
3.16	Melihat Data Sebagian	59
3.17	Melihat Jumlah Data	59
3.18	Mengubah menjadi kolom	59
3.19	Lihat sebagian data awal	59

3.20 Melihat jumlah data	59
3.21 Mengelompokkan burung	60
3.22 Melalukan pivot	60
3.23 Melihat data awal imgid	60
3.24 Melihat jumlah data imgid	60
3.25 Data ciri label dari join	61
3.26 Mengubah menjadi kolom	61
3.27 Melihat isi data frame	61
3.28 Membagi data	61
3.29 Kelas Random Forest	61
3.30 Membangun Random forest	62
3.31 Melihat hasil	62
3.32 Lihat hasil score	62
3.33 Memetakan ke confusion matrix	62
3.34 Melihat hasil	62
3.35 Melakukan Plot	63
3.36 Plotting nama data	63
3.37 Melakukan perintah plot	63
3.38 Klasifikasi menggunakan decision tree	64
3.39 Klasifikasi menggunakan SVM	64
3.40 Pengecekan cross validation random forest	64
3.41 Pengecekan cross validation decision tree	65
3.42 Pengecekan cross validation SVM	65
3.43 Pengamatan Komponen	65
3.44 Plot informasi	66
3.45 Skrinsut Error	66
3.46 Penyelesaian	66
3.47 Hasil	67
3.48 Random Forest.	68
3.49 Tabel Confusion Matriks	69
3.50 Voting	70
3.51 Pandas	71
3.52 Numpy	71
3.53 Matplotlib	72
3.54 Gambar1	72
3.55 Gambar2	73

3.56 Gambar3	73
3.57 Gambar 4	73
3.58 Gambar 5	74
3.59 Gambar 6	74
3.60 Gambar 7	75
3.61 Gambar 8	75
3.62 Gambar 9	76
3.63 Gambar 10	76
3.64 Gambar 11	77
3.65 Gambar 12	77
3.66 Gambar 13	78
3.67 Gambar 15	79
3.68 Gambar 16	79
3.69 Gambar 17	79
3.70 Gambar 18	79
3.71 Gambar 20	79
3.72 Gambar 21	80
3.73 Gambar 23	80
3.74 SVM	80
3.75 Decission Tree	80
3.76 Cross Validation 1	81
3.77 Cross Validation 2	81
3.78 Cross Validation 3	81
3.79 Program Pengamatan Komponen Informasi 1	82
3.80 Program Pengamatan Komponen Informasi 2	82
3.81 Error	82
4.1 Aip-Klasifikasi teks	83
4.2 Aip-Klasifikasi bunga	84
4.3 Aip-Teknik YouTube	84
4.4 Aip-Bag of Word	85
4.5 Aip-TF IDF	85
4.6 Dataset Original Aip	86
4.7 Dataset Dummy Aip	87
4.8 Split DataFrame Aip	87
4.9 Dataset Youtube05-Shakira Aip	88

4.10	Dataset Youtube05-Shakira Aip	88
4.11	Dataset Youtube05-Shakira SVM Aip	88
4.12	Dataset Youtube05-Shakira Aip	89
4.13	Confusion Matrix Aip	90
4.14	Cross Validation Aip	90
4.15	Hasil Cross Validation Aip	91
4.16	Program Komponen Informas Aip	91
4.17	Error Key Aip	92
4.18	Error Key Aip	92
4.19	Error Key Aip	92
4.20	Error Key Aip	93
4.21	Klasifikasi teks	98
4.22	Klasifikasi bunga	99
4.23	Teknik YouTube	99
4.24	Bag of Word	99
4.25	TF-IDF	100
4.26	Data Dummy 500 Data	100
4.27	Membagi 2 Dataframe	100
4.28	Vektorisasi dan Klasifikasi Data	100
4.29	Data Content	101
4.30	DataFrame Kata-kata Pada Content	101
4.31	Klasifikasi SVM Dari Data Vektorisasi	101
4.32	Klasifikasi Decision Tree Dari Data Vektorisasi	101
4.33	Plot Confusion Matrix Menggunakan Matplotlib	101
4.34	Program Cross Validation Pada Data Vektorisasi	102
4.35	Program Pengamatan Komponen Informasi	102
4.36	Eror matplotlib.pyplot	102
5.1	Ilustrasi Soal No.1	104
5.2	Ilustrasi Soal No. 2	104
5.3	Ilustrasi Soal No. 3	105
5.4	Ilustrasi Soal No. 4	105
5.5	Ilustrasi Soal No. 5	106
5.6	Ilustrasi Soal No. 6	107
5.7	Love	107
5.8	Faith	107

5.9 Fall	108
5.10 Sick	108
5.11 Clear	108
5.12 Shine	109
5.13 Bag	109
5.14 Car	109
5.15 Wash	110
5.16 Motor	110
5.17 Cycle	110
5.18 Similariti Pada Kata Motor dan Cycle	111
5.19 Similariti Pada Kata Wash dan Motor	111
5.20 Extract_Words	111
5.21 Permute Sentences	112
5.22 Vektorisasi Kata Aip	113
5.23 Google Dataset Aip	113
5.24 Vektorisasi Kata Aip	114
5.25 Vektorisasi Dokumen Aip	114
5.26 Mean Aip	115
5.27 Standar Deviasi Aip	115
5.28 Skip-Gram Aip	116
5.29 Vektor Love Aip	116
5.30 Vektor Faith Aip	116
5.31 Vektor Fall Aip	117
5.32 Vektor Sick Aip	117
5.33 Vektor Clear Aip	117
5.34 Vektor Shine Aip	117
5.35 Vektor Bag Aip	117
5.36 Vektor Car Aip	117
5.37 Vektor Wash Aip	118
5.38 Vektor cycle Aip	118
5.39 Similariti Aip	118
5.40 Extract Words Aip	119
5.41 PermuteSentencesi Aip	119
5.42 Library Gensim TaggedDocument Dan Doc2Vac Aip	119
5.43 Data 1 - Aip	120
5.44 Data 2 - Aip	120

5.45	Data 2 - Aip	120
5.46	Pengocokan dan Pembersihan Data - Aip	120
5.47	Pengocokan dan Pembersihan Data - Aip	120
5.48	Model Disave dan Temporari Train Hapus - Aip	121
5.49	Model Disave dan Temporari Train Hapus - Aip	121
5.50	Infercode - Aip	121
5.51	Cosinesimilarity - Aip	121
5.52	Cosinesimilarity - Aip	121
5.53	Cosinesimilarity - Aip	122
5.54	Cosinesimilarity - Aip	122
5.55	Score Cross Validation - Aip	122
5.56	Score Cross Validation - Aip	122
5.57	Score Cross Validation - Aip	122
5.58	Score Cross Validation - Aip	123
5.59	Error Key Aip	123
6.1	MFCC - Aip	125
6.2	Konsep Dasar Neural Network - Aip	125
6.3	Konsep Pembobotan Neural Network - Aip	126
6.4	Konsep Fungsi Aktifasi - Aip	127
6.5	Plot MFCC - Aip	127
6.6	One-Hot Encoding - Aip	127
6.7	np.unique - Aip	128
6.8	to.categorical - Aip	128
6.9	Sequential - Aip	129
6.10	Hasil No 1 Aip	129
6.11	Kode Program No 2 Aip	130
6.12	Hasil No 2 Aip	131
6.13	Hasil No 3 Aip	132
6.14	Hasil No 4 Aip	133
6.15	Hasil No 5 Aip	134
6.16	Pemisahan Data Training Dan Dataset - Aip	137
6.17	Pemisahan Data Training Dan Dataset 2 - Aip	137
6.18	Fungsi Sequential - Aip	137
6.19	Fungsi Compile - Aip	138
6.20	Fungsi Fit-Aip	139

6.21	Fungsi Evaluate - Aip	139
6.22	Fungsi Predict - Aip	140
6.23	Error Aip	140
6.24	MFCC	141
6.25	Konsep Dasar Neural Network	141
6.26	Konsep Pembobotan	142
6.27	Konsep Aktivasi	143
6.28	Cara Membaca Hasil Plot MLCC	143
6.29	One Hot Encoding	144
6.30	NP Unique	144
6.31	To Categorical	144
6.32	Sequential	145
6.33	NO 1	146
6.34	Kode Program No 2	146
6.35	Hasil No 2	146
6.36	Hasil No 3	147
7.1	Tokenizer	150
7.2	KFold Cross validation	150
7.3	Ilustrasi for train, test in splits	150
7.4	Fungsi Tokenizer	152
7.5	Matrix TF IDF	153
7.6	Matrix TFID	153
7.7	Matrix Training dan Testing	154
7.8	Konvolusi	156
7.9	Algoritma Perhitungan Konvolusi	157
7.10	In[1]	157
7.11	In[2]	159
7.12	In[3]	160
7.13	In[4]	161
7.14	In[5]	162
7.15	In[6]	162
7.16	In[7]	163
7.17	In[8]	164
7.18	In[9]	165
7.19	In[10]	167

7.20 In[11]	167
7.21 In[12]	169
7.22 In[13]	171
7.23 In[14]	173
7.24 In[15]	174
7.25 In[16]	175
7.26 In[17]	175
7.27 In[18]	176
7.28 In[19]	177
7.29 In[20]	178
7.30 Eror	179
7.31 Tokenizer Aip	179
7.32 K-Fold Cross Validation Aip	180
7.33 No 3 Aip	181
7.34 No 5 Aip	181
7.35 No 6 Aip	182
7.36 No 7 Aip	183
7.37 No 8 Aip	183
7.38 Langkah Algoritma Konvolusi Berdasarkan NPM- Aip	186
7.39 Plagiarisme- Aip	186
7.40 In 1 Aip	186
7.41 In 2 Aip	187
7.42 In 3 Aip	189
7.43 In 4 Aip	189
7.44 In 5 Aip	190
7.45 In 6 Aip	191
7.46 In 7 Aip	191
7.47 In 8 Aip	192
7.48 In 9 Aip	193
7.49 In 10 Aip	194
7.50 In 11 Aip	195
7.51 In 12 Aip	196
7.52 In 13 Aip	198
7.53 In 14 Aip	200
7.54 In 15 Aip	201
7.55 In 16 Aip	201

7.56 In 17 Aip	202
7.57 In 18 Aip	203
7.58 In 19 Aip	203
7.59 In 20 Aip	204
7.60 Error Aip	205
A.1 Form nilai bagian 1.	214
A.2 form nilai bagian 2.	215

Chapter 1

Mengenal Kecerdasan Buatan dan Scikit-Learn

Buku umum yang digunakan adalah [4] dan untuk sebelum UTS menggunakan buku *Python Artificial Intelligence Projects for Beginners*[2]. Dengan praktek menggunakan python 3 dan editor anaconda dan library python scikit-learn. Tujuan pembelajaran pada pertemuan pertama antara lain:

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

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

1.1 Teori

Praktek teori penunjang yang dikerjakan :

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

1.2 Instalasi

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

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

1.3 Penanganan Error

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

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

1.4 andi muh aslam/1164064

1.4.1 sejarah dan perkembangan kecerdasan buatan

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

didefinikasikan sebagai kecerdasan yang ditinjukkan oleh suatu entitas buatan. Sistem seperti ini umumnya di anggap komputer. Kecerdasan diciptakan dan dimasukkan melakukan pekerjaan seperti yang dapat dilakukan manusia.

2. 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[3].
3. 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.
4. 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.
5. 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.
6. 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.
7. 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].

1.5 Instalasi

1.5.1 instalasi Library Scikit dari Anaconda

1. Download aplikasi Anaconda terlebih dahulu

```
Windows PowerShell
Copyright (C) Microsoft corporation. All rights reserved.

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
```

Figure 1.1: Langkah 1 instalasi anaconda..

2. Proceed install anaconda

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

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

Figure 1.2: Langkah 2 instalasi anaconda.

3. install scikit-learn

4. perintah print

1.5.1.1 Mencoba Loading an example dataset

1. Masuk Pyhton terlebih dahulu
2. from sklearn import datasets(pada baris ini merupakan sebuah perintah untuk mengimport sebuah datasets dari file sklearn).

```
PS C:\WINDOWS\system32> pip install -U scikit-learn
Collecting scikit-learn
  Downloading https://files.pythonhosted.org/packages/ee/c8/c89ebdc0d7dbba6e6fd222daabd257da3c28a96...
    100% |██████████| 4.3MB 463kB/s
Requirement not upgraded as not directly required: numpy>=1.8.2 in c:\programdata\anaconda3\lib\site...
Requirement not upgraded as not directly required: scipy>=0.13.3 in c:\programdata\anaconda3\lib\site...
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 1.3: Langkah 3 instalasi anaconda.

```
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 ('Andi')
Andi
```

Figure 1.4: Langkah 4 instalasi anaconda.

3. iris datasets.load_iris()(pada baris kedua ini dimana iris merupakan suatu variable yang berfungsi untuk mengambil data pada datasets dengan perintah .load_iris).
4. digits datasets.load_digits()(pada baris ketiga ini dimana digits merupakan suatu variable yang berfungsi untuk mengambil data pada datasets dengan perintah .load_digits)
5. print(digits.data)(pada baris keempat ini merupakan perintah yang berfungsi untuk memanggil atau menampilkan variable digits.data) jjjjjj HEAD

1.6 Aip Suprapto Munari/1164063

1.6.1 Teori

- (a) Definisi, sejarah, dan perkembangan kecerdasan buatan.

Definisi kecerdasan buatan adalah ilmu pengetahuan yang dapat membuat komputer untuk meniru kecerdasan manusia yang berhubungan dengan penangkapan, pemodelan, dan penyimpanan kecerdasan manusia dalam

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

Figure 1.5: Langkah 1 dataset.

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

Figure 1.6: Langkah 2 dataset.

sebuah sistem teknologi. Contohnya yaitu melakukan analisa penalaran untuk mengambil suatu kesimpulan atau penerjemahan atau keputusan dari satu bahasa satu ke bahasa lain.

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

- (b) 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 merupakan salah satu topik utama dalam data mining atau machine learning. Klasifikasi yaitu suatu pengelompokan data dimana data tersebut digunakan untuk 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.

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

Figure 1.7: Langkah 3 dataset.

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

Figure 1.8: Langkah 4 dataset.

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[?].

1.6.2 Instalasi

1.6.2.1 Instalasi Library Scikit dari Anaconda

- (a) Download aplikasi Anaconda terlebih dahulu. Lihat pada gambar 1.1
- (b) Install aplikasi Anaconda yang sudah di download tadi. Lihat pada gambar 1.2
- (c) Centang Keduanya lalu tekan tombol install. Lihat pada gambar 1.3
- (d) Setelah itu tunggu sampai proses instalasi selesai lalu jika sudah tekan tombol finish. Lihat pada gambar 1.4
- (e) Lalu buka command prompt anda dan tuliskan perintah berikut ini untuk mengecek apakah aplikasinya sudah terinstall. Lihat pada gambar 1.5
- (f) Kemudian ketikkan perinta pip install -U scikit-learn seperti gambar berikut. Lihat pada gambar 1.6
- (g) Lalu jika sudah ketikkan juga perintah conda install scikit-learn. Lihat pada gambar 1.7
- (h) dan setelah itu pilih y. Lihat pada gambar 1.8
- (i) Hasil version yang diinstall. Lihat pada gambar 1.9

```

>>> print(digits.data)
[[ 0.   0.0053...  0.   0.1) 0.]
 [ 0.   0.0053... 10.   0.   0.] ...
 [ 0.   0.0053... 16.   9.   0.] ...
 ...
 [ 0.   0.0053... 6.   0.   0.] ...
 [ 0.   0.0053... 12.   0.   0.] ...
 [ 0.   0.0053... 12.   1.   0.]]
>>>

```

Figure 1.9: Langkah 5 dataset.

1.6.2.2 Mencoba Loading an example Dataset

- from sklearn import datasets(pada baris ini merupakan sebuah perintah untuk mengimport class datasets dari packaged sklearn).
- iris = datasets.load_iris()(pada baris kedua ini dimana iris merupakan suatu estimator/parameter yang berfungsi untuk mengambil data pada item datasets.load_iris).
- digits = datasets.load_digits()(pada baris ketiga ini dimana digits merupakan suatu estimator/parameter yang berfungsi untuk mengambil data pada item datasets.load_digits).
- print(digits.data)(pada baris keempat ini merupakan perintah yang berfungsi untuk menampilkan estimator/parameter yang dipanggil pada item digits.data dan menampilkan outputannya) Lihat gambar 1.10.
- digits.target(barisan ini untuk mengambil target pada estimator/parameter digits dan menampilkan outputannya) Lihat gambar 1.11.
- digits.images[0](barisan ini untuk mengambil images[0] pada estimator/-parameter digits dan menampilkan outputannya) Lihat gambar 1.12.

=====

78cdb6514db9716f252ef2024a7df6097aace611

1.6.2.3 Learning and Predicting

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

- `clf = svm.SVC(gamma=0.001, C=100.)`(pada baris kedua ini clf sebagai estimator/parameter, svm.SVC sebagai class, gamma sebagai parameter untuk menetapkan nilai secara manual)
- `clf.fit(digits.data[:-1], digits.target[:-1])`(pada baris ketiga ini clf sebagai estimator/parameter, fit sebagai metode, digits.data sebagai item, `[:-1]` sebagai syntax pythonnya dan menampilkan outputannya) Lihat gambar 1.13.
- `clf.predict(digits.data[-1:])`

|||||| HEAD

1.6.3 Mencoba Model Persistance, menjelaskan maksud dari tulisan tersebut dan mengartikan per baris

1. Pada Python Shell ketikan ”from sklearn import svm” artinya akan mengimport sebuah Support Vector Machine(SVM) yang merupakan algoritma classification yang akan diambil dari Scikit-Learn.
2. Kemudian, lanjutkan dengan ”from sklearn import datasets” yang artinya akan mengambil package datasets dari Scikit-Learn.
3. ketikan, `clf = svm.SVC(gamma='scale')` berfungsi untuk mendeklarasikan suatu value yang bernama clf yang berisi gamma. Parameter gamma menentukan seberapa jauh pengaruh dari satu contoh training.
4. Ketikan, `X, y = iris.data, iris.target`, artinya X sebagai data iris, dan y merupakan larik target.
5. Ketikan, `clf.fit(X, y)` berfungsi untuk melakukan pengujian classifier. hasilnya seperti ini

```
SVC(C=1.0, cache_size=200, class_weight=None, coef0=0.0,
```

Figure 1.10: Hasil Pengujian Classifier

Dari gambar diatas dapat dijelaskan bahwa akan mengimport Pickle dari Python. Pickle digunakan untuk serialisasi dan de-serialisasi struktur objek Python. Objek apa pun dengan Python dapat di-Pickle sehingga dapat disimpan di disk.

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

Figure 1.11: Hasil Pengujian Classifier

kemudian menyimpan data objek ke file CLF sebelumnya dengan menggunakan function pickle.dumps(clf).

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

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

Figure 1.12: Pickle Pada Python

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

Figure 1.13: Pengujian Classifier Pickle

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

9. Dalam kasus khusus scikit-learn, mungkin lebih menarik untuk menggunakan joblib (dump dan load) untuk menggantikan Pickle, yang lebih efisien pada data besar tetapi hanya bisa di Pickle ke disk dan tidak ke string. untuk menggunakan Joblib pertama ketikan "from joblib import dump , load" yang artinya akan Merekonstruksi objek Python dari file yang sudah ada.

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

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

1. Import numpy as np, digunakan untuk mengimport Numpy sebagai np.
From sklearn import randomprojection artinya modul yang mengimplementasi

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

Figure 1.14: Penggunaan Joblib

tasikan cara sederhana dan efisien secara komputasi untuk mengurangi dimensi data dengan memperdagangkan sejumlah akurasi yang terkendali (sebagai vari- an tambahan) untuk waktu pemrosesan yang lebih cepat dan ukuran model yang lebih kecil.

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

Figure 1.15: Deklarasi Numpy

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

Figure 1.16: Contoh Type Casting

Pada gambar diatas dapat dijelaskan bahwa :

`rng = np.random.RandomState(0)`, digunakan untuk menginisialisasikan ran- dom number generator.

`X = rng.rand(10, 2000)` artinya akan merandom value antara 10 sampai 2000.
`X = np.array(X, dtype='float32')` Array numpy terdiri dari buffer memori "men- tah" yang diartikan sebagai array melalui "views". Anda dapat menganggap semua array numpy sebagai tampilan. Mendeklarasikan X sebagai float32.

3. Dalam contoh ini, X adalah float32, yang dilemparkan ke float64 oleh fittrans- form (X).
4. Target regresi dilemparkan ke float64 dan target klasifikasi dipertahankan.

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

Figure 1.17: Menggunakan FitTransform

list(clf.predict(irisdata[:3])), akan memprediksi 3 data dari iris.

clf.fit(irisdata, iristargetnames[iristarget]) menguji classifier dengan ada targetnya yaitu irisnya sendiri.

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

Di sini, prediksi pertama () mengembalikan array integer, karena iristarget

```
C:\Users\Asus-PC>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
>>> from sklearn.svm import SVC
>>> iris = datasets.load_iris()
>>> clf = SVC(gamma='scale')
>>> clf.fit(iris.data, iris.target)
SVC(C=1.0, cache_size=200, class_weight=None, coef0=0.0,
    decision_function_shape='ovr', degree=3, gamma='scale', kernel='rbf',
    max_iter=-1, probability=False, random_state=None, shrinking=True,
    tol=0.001, verbose=False)
>>> list(clf.predict(iris.data[:3]))
[0, 0, 0]
>>> clf.fit(iris.data, iris.target_names[iris.target])
SVC(C=1.0, cache_size=200, class_weight=None, coef0=0.0,
    decision_function_shape='ovr', degree=3, gamma='scale', kernel='rbf',
    max_iter=-1, probability=False, random_state=None, shrinking=True,
    tol=0.001, verbose=False)
>>> list(clf.predict(iris.data[:3]))
['setosa', 'setosa', 'setosa']
```

Figure 1.18: Regresi Yang Dilempar

(array integer) yang digunakan sesuai. Prediksi kedua () mengembalikan array string, karena iristargetnames cocok.

5. Refitting dan Memperbarui Parameter

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

clf.setparams(kernel='linear').fit(X, y) mengubah kernel default menjadi linear

clf.set_params(kernel='rbf', gamma='scale').fit(X, y) Di sini, kernel default rbf pertama kali diubah menjadi linear melalui SVC.set_params () setelah estimator dibuat, dan diubah kembali ke rbf untuk mereparasi estimator dan membuat prediksi kedua.

```
>>> import numpy as np
>>> from sklearn.svm import SVC
>>> rng = np.random.RandomState(0)
>>> X = rng.rand(100, 10)
>>> y = rng.binomial(1, 0.5, 100)
>>> X_test = rng.rand(5,10)
>>> clf = SVC()
>>> clf.set_params(kernel='linear').fit(X,y)
SVC(C=1.0, cache_size=200, class_weight=None, coef0=0.0,
    decision_function_shape='ovr', degree=3, gamma='auto_deprecated',
    kernel='linear', max_iter=-1, probability=False, random_state=None,
    shrinking=True, tol=0.001, verbose=False)
>>> clf.predict(X_test)
array([1, 0, 1, 1, 0])
>>> clf.set_params(kernel='rbf' ,gamma='scale').fit(X,y)
SVC(C=1.0, cache_size=200, class_weight=None, coef0=0.0,
    decision_function_shape='ovr', degree=3, gamma='scale', kernel='rbf',
    max_iter=-1, probability=False, random_state=None, shrinking=True,
    tol=0.001, verbose=False)
>>> clf.predict(X_test)
array([1, 0, 1, 1, 0])
```

Figure 1.19: Refitting dan Memperbaharui Parameter

6. MultiClass VS MultiLabel Classifier

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

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

7. Di sini, classifier cocok () pada representasi label biner 2d dari y, menggunakan LabelBinarizer. Dalam hal ini predict () mengembalikan array 2d yang mewakili prediksi multilabel yang sesuai.

```

>>> 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, 11]
... y = [0, 0, 1, 1, 2]
      File "<stdin>", line 2
        y = [0, 0, 1, 1, 2]
          ^
SyntaxError: invalid syntax
>>> X = [[1, 2], [2, 4], [4, 5], [3, 2], [3, 11]]
>>> y = [0, 0, 1, 1, 2]
>>> classif = OneVsRestClassifier(estimator=SVC(gamma='scale',
... random_state=0))
>>> classif.fit(X, y).predict(X)
array([0, 0, 1, 1, 2])
>>>

```

Figure 1.20: MultiClass Classifier

```

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

```

Figure 1.21: MultiClass Classifier biner 2D

8. from sklearn.preprocessing import MultiLabelBinarizer , artinya Transformasi antara iterable dari iterables dan format multilabel.

Dalam hal ini, penggolongnya sesuai pada setiap instance yang diberi beberapa label. MultiLabelBinarizer digunakan untuk membuat binarize array 2d dari multilabel agar sesuai. Hasilnya, predict () mengembalikan array 2d dengan beberapa label yang diprediksi untuk setiap instance.

```

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

```

Figure 1.22: MultiLabel Classifier

1.7 Penanganan Error

HARI KEDUA

1. Berikut ini merupakan eror yang ditemui pada saat melakukan percobaan skrip.

```
NameError: name 'clf' is not defined
>> from joblib import dump, load
>>
```

Figure 1.23: Eror Import

2. Pada gambar eror diatas, kode erornya adalah "ImportError: No Module Named" artinya mengalami masalah saat mengimpor modul yang ditentukan.
3. Solusinya bisa dilakukan seperti berikut :
eror diatas terjadi dikarenakan Library Joblib belum terinstal pada PC. Maka dari itu sekarang kita harus menginstalnya dulu.
4. Buka CMD, kemudian ketikan "pip install joblib" tunggu sampai instalasi berhasil seperti gambar berikut.

```
C:\Users\Asus-PC>pip install joblib
Collecting joblib
  Downloading https://files.pythonhosted.org/packages/cd/c1/50a758e8247561e58cb8730...
    100% |██████████| 286kB 1.8MB/s
distributed 1.21.8 requires msgpack, which is not installed.
Installing collected packages: joblib
Successfully installed joblib-0.13.2
You are using pip version 10.0.1, however version 19.0.3 is available.
You should consider upgrading via the 'python -m pip install --upgrade pip' command
```

Figure 1.24: Instal Library Joblib

5. Apabila sudah terinstall, dapat dilakukan lagi import library joblib, maka akan berhasil seperti dibawah berikut

=====

```
C:\Users\Asus-PC>python
Python 3.6.5 |Anaconda, Inc.| (default, Mar 29 2018, 13:23:52) [MSC v.1900 32 bit (R
Type "help", "copyright", "credits" or "license" for more information.
>>> from joblib import dump, load
>>> dump(clf, 'filename.joblib')
```

Figure 1.25: Berhasil Import Library Joblib

Filename	Size	Last Modified	MD5
Anaconda2-2018.12-Linux-ppc64le.sh	289.7M	2018-12-21 13:14:33	d50ce6eb037f72edfe8f94f90d61aca6
Anaconda2-2018.12-Linux-x86.sh	518.6M	2018-12-21 13:13:15	7d26c7551af6802eb83ecd34282056d7
Anaconda2-2018.12-Linux-x86_64.sh	628.2M	2018-12-21 13:13:10	84f39388da2c747477cf14cb02721b93
Anaconda2-2018.12-MacOSX-x86_64.pkg	640.7M	2018-12-21 13:14:30	c2bfefef310714501a59fd58166e6393d
Anaconda2-2018.12-MacOSX-x86_64.sh	547.1M	2018-12-21 13:14:31	f4d8b10e9a754884fb96e68e0e0b276a
Anaconda2-2018.12-Windows-x86.exe	458.6M	2018-12-21 13:16:27	f123fda0ec8928bb7d55d1ca72c0d784
Anaconda2-2018.12-Windows-x86_64.exe	560.6M	2018-12-21 13:16:17	10ff4176a94fcff86e6253b0c82c782
Anaconda3-2018.12-Linux-ppc64le.sh	313.6M	2018-12-21 13:13:03	a775fb6d6c441b899ff2327bd9dadcd
Anaconda3-2018.12-Linux-x86.sh	542.7M	2018-12-21 13:13:14	4c9922d1547128b866c6b9cf750c03c7
Anaconda3-2018.12-Linux-x86_64.sh	652.5M	2018-12-21 13:13:06	c9af603d89656bc89680889ef1f92623

Figure 1.26: Download Anaconda.

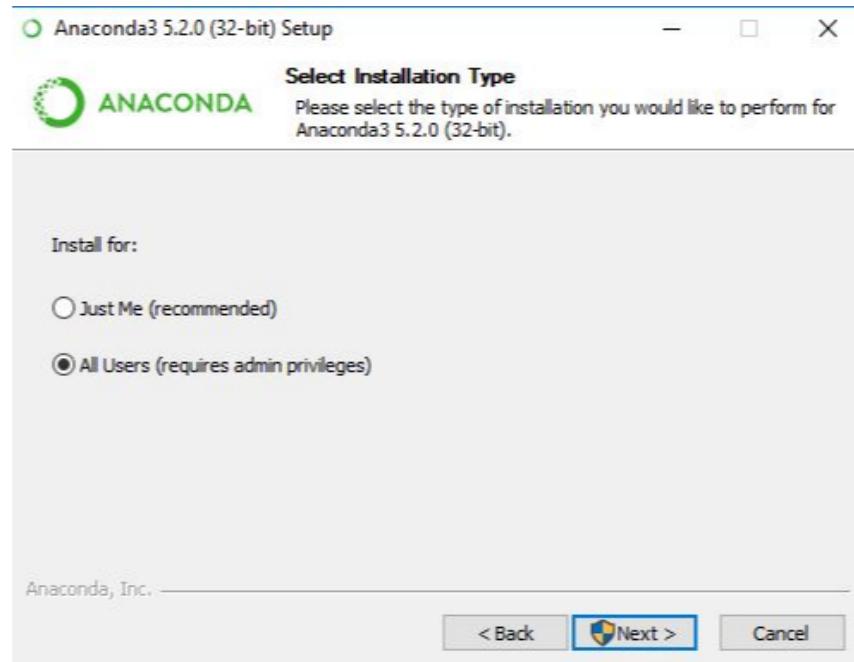


Figure 1.27: Langkah pertama instalasi anaconda.

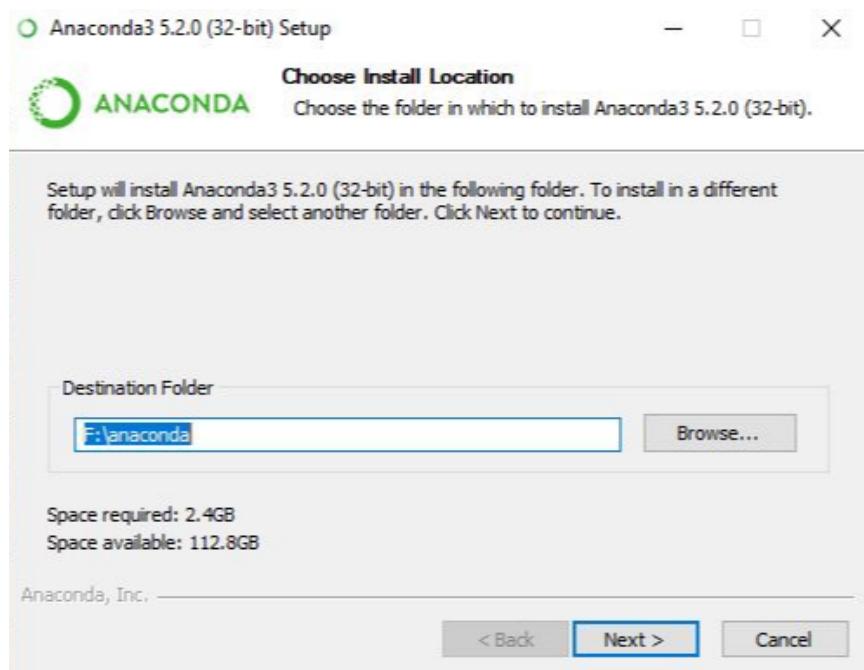


Figure 1.28: Langkah kedua instalasi anaconda.

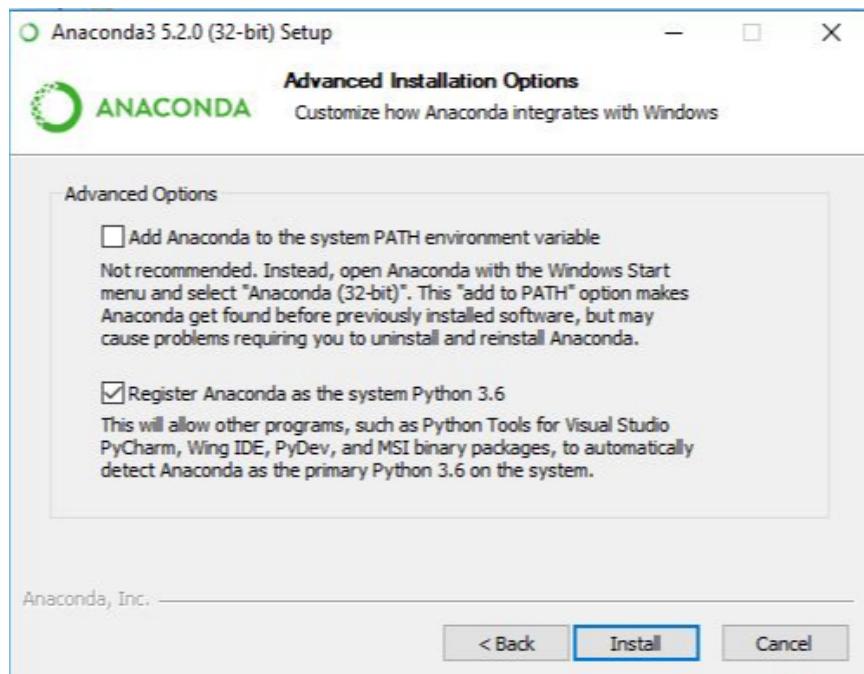


Figure 1.29: Langkah ketiga instalasi anaconda.

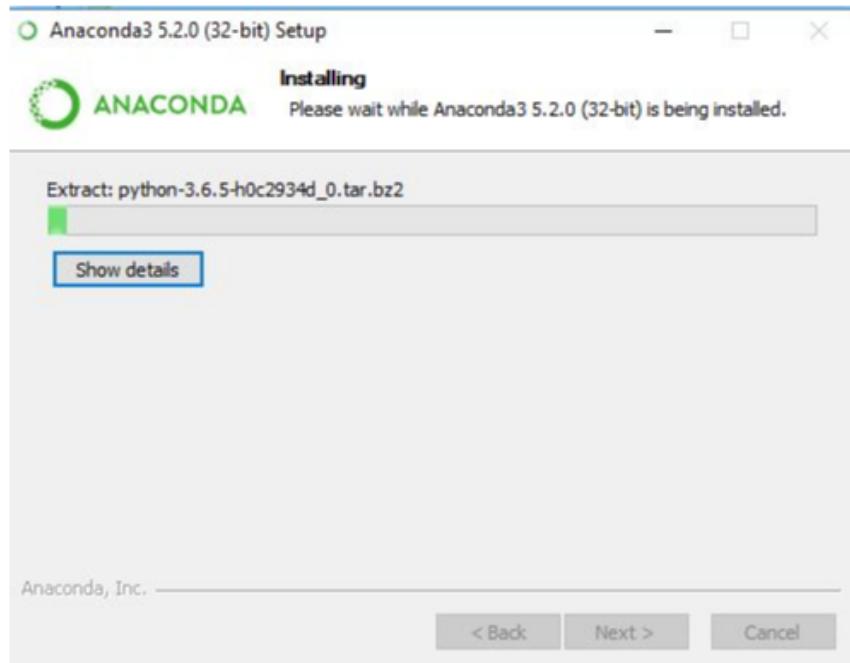


Figure 1.30: Langkah terakhir instalasi anaconda.

```
C:\Users\Asus-PC>pip install -U scikit-learn
Collecting scikit-learn
  Downloading https://files.pythonhosted.org/packages/ee/c8/c89ebdcdd7dbade6fd222daabd257da3c28a067dd7c352d427b2e1cef/scikit_learn-0.20.2-cp36-cp36m-win32.whl (4.3MB)
    100% |████████████████████████████████| 4.3MB 660kB/s
Requirement already up-to-date: numpy<1.8.2 in c:\programdata\anaconda3\lib\site-packages (from scikit-learn) (1.14.3)
Requirement already up-to-date: scipy<0.19.3 in c:\programdata\anaconda3\lib\site-packages (from scikit-learn) (1.1.0)
Requirement already up-to-date: joblib<0.12.0 in c:\programdata\anaconda3\lib\site-packages (from scikit-learn) (0.12.0)
Installing collected packages: scikit-learn
  Found existing installation: scikit-learn 0.19.1
  Uninstalling scikit-learn-0.19.1:
    Successfully uninstalled scikit-learn-0.19.1
Successfully installed scikit-learn-0.20.2
You are using pip version 10.0.1, however version 19.0.3 is available.
You should consider upgrading via the 'python -m pip install --upgrade pip' command.
```

Figure 1.31: Langkah pertama instalasi scikit pada CMD.

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

## Package Plan ##

environment location: C:\ProgramData\Anaconda3

added / updated specs:
- scikit-learn

The following packages will be downloaded:

  package                               | build
  conda-4.6.7                            | py36_0      1.7 MB

The following packages will be UPDATED:

  conda: 4.5.4-py36_0 --> 4.6.7-py36_0
```

Figure 1.32: Langkah ketiga instalasi conda scikit pada CMD.

```

Proceed ([y]/n)? y

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

```

Figure 1.33: Langkah kedua pilih y.

```

C:\Users\Asus-PC>conda --version
conda 4.6.7

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

C:\Users\Asus-PC>

```

Figure 1.34: Langkah cek version yang diinstall.

```

C:\Users\Asus-PC>python
Python 3.6.5 |Anaconda, Inc.| (default, Mar 29 2018, 13:23:52) [MSC v.1900 32 bit (Intel)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> from sklearn.metrics import confusion_matrix
>>> y_true = [2, 0, 2, 2, 0, 1]
>>> y_pred = [0, 0, 2, 2, 0, 2]
>>> confusion_matrix(y_true, y_pred)
array([[2, 0, 0],
       [0, 0, 1],
       [1, 0, 2]], dtype=int64)
>>> y_true = ["cow", "goat", "cow", "goat", "horse"]
>>> y_pred = ["goat", "goat", "cow", "goat", "cow"]
>>> confusion_matrix(y_true, y_pred, labels=["cow", "goat", "horse"])
array([[1, 1, 0],
       [0, 2, 0],
       [1, 0, 0]], dtype=int64)
>>> ab, fa, fb, aa = confusion_matrix([0, 1, 0, 1], [1, 1, 1, 0]).ravel()
>>> (ab, fa, fb, aa)
(0, 2, 1, 1)
>>>

```

Figure 1.35: Hasil Tampilan 1.

```

>>> y_true = ["cow", "goat", "cow", "goat", "horse"]
>>> y_pred = ["goat", "goat", "cow", "goat", "cow"]
>>> confusion_matrix(y_true, y_pred, labels=["cow", "goat", "horse"])
array([[1, 1, 0],
       [0, 2, 0],
       [1, 0, 0]], dtype=int64)

```

Figure 1.36: Hasil Tampilan 2.

```

>>> ab, fa, fb, aa = confusion_matrix([0, 1, 0, 1], [1, 1, 1, 0]).ravel()
>>> (ab, fa, fb, aa)
(0, 2, 1, 1)

```

Figure 1.37: Hasil Tampilan 3.

1.7.0.1 Model Persistence

- `from sklearn import svm`(pada baris ini merupakan sebuah perintah untuk mengimport class svm dari packaged sklearn).
- `from sklearn import datasets`(pada baris ini merupakan sebuah perintah untuk mengimport class datasets dari packaged sklearn).
- `clf = svm.SVC(gamma='scale')`
(pada baris ketiga ini clf sebagai estimator/parameter, svm.SVC sebagai class, gamma sebagai parameter untuk menetapkan nilai secara manual dengan nilai scale).
- `iris = datasets.load_iris()`
(pada baris keempat ini iris sebagai estimator/parameter, datasets.load_iris() sebagai item dari suatu nilai).
- `X, y = iris.data, iris.target`
(pada baris kelima ini X, y sebagai estimator/parameter, iris.data, iris.target sebagai item dari 2 nilai yang ada).
- `clf.fit(X, y)`
(pada baris keenam ini clf sebagai estimator/parameter dengan menggunakan metode fit untuk memanggil estimator X, y dengan outputannya)
- `import pickle`
(pickle merupakan sebuah class yang di import).
- `s = pickle.dumps(clf)`
(pada baris ini s sebagai estimator/parameter dengan pickle.dumps merupakan suatu nilai/item dari estimator/parameter clf)
- `clf2 = pickle.loads(s)`
(pada baris ini clf2 sebagai estimator/parameter, pickle.loads sebagai suatu item, dan s sebagai estimator/parameter yang dipanggil)
- `clf2.predict(X[0:1])`
(pada baris ini clf2.predict sebagai suatu item dengan menggunakan metode predict untuk menentukan suatu nilai dari (X[0:1]))

- `y[0]`

(pada estimator/parameter y berapapun angka yang diganti nilainya akan selalu konstan yaitu 0)

- `from joblib import dump, load`

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

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

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

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

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

- dari ketiga baris akhir tersebut jika di jalankan atau dituliskan perintah seperti itu maka akan menampilkan tampilan eror

1.7.0.2 Conventions

1. Type Casting

- `from sklearn import svm`

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

- `from sklearn import random_projection`

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

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

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

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

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

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

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

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

2. Refitting and Updating Parameters

- `import numpy as np`
(pada baris ini merupakan sebuah perintah untuk mengimport class `svm` dari `np`).

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

3. Multiclass vs. Multilabel Fitting

- `from sklearn.svm import SVC`
(pada baris ini merupakan sebuah perintah untuk mengimport class SVC dari packaged sklearn.svm).
- `from sklearn.multiclass import OneVsRestClassifier`
(pada baris ini merupakan sebuah perintah untuk mengimport class OneVsRestClassifier dari packaged sklearn.multiclass).
- `from sklearn.preprocessing import LabelBinarizer`
(pada baris ini merupakan sebuah perintah untuk mengimport class LabelBinarizer dari packaged sklearn.preprocessing).
- `X = [[1, 2], [2, 4], [4, 5], [3, 2], [3, 1]]`

- `y = [0, 0, 1, 1, 2]`
- `classif = OneVsRestClassifier(estimator=SVC(gamma='scale', random_state=0))`
- `classif.fit(X, y).predict(X)`
- `y = LabelBinarizer().fit_transform(y)`
- `classif.fit(X, y).predict(X)`
- `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)`

1.7.1 Penanganan eror

1.7.1.1 ScreenShoot Eror

```
>>> 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 1.38: Hasil Tampilan Error.

1.7.1.2 Tuliskan Kode Eror dan Jenis Erornya

- `from joblib import dump, load`
(Kode baris pertama)

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

(Errornya)

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

(Kode baris kedua)

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

(Errornya)

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

(Kode baris ketiga)

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

(Errornya)

1.7.1.3 Solusi Pemecahan Masalah Error

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

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

Figure 1.39: Hasil Tampilan Uji coba perintah joblib.

78cdb6514db9716f252ef2024a7df6097aace611

Chapter 2

Related Works

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

2.1 Aip Suprapto Munari/1164063

2.1.1 Teori

2.1.2 Binary Classification

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. Beberapa contoh yang meliputi Binary Classification adalah

- Deteksi Transaksi Penipuan Kartu Kredit
- Diagnosa medis
- Deteksi Spam

Untuk contoh Binary Classification dapat dilihat pada gambar 7.2

2.1.3 Supervised Learning, Unsupervised Learning, Dan Clustering

1. Supervised Learning merupakan sebuah pendekatan yang dimana sudah adanya sdata yang dilatih dan telah terdapat variabel yang telah ditargetkan sehingga

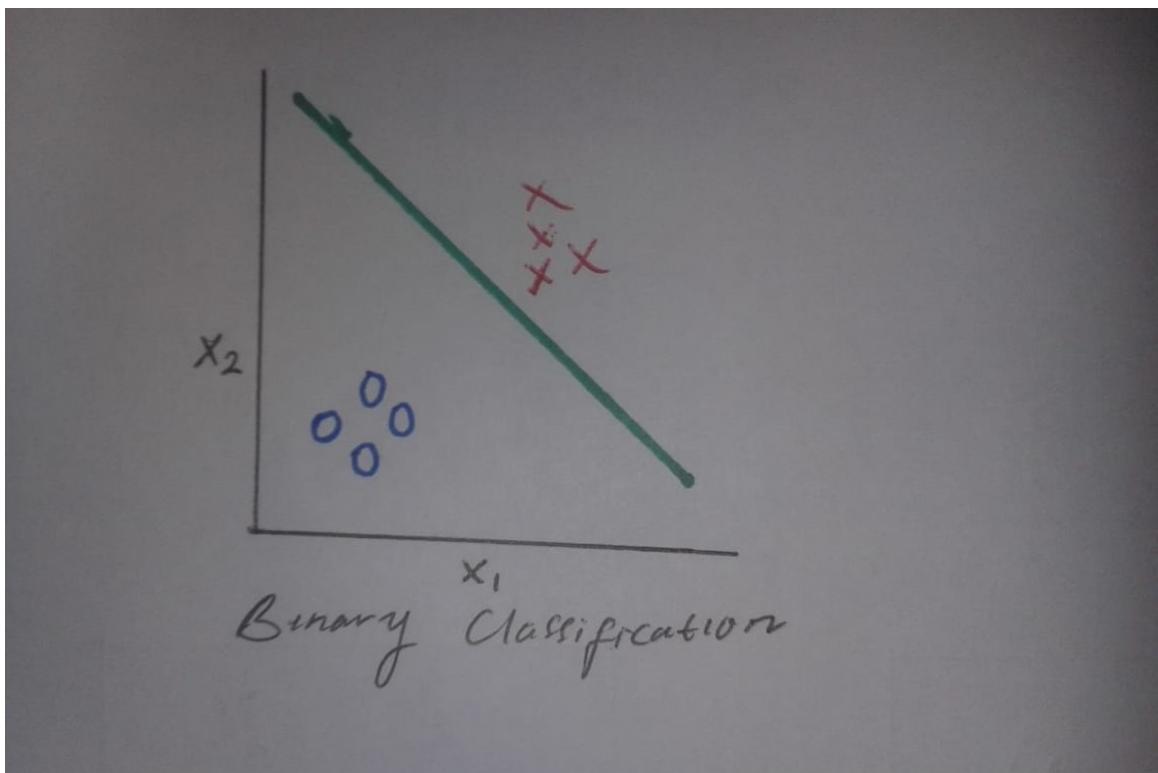


Figure 2.1: Binary Classification.

bertujuan untuk mengelompokkan suatu data ke data yang sudah ada. Contoh dalam Supervised Learning yaitu ketika anda memiliki sejumlah buku yang yang telah dilabel dengan urutan kategori tertentu. Ketika anda akan membeli sebuah buku baru, maka harus di identifikasi isi dari buku tersebut dan memasukkannya kedalam kategori tertentu. Ketika anda membeli sebuah buku tersebut maka anda telah menerapkan sebuah logika fuzzy. Ilustrasi Supervised Learning dapat dilihat pada gambar 7.3.

2. 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. Ilustrasi Unsupervised Learning dapat dilihat pada gambar 7.3.
3. Clustering 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.

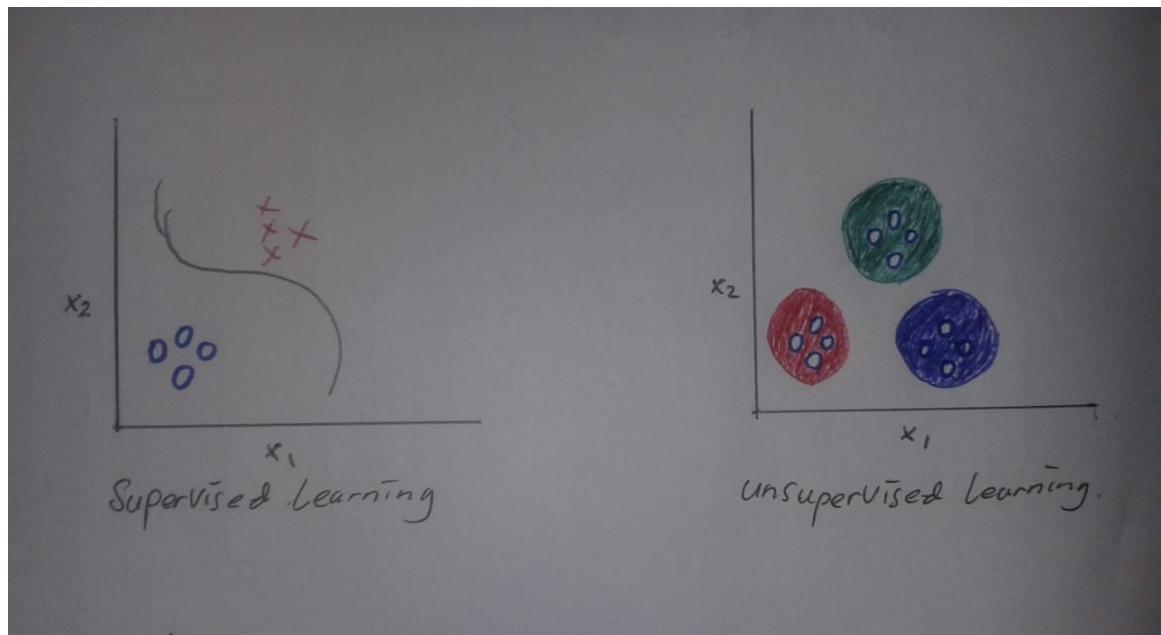


Figure 2.2: Supervised Learning.

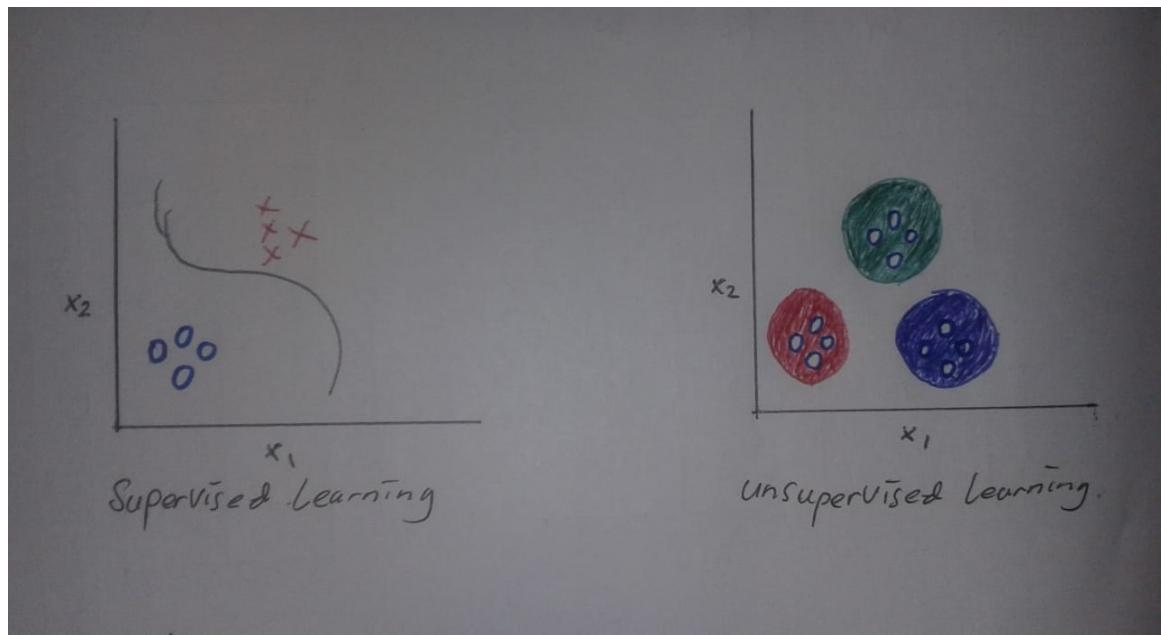


Figure 2.3: Unsupervised Learning.

Parameter bagi orang yang memiliki tubuh yang kuat adalah orang yang terlihat bugar dan sehat maka dengan orang yang memiliki parameter adalah orang yang memiliki kekuatan tubuh yang kuat dan untuk kekuatan tubuh yang lemah adalah sebaliknya. Ilustrasi gambar dapat di lihat di gambar 7.4

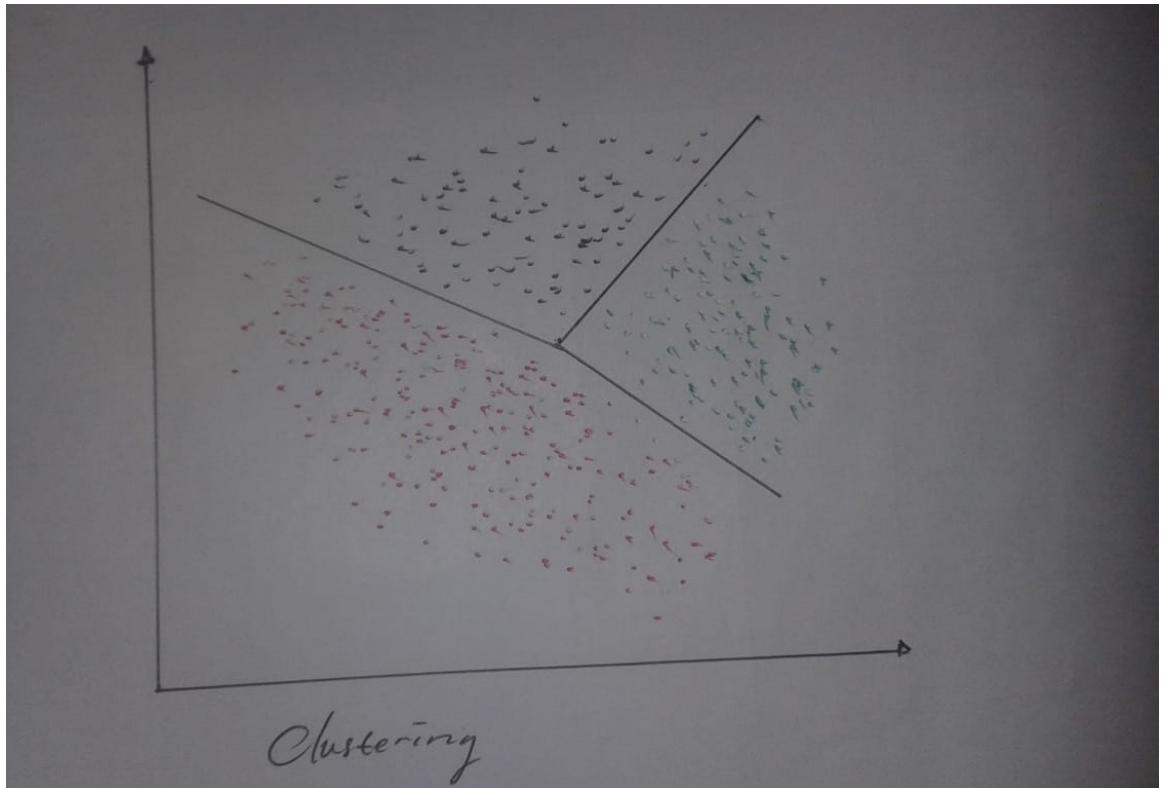


Figure 2.4: Clustering.

2.1.4 Evaluasi Dan Akurasi

1. Evaluasi dan akurasi adalah bagaimana cara kita dapat mengevaluasi seberapa baik model melakukan pekerjaannya dengan cara mengukur akurasinya. Akurasi akan didefinisikan sebagai persentase kasus yang telah diklasifikasikan dengan benar. Kita dapat melakukan analisis kesalahan yang telah dibuat oleh model. Dalam tabel tersebut baris true mangga dan true anggur menunjukkan kasus apakah itu objek mangga atau anggur. Kolom telah diprediksi dan dibuat oleh model. Ada 20 cow yang diprediksi benar dan ada 5 buffalo yang diprediksi salah. Ilustrasi dapat di lihat pada gambar 7.5

2.1.5 Confusion Matrix

1. Ada beberapa cara untuk membuat dan membaca confusion matrix antara lain

	<u>Predicted cow</u>	<u>Predicted buffalo</u>
<u>True cow</u>	15	5
<u>True buffalo</u>	7	13

Figure 2.5: Evaluasi Dan Akurasi.

- Tentukan pokok permasalahan serta atributnya
- Buat Decision Tree
- Buat Data Testing
- Mencari nilai variabelnya misal a,b,c, dan d
- Mencari nilai recall, precision, accuracy, dan error rate

Di bawah ini adalah contoh dari confusion matrix

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

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

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

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

2.1.6 Cara Kerja K-Fold Cross Validation

1. Untuk cara kerja K-Fold Cross Validation adalah sebagai berikut

- Total instance dibagi menjadi N bagian.
- Fold yang pertama adalah bagian pertama menjadi 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.

Untuk ilustrasi K-Fold Cross Validation data di lihat pada gambar 7.6

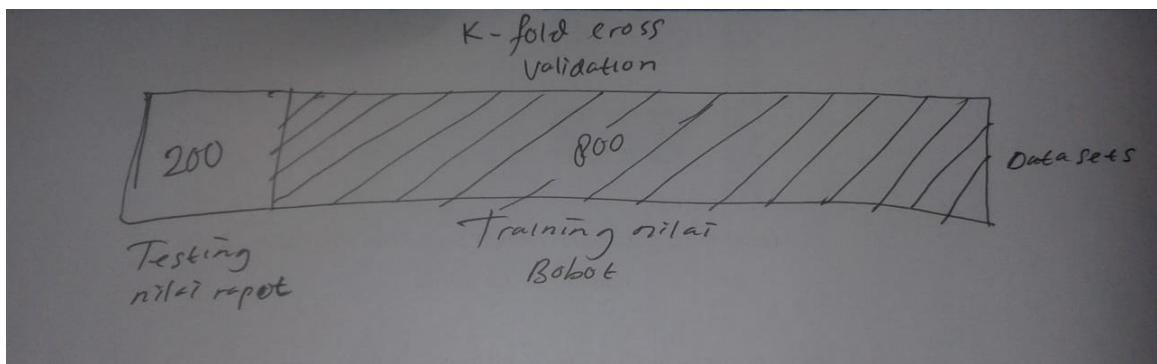


Figure 2.6: K-Fold Cross Validation.

2.1.7 Decision Tree

1. Decision Tree adalah sebuah metode pembelajaran yang digunakan untuk melakukan klarifikasi dan regresi. Decision Tree digunakan untuk membuat sebuah model yang dapat memprediksi sebuah nilai variabel target dengan cara mempelajari aturan keputusan dari fitur data. Contoh Decision Tree adalah untuk melakukan prediksi apakah Sapi termasuk hewan herbivora atau bukan, lihat pada gambar 7.7.

2.1.8 Gain Dan Entropi

1. 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.
2. Entropi adalah ukuran ketidakpastian sebuah variabel acak sehingga dapat diartikan entropi adalah ukuran ketidakpastian dari sebuah atribut.

Ilustrasi dari gain dan entropi adalah bagaimana kita memprediksi jenis kelamin berdasarkan atributnya, perhatikan pada gambar 7.8

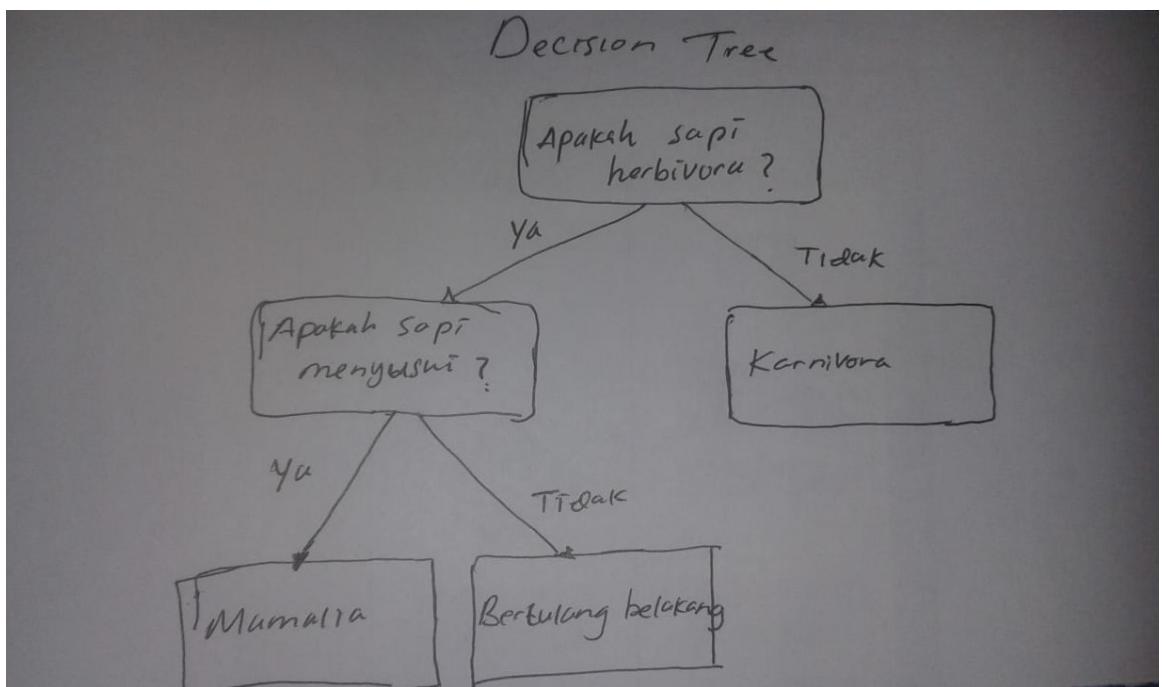


Figure 2.7: Decision Tree.

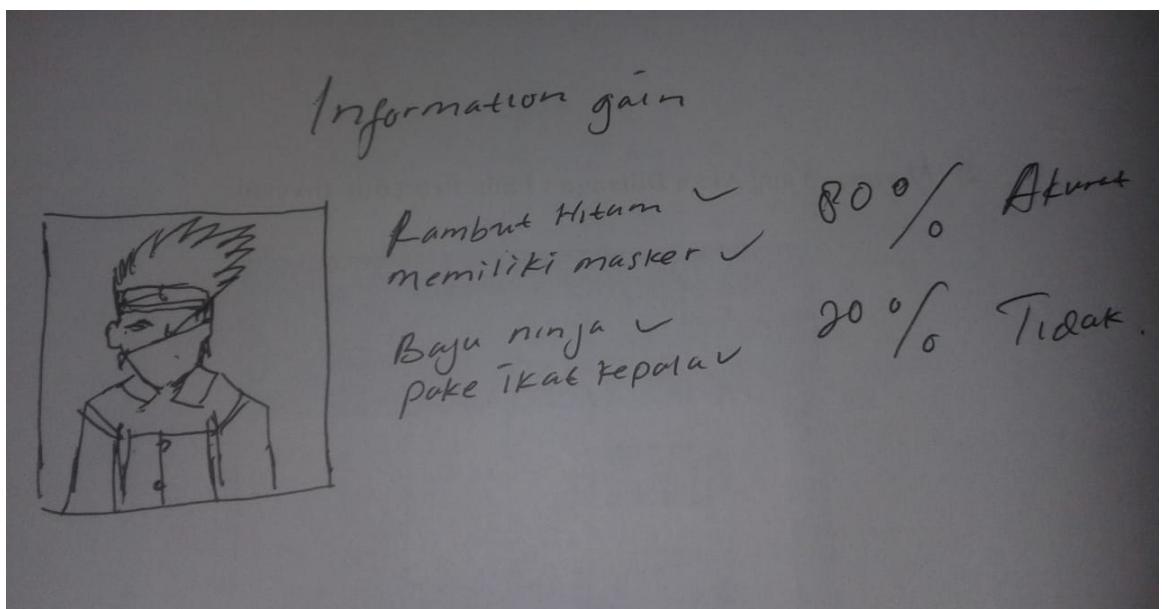


Figure 2.8: Gain Dan Entropi.

2.2 Aip Suprapto Munari/1164063

2.2.1 Scikit-learn

Penyelesaian Tugas Harian 4

- Pembahasan Codingan Dan Hasilnya

1. Gambar Pertama :

Penjelasan : Pada baris pertama itu merupakan import library sebagai variabel siomay Dan pada baris kedua variabel siomay membaca file csv nya. Dan pada baris ketiga merupakan hasilnya yaitu 395.

– Hasil Gambar Pertama :

```
In [6]: import pandas as Batagor
...: Siomay = Batagor.read_csv('D:\KULIAH\SEMESTER 6\KECERDASAN BUATAN\PRAKTEK
\Python-Artificial-Intelligence-Projects-for-Beginners-master\Python-Artificial-
Intelligence-Projects-for-Beginners-master\Chapter01\dataset\student-mat.csv',
sep';')
...: len(Siomay)
Out[6]: 395
```

Figure 2.9: Gambar pertama

2. Gambar Kedua :

Penjelasan : Variabel Siomay mengimplementasikan baris 1, dari baris G1, G2, G3. Dan variabel siomay akan ngedrop kolom G1, G2,G3. Dan hasilnya akan seperti gambar di out nya.

– Hasil Gambar Kedua :

```
In [8]: Siomay['pass'] = Siomay.apply(lambda row: 1 if (row['G1']+row['G2']
+row['G3']) ...
...: >= 35 else 0, axis=1)
...: Siomay = Siomay.drop(['G1', 'G2', 'G3'], axis=1)
...: Siomay.head()
Out[8]:
   school sex  age address famsize ...  Dalc  Walc  health absences pass
0      GP    F   18      U    GT3 ...     1     1      3       6      0
1      GP    F   17      U    GT3 ...     1     1      3       4      0
2      GP    F   15      U    LE3 ...     2     3      3      10      0
3      GP    F   15      U    GT3 ...     1     1      5       2      1
4      GP    F   16      U    GT3 ...     1     2      5       4      0
[5 rows x 31 columns]
```

Figure 2.10: Gambar kedua

3. Gambar Ketiga :

Penjelasan : Variabel Siomay mengambil atau get data dari dalam kolom. Atau yang tulisan berwarna hijau. Dan kemudian ditampilkan pada outputan yang dibawah atau menampilkan hasilnya.

– Hasil Gambar Ketiga :

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

Figure 2.11: Gambar Ketiga

4. Gambar Keempat :

Penjelasan : Penejelasan pada gambar keempat adalah variabel Siomay akan menampilkan sampel data dari 500 training data dan 500 tetsing data. Kemudia data akan dicetak atau di print dari training data dan testing data.

– Hasil Gambar Keempat :

```
In [10]: Siomay = Siomay.sample(frac=1)
...: # split training and testing data
...: Siomay_train = Siomay[:500]
...: Siomay_test = Siomay[500:]
...:
...: Siomay_train_att = Siomay_train.drop(['pass'], axis=1)
...: Siomay_train_pass = Siomay_train['pass']
...:
...: Siomay_test_att = Siomay_test.drop(['pass'], axis=1)
...: Siomay_test_pass = Siomay_test['pass']
...:
...: Siomay_att = Siomay.drop(['pass'], axis=1)
...: Siomay_pass = Siomay['pass']
...:
...: # number of passing students in whole dataset:
...: import numpy as np
...: print("Passing: %d out of %d (%.2f%%)" % (np.sum(Siomay_pass), len(Siomay_pass),
...: 100*float(np.sum(Siomay_pass)) / len(Siomay_pass)))
Passing: 166 out of 395 (42.03%)
```

Figure 2.12: Gambar Keempat

5. Gambar Kelima :

Penjelasan : Pada gambar tersebut variabel hanya melakukan pengetesan/pengecekan terhadap decission tree. Apabila decission tree nya benar maka kodingan tidak eror tapi jika tidak benar maka kodingan akan error.

– Hasil Gambar Kelima :

```
In [12]: from sklearn import tree
...: lepet = tree.DecisionTreeClassifier(criterion="entropy", max_depth=5)
...: lepet = lepet.fit(Siomay_train_att, Siomay_train_pass)
```

Figure 2.13: Gambar Kelima

6. Gambar Keenam :

Penjelasan : Pada gambar nomor 6, terjadi kesalahan error yaitu pada import graphviz.

– Hasil Gambar Keenam :

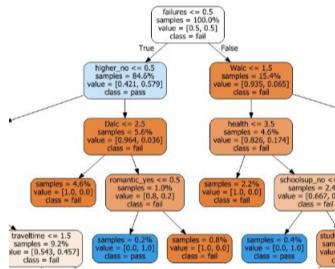


Figure 2.14: Gambar Keenam

7. Gambar Ketujuh :

Penjelasan : Pada gambar 7 akan menampilkan yang terdapat pada Library Graphviz, apabila benar akan menampilkan hasil output seperti yang terdapat pada gambar atau kalau pengujian gagal akan terdapat error.

– Hasil Gambar Ketujuh :

```
In [22]:  
In [22]: from sklearn import tree  
...: tree.export_graphviz(Batagor, out_file="student-performance.dot", label="all",  
impurity=False, proportion=True,  
...:  
"pass"],  
...:  
filled=True, rounded=True)
```

Figure 2.15: Gambar Ketujuh

8. Gambar Kedelapan :

Penjelasan : Pada gambar 8 menampilkan hasil perhitungan dari kedua parameter yang terdapat pada code tersebut.

– Hasil Gambar Kedelapan :

```
In [34]: Batagor.score(Siomay_test_att, Siomay_test_pass)  
Out[34]: 0.8583579357035
```

Figure 2.16: Gambar Kedelapan

9. Gambar Kesembilan:

Penjelasan : Pada gambar 9, kodingan teresbut mnedefinisikan library sklearn model selection dan import cross val score. Dan kemudian variabel scores mengeksekusi fungsi cross val score(Batagor, Siomay att, Siomay pass, cv=5). Kemudian akan menampilkan nilai dari fungsi akurasinya.

– Hasil Gambar Kesembilan :

```
In [26]: from sklearn.model_selection import cross_val_score
...: scores = cross_val_score(Batagor, Siomay_att, Siomay_pass, cv=5)
...: # show average score and +/- two standard deviations away (covering 95% of
...: scores)
...: print("Accuracy: %0.2f (+/- %0.2f)" % (scores.mean(), scores.std() * 2))
Accuracy: 0.56 (+/- 0.06)
```

Figure 2.17: Gambar Kesembilan

10. Gambar Kesepuluh :

Penjelasan : 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.

– Hasil Gambar Kesepuluh :

```
In [27]: for max_depth in range(1, 20):
...:     Batagor = tree.DecisionTreeClassifier(criterion="entropy",
max_depth=max_depth)
...:     scores = cross_val_score(Batagor, Siomay_att, Siomay_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 (+/- 0.08)
Max depth: 3, Accuracy: 0.53 (+/- 0.05)
Max depth: 4, Accuracy: 0.54 (+/- 0.05)
Max depth: 5, Accuracy: 0.56 (+/- 0.07)
Max depth: 6, Accuracy: 0.58 (+/- 0.09)
Max depth: 7, Accuracy: 0.57 (+/- 0.06)
Max depth: 8, Accuracy: 0.57 (+/- 0.08)
Max depth: 9, Accuracy: 0.56 (+/- 0.06)
Max depth: 10, Accuracy: 0.56 (+/- 0.06)
Max depth: 11, Accuracy: 0.55 (+/- 0.05)
Max depth: 12, Accuracy: 0.56 (+/- 0.06)
Max depth: 13, Accuracy: 0.55 (+/- 0.05)
Max depth: 14, Accuracy: 0.57 (+/- 0.09)
Max depth: 15, Accuracy: 0.56 (+/- 0.06)
Max depth: 16, Accuracy: 0.57 (+/- 0.07)
Max depth: 17, Accuracy: 0.55 (+/- 0.07)
Max depth: 18, Accuracy: 0.57 (+/- 0.06)
Max depth: 19, Accuracy: 0.56 (+/- 0.07)

In [28]: |
```

Figure 2.18: Gambar Kesepuluh

11. Gambar Kesebelas :

Penjelasan : Pada gambar 11 dijelaskan bahwa variable scores akan menampilkan atau mendefinisikan nilai dari variabel score yang mana isi dari variable score yaitu Batagor, Siomay att, Siomay pass, cv=5. Yang mana hasil tampilan dari kodingannya adalah outputan seperti gambar 11.

– Hasil Gambar Kesebelas :

```
In [28]: depth_acc = np.empty((19,3), float)
....: i = 0
....: for max_depth in range(1, 20):
....:     t = tree.DecisionTreeClassifier(criterion="entropy", max_depth=max_depth)
....:     scores = cross_val_score(Batagor, Siomay_att, Siomay_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[28]:
array([[ 1.        ,  0.54674294,  0.04923122],
       [ 2.        ,  0.55936952,  0.06579881],
       [ 3.        ,  0.57949692,  0.09362112],
       [ 4.        ,  0.57202856,  0.05271394],
       [ 5.        ,  0.56953019,  0.06961587],
       [ 6.        ,  0.56193363,  0.04121254],
       [ 7.        ,  0.56186871,  0.0611709 ],
       [ 8.        ,  0.57209185,  0.08326673],
       [ 9.        ,  0.55687033,  0.05993239],
       [10.        ,  0.58196284,  0.08779916],
       [11.        ,  0.53408309,  0.07935713],
       [12.        ,  0.56702856,  0.0587212 ],
       [13.        ,  0.57693119,  0.08667513],
       [14.        ,  0.56712512,  0.04993824],
       [15.        ,  0.54430785,  0.05251297],
       [16.        ,  0.55953019,  0.05094885],
       [17.        ,  0.5593038 ,  0.07170369]]
```

Figure 2.19: Gambar Kesebelas

12. Gambar Keduabelas :

Penjelasan : Pada gambar di atas dijelaskan bahwa pada library matplotlib akan menampilkan gambar grafik pada gambar 12 dari eksekusi fungsi ax.errorbar.

– Hasil Gambar Keduabelas :

```
In [30]:
In [30]: import matplotlib.pyplot as plt
....: fig, ax = plt.subplots()
....: ax.errorbar(depth_acc[:,0], depth_acc[:,1], yerr=depth_acc[:,2])
....: plt.show()
```

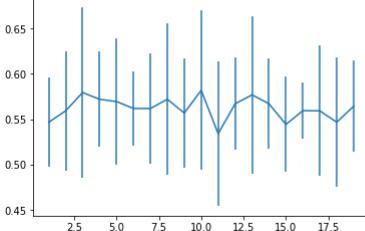


Figure 2.20: Gambar Keduabelas

2.2.2 Penanganan Error

1. Skrinsut Error

```
In [65]: import graphviz
...: dot_data = tree.export_graphviz(solo, out_file=None, label="all", impurity=False, proportion=True,
...:                                 feature_names=list(solok_train_att), class_names=["fail", "pass"],
...:                                 filled=True, rounded=True)
...: graph = graphviz.Source(dot_data)
...: graph
Traceback (most recent call last):
File "<ipython-input-65-47940b5e4aa1>", line 1, in <module>
    import graphviz
ModuleNotFoundError: No module named 'graphviz'
```

Figure 2.21: Gambar Ketigabelas

2. Kode Error dan Jenis Errornya

Kode Error: "Import Graphiz" dan "ModulNotFoundError".

Jenis Error: Pada Grafik

3. Penanganan

Melakukan install ulang pada graphiz

2.3 Andi Aslam/1164064

2.3.1 Binary Clasification beserta gambar

1. Output aktual dari banyak algoritma klasifikasi biner 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, sebagai konsumen skor ini, Anda akan menginterpretasikan skor dengan memilih ambang klasifikasi (cut-off) dan membandingkan skor dengan itu. Setiap pengamatan dengan skor lebih tinggi dari ambang kemudian diprediksi sebagai kelas positif dan skor lebih rendah dari ambang diprediksi sebagai kelas negatif.

2.3.2 supervised learning dan unsupervised learning dan clustering dengan ilustrasi gambar

1. Supervised learning adalah tugas pembelajaran mesin untuk mempelajari suatu fungsi yang memetakan input ke output berdasarkan contoh pasangan input-output. Ini menyimpulkan fungsi dari data pelatihan berlabel yang terdiri dari serangkaian contoh pelatihan. Dalam pembelajaran yang diawasi, setiap contoh

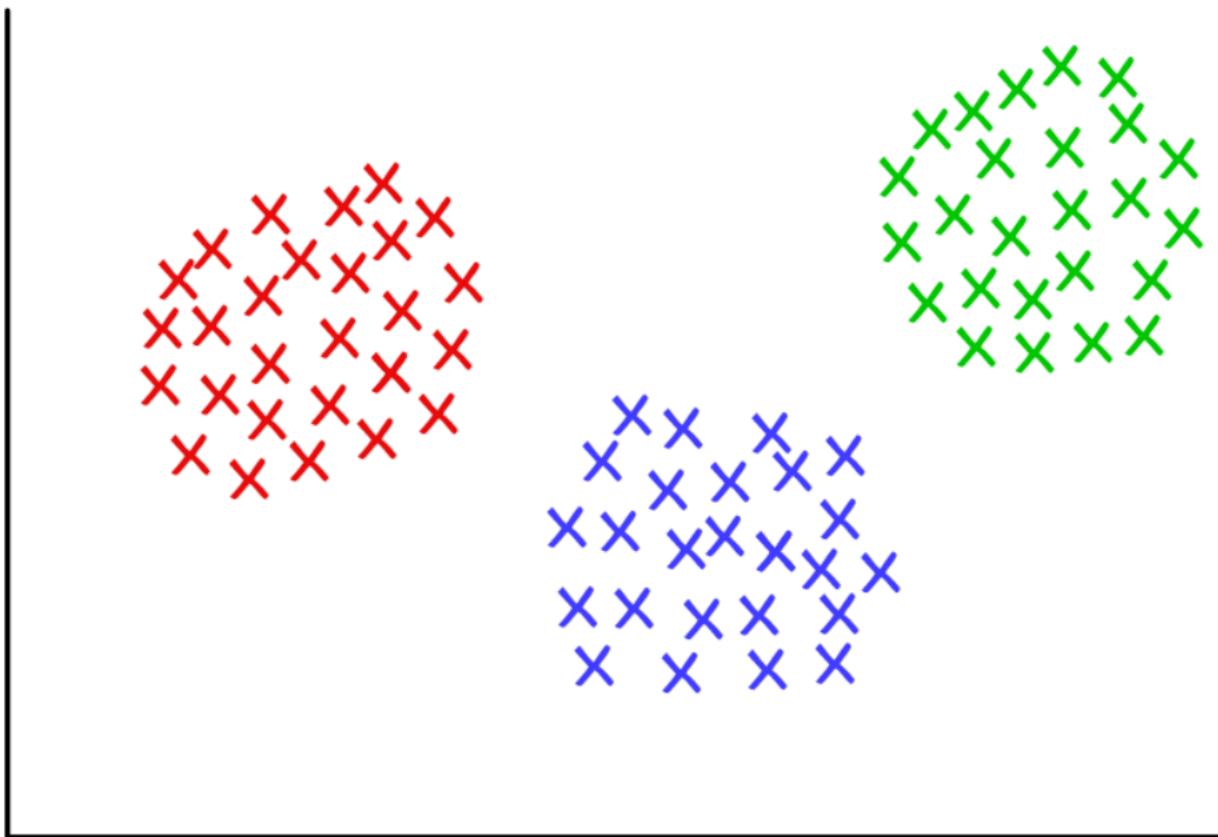


Figure 2.22: Binary Clasification

adalah pasangan yang terdiri dari objek input (biasanya vektor) dan nilai output yang diinginkan (juga disebut sinyal pengawas). Algoritma pembelajaran yang diawasi menganalisis data pelatihan dan menghasilkan fungsi yang disimpulkan, yang dapat digunakan untuk memetakan contoh-contoh baru. Skenario optimal akan memungkinkan algoritma menentukan label kelas dengan benar untuk instance yang tidak terlihat. Ini membutuhkan algoritma pembelajaran untuk menggeneralisasi dari data pelatihan untuk situasi yang tidak terlihat dengan cara yang "masuk akal" (lihat bias induktif).

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. Anal-

isis cluster sebagai cabang pembelajaran mesin yang mengelompokkan data yang belum diberi label, diklasifikasikan atau dikategorikan. Alih-alih menanggapi umpan balik, analisis klaster mengidentifikasi kesamaan dalam data dan bereaksi berdasarkan ada tidaknya kesamaan di setiap potongan data baru. Berikut merupakan contoh Unsupervised Learning dengan Gaussian mixture models.

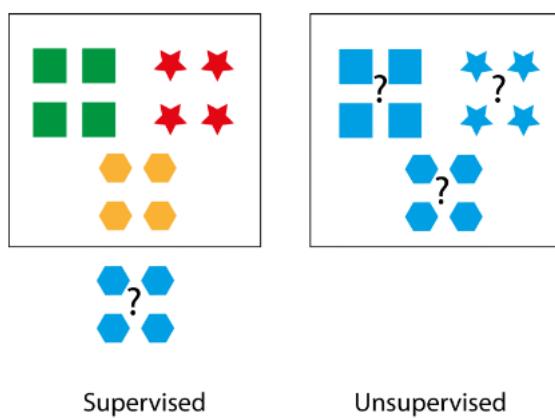


Figure 2.23: Supervised Learning

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

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

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

1. Confusion matrix :

- 1) Tentukan pokok permasalahan dan atributanya, misal gaji dan listrik.
- 2) Buat pohon keputusan

	<u>Pradicted Game</u>	<u>Pradicted Movie</u>
<u>True Game</u>	99%	56%
<u>True Movie</u>	66%	73%

Figure 2.24: Evaluasi dan Akurasi

- 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.

K-Fold Cross Validation

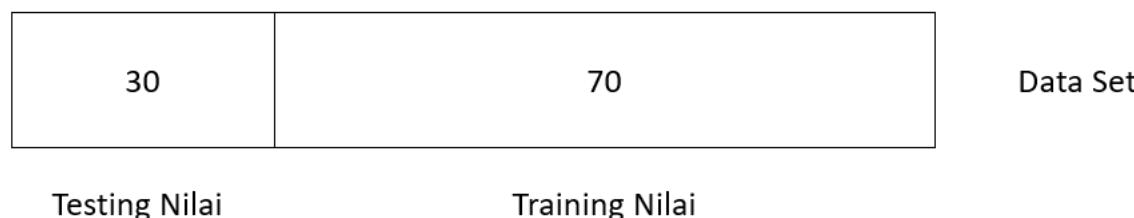


Figure 2.25: K-fold cross validation

2.3.6 decision tree dengan gambar ilustrasi

1. Decision Tree adalah metode pembelajaran yang diawasi non-parametrik yang digunakan untuk klasifikasi dan regresi. Tujuannya adalah untuk membuat model yang memprediksi nilai variabel target dengan mempelajari aturan keputusan sederhana yang disimpulkan dari fitur data.
Misalnya, dalam contoh di bawah ini, decision tree belajar dari data untuk memperkirakan kurva sinus dengan seperangkat aturan keputusan if-then-else. Semakin dalam pohon, semakin rumit aturan keputusan dan semakin bugar modelnya.

2.3.7 Information Gain dan entropi dengan gambar ilustrasi

1. Information gain didasarkan pada penurunan entropi setelah dataset dibagi pada atribut. Membangun decision tree adalah semua tentang menemukan atribut yang mengembalikan perolehan informasi tertinggi (mis., Cabang yang paling homogen).
2. Entropi adalah ukuran keacakan dalam informasi yang sedang diproses. Semakin tinggi entropi, semakin sulit untuk menarik kesimpulan dari informasi itu. Membalik koin adalah contoh tindakan yang memberikan informasi yang

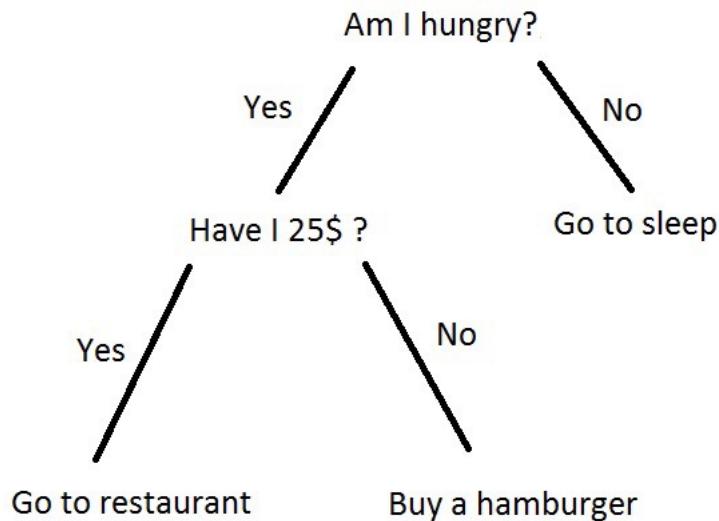


Figure 2.26: Decision Tree

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

Information Gain

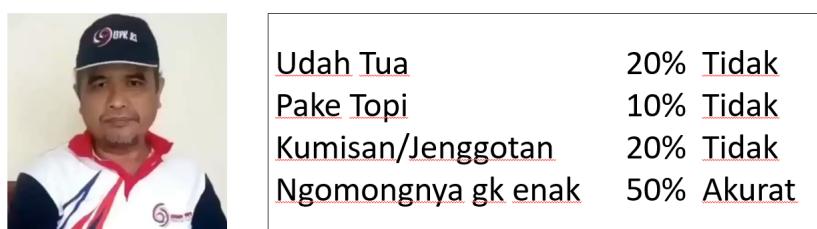


Figure 2.27: Entropi

2.4 Andiaslam/1164064

2.4.1 Scikit-learn

1. Pembahasan Dan Hasil
2. variabel Dan pada baris kedua variabel solok membaca file csv nya. Dan pada baris ketiga merupakan hasilnya yaitu 395.

- Hasil Code 1:

```
In [6]: import pandas as Surabaya
...: Jatim = Surabaya.read_csv('D:\\Tugas Kuliah\\Semester 6\\KECERDASAN BUATAN\\Python-Artificial-Intelligence-Projects-for-Beginners\\Chapter01\\dataset\\student-mat.csv', sep=',')
...: len(Jatim)
Out[6]: 395
```

Figure 2.28: Hasil Code 1

3. variable dapat mengimplementasi bari ke 1 dari baris G1, G2, dan G3.

- Hasil 2 :

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

Figure 2.29: Hasil Code 2

4. Code 3.

Penjelasan:

variabel mengambil data dari kolom. kemudian akan di tampilkan pada output

- Hasil Code 3:

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

Figure 2.30: Hasil Code 3

5. Code 4.

Penjelasan:

Variabel akan menampilkan sampel data dari 500 data training dan 500 data testing.

- Hasil Code 4:

```

.... Datim_train = Datim[500]
.... Datim_test = Datim[500:]
.... Datim_train_att = Datim_train.drop(['pass'], axis=1)
.... Datim_train_pass = Datim_train['pass']
.... Datim_test_att = Datim_test.drop(['pass'], axis=1)
.... Datim_test_pass = Datim_test['pass']
.... Datim_att = Datim.drop(['pass'], axis=1)
.... Datim_pass = Datim['pass']

.... # number of passing students in whole dataset
.... import numpy as ponpes
.... print("Passing: %d out of %d (%.2f%%)" % (ponpes.sum(Datim_pass), len(Datim_pass), 100*float(ponpes.sum(Datim_pass)) / len(Datim_pass)))
Passing: 166 out of 395 (42.03%)

```

Figure 2.31: Hasil Code 4

6. Code 5.

Penjelasan:

variabel hanya melakukan pengetesan/pengecekan terhadap decision tree.

- Hasil Code 5:

```
In [33]: from sklearn import tree
.... amanatul = tree.DecisionTreeClassifier(criterion="entropy", max_depth=5)
.... amanatul = amanatul.fit(Datim_train_att, Datim_train_pass)
```

Figure 2.32: Hasil Code 5

7. Code 6.

Penjelasan:

terjadi error pada import graphviz.

- Hasil Code 6:

```
In [42]: import graphviz
.... dot_data = tree.export_graphviz(amanatul, out_file=None, label="all", impurity=False
.... proportion=True,
.... feature_names=list(Datim_train_att), class_names=["fail", "pass"],
.... filled=True, rounded=True)
.... graph = graphviz.Source(dot_data)
.... graph
Traceback (most recent call last):
File "<ipython-input-42-554c8002ecf8>", line 1, in <module>
    import graphviz
ModuleNotFoundError: No module named 'graphviz'
```

Figure 2.33: Hasil Code 6

8. Code 7.

Penjelasan:

menampilkan yang terdapat pada Library Graphviz, apabila benar akan menampilkan hasil output seperti yang terdapat pada gambar atau kalau pengujian gagal akan terdapat error.

- Hasil Code 7:

```
In [34]: tree.export_graphviz(amanatul, out_file="student-performance.dot", label="all",
    impurity=False, proportion=True,
    ...:                                         feature_names=list(Jatim_train_att), class_names=['fail',
    "pass"], ...
    ...:                                         filled=True, rounded=True)
```

Figure 2.34: Hasil Code 7

9. Code 8.

Penjelasan:

menampilkan hasil perhitungan dari kedua parameter yang terdapat pada code tersebut.

- Hasil Code 8:

```
In [43]: amanatul.score(Jatim_test_att, Jatim_test_pass)
Traceback (most recent call last):
File "<ipython-input-43-8e7ca9451ca2>", line 1, in <module>
    amanatul.score(Jatim_test_att, Jatim_test_pass)
File "C:\ProgramData\Anaconda3\lib\site-packages\sklearn\base.py", line 288, in score
    return accuracy_score(y, self.predict(X), sample_weight=sample_weight)
File "C:\ProgramData\Anaconda3\lib\site-packages\sklearn\tree\tree.py", line 416, in predict
    X = self._validate_X_predict(X, check_input)
File "C:\ProgramData\Anaconda3\lib\site-packages\sklearn\tree\tree.py", line 377, in _validate_X_predict
    X = check_array(X, dtype=DTYPE, accept_sparse="csr")
File "C:\ProgramData\Anaconda3\lib\site-packages\sklearn\utils\validation.py", line 582, in check_array
    context))
```

Figure 2.35: Hasil Code 8

10. Code 9.

Penjelasan:

kodingan tersebut mendefinisikan library sklearn model selection dan import cross_val_score. Dan kemudian variabel scores mengeksekusi fungsi cross_val_score(solo, solok att, solok pass, cv=5). Kemudian akan menampilkan nilai dari fungsi akurasinya.

- Hasil Code 9:

```
In [13]: from sklearn.model_selection import cross_val_score
...: scores = cross_val_score(amanatul, Jatim_att, Jatim_pass, cv=5)
...: # show average score and +/- two standard deviations away (covering 95% of scores)
...: print("Accuracy: %0.2f (+/- %0.2f)" % (scores.mean(), scores.std() * 2))
Accuracy: 0.57 (+/- 0.11)
```

Figure 2.36: Hasil Code 9

11. Code 10.

Penjelasan:

berfungsi untuk menampilkan hasil dari fungsi Max Depth dan Accuracy dari Decission Tree. Yaitu menampilkan data dari angka 1-20.

- Hasil Code 10:

```
In [15]: for max_depth in range(1, 20):
...:     amanatul = tree.DecisionTreeClassifier(criterion="entropy", max_depth=max_depth)
...:     scores = cross_val_score(amanatul, Jatim_att, Jatim_pass, cv=5)
...:     print("Max depth: %d, Accuracy: %0.2f (+/- %0.2f)" % (max_depth, scores.mean(), scores.std() * 2))

Max depth: 1, Accuracy: 0.55 (+/- 0.01)
Max depth: 2, Accuracy: 0.55 (+/- 0.09)
Max depth: 3, Accuracy: 0.56 (+/- 0.07)
Max depth: 4, Accuracy: 0.58 (+/- 0.08)
Max depth: 5, Accuracy: 0.57 (+/- 0.11)
Max depth: 6, Accuracy: 0.58 (+/- 0.05)
Max depth: 7, Accuracy: 0.57 (+/- 0.08)
Max depth: 8, Accuracy: 0.57 (+/- 0.09)
Max depth: 9, Accuracy: 0.57 (+/- 0.05)
Max depth: 10, Accuracy: 0.58 (+/- 0.07)
Max depth: 11, Accuracy: 0.59 (+/- 0.09)
Max depth: 12, Accuracy: 0.59 (+/- 0.09)
Max depth: 13, Accuracy: 0.57 (+/- 0.10)
Max depth: 14, Accuracy: 0.55 (+/- 0.12)
Max depth: 15, Accuracy: 0.57 (+/- 0.07)
Max depth: 16, Accuracy: 0.56 (+/- 0.15)
Max depth: 17, Accuracy: 0.57 (+/- 0.07)
Max depth: 18, Accuracy: 0.55 (+/- 0.08)
Max depth: 19, Accuracy: 0.55 (+/- 0.10)
```

Figure 2.37: Hasil Code 10

12. Code 11.

Penjelasan:

variable scores akan menampilkan atau mendefinisikan nilai dari variabel score yang mana isi dari variable score Yang mana hasil tampilan dari kodingannya adalah output

- Hasil Code 11:

```
In [38]:
In [38]: depth_acc = np.zeros((19,3), float)
...: i = 0
...: for max_depth in range(1, 20):
...:     amanatul = tree.DecisionTreeClassifier(criterion="entropy", max_depth=max_depth)
...:     scores = cross_val_score(amanatul, Jatim_att, Jatim_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[38]:
array([[1.00000000e+00, 5.79751704e-01, 6.30768599e-03],
       [2.00000000e+00, 5.87540571e-01, 5.55999992e-02],
       [3.00000000e+00, 5.72188413e-01, 6.54746741e-02],
       [4.00000000e+00, 5.79718436e-01, 4.96526886e-02],
       [5.00000000e+00, 5.99717624e-01, 1.13283469e-01],
       [6.00000000e+00, 5.92316618e-01, 7.88112865e-02],
       [7.00000000e+00, 5.92314164e-01, 9.25897095e-02],
       [8.00000000e+00, 6.05004869e-01, 8.47416799e-02],
```

Figure 2.38: Hasil Code 11

13. Code 12.

Penjelasan:

pada library matplotlib akan menampilkan gambar grafik pada gambar 12 dari eksekusi fungsi ax.errorbar.

- Hasil Code 12:

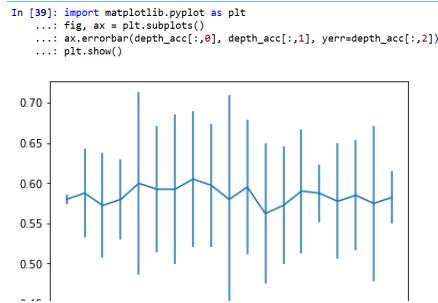


Figure 2.39: Hasil Code 12

```
IPython 6.4.0 -- An enhanced Interactive Python.
In [39]: import matplotlib.pyplot as plt
...: fig, ax = plt.subplots()
...: ax.errorbar(depth_acc[:,0], depth_acc[:,1], yerr=depth_acc[:,2])
...: plt.show()

0.70
0.65
0.60
0.55
0.50
...
```

Figure 2.40: HASIL YANG MASIH ERROR

2.4.2 Praktek Penanganan Error

Traceback (most recent call last):

```
File "<ipython-input-33-b4843d06cfa2>", line 1, in <module>
    import graphviz
```

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

Solusi dari error tersebut adalah memasukkan path dari graphviz ke environment variabels, maka akan menampilkan hasil yaitu sebuah decision tree.

2.5 Same Topics

Cite every latest journal with same topic

2.5.1 Topic 1

cite for first topic

2.5.2 Topic 2

if you have two topics you can include here to

2.6 Same Method

write and cite latest journal with same method

2.6.1 Method 1

cite and paraphrase method 1

2.6.2 Method 2

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

Chapter 3

Methods

3.1 The data

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

3.2 Method 1

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

3.3 Method 2

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

3.4 Aip Suprapto Munari/1164063

3.4.1 Teori

Tugas Harian 5

1. Random Forest Dan Ilustrasi Gambarnya

- Pengertian Random Forest:

Random Forest adalah suatu algoritma yang digunakan pada klasifikasi data dalam jumlah yang besar. Klasifikasi random forest dilakukan melalui penggabungan pohon dengan melakukan training pada sampel data yang dimiliki. Penggunaan pohon (tree) yang semakin banyak akan mempengaruhi akurasi yang akan didapatkan menjadi lebih baik. Penentuan klasifikasi dengan random forest diambil berdasarkan hasil voting dari pohon

yang terbentuk. Pemenang dari pohon yang terbentuk ditentukan dengan vote terbanyak.

Pembangunan pohon pada random forest sampai dengan mencapai ukuran maksimum dari pohon data. Akan tetapi,pembangunan pohon random forest tidak dilakukan pemangkasan yang merupakan sebuah metode untuk mengurangi kompleksitas ruang.

- Ilustrasi Gambar Random Forest :

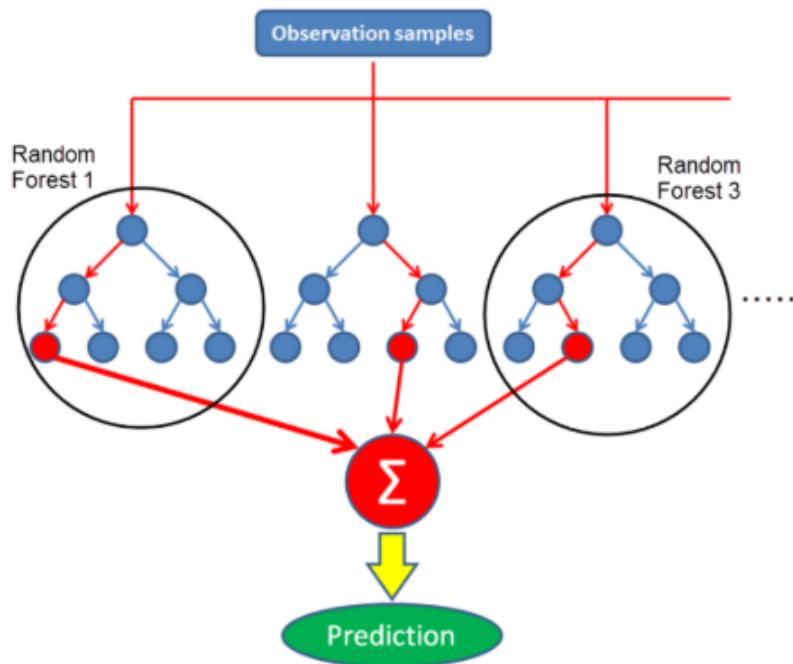


Figure 3.1: Random Forest

2. Cara Membaca Dataset

Berikut adalah cara membaca dataset :

- Buka Anaconda Navigator lalu jalankan Syder, kemudian import libraries yang dibutuhkan.
- Masukkan kode python untuk membaca file csv, lalu jalankan

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

Figure 3.2: (b)

- (c) Maka pada window console akan menampilkan pesan berikut :

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

Figure 3.3: (c)

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

Figure 3.4: (d)

(d) Dari explorer dapat terlihat dataset yang terimport.

(e) Lalu klik dataset cell, maka akan muncul seperti berikut :

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

Figure 3.5: (e)

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

(g) Selanjutnya buat 2 matrix of features yang berisi values dari independent variable dan dependent variable.

(h) Lalu tuliskan perintah berikut :

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

Figure 3.6: (h)

(i) Perintah yang telah dibuat di atas akan membuat sebuah global environment baru dan muncul dataset.

(j) Klik dataset tersebut maka muncul tabel berisi dataset.

3. Cross Validation

- Pengertian Cross Validation :

Cross Validation adalah sebuah teknik validasi model yang digunakan untuk menilai bagaimana hasil analisis statistik akan digeneralisasi ke kumpulan data independen. Cross validation digunakan dengan tujuan prediksi, dan bila kita ingin memperkirakan seberapa akurat model prediksi yang dilakukan dalam sebuah praktik. Tujuan dari cross validation yaitu untuk mendefinisikan dataset guna menguju dalam fase pelatihan untuk membatasi masalah seperti overfitting dan underfitting serta mendapatkan wawasan tentang bagaimana model akan digeneralisasikan ke set data independen.

4. Penjelasan / Maksud Dari Score pada :

- Random forest (44%)

Maksud arti score 44% pada random forest adalah hasil dari akurasi. Yang menggunakan 5 buah atribut yaitu dari 5 baris pertama dari set pelatihan yang akan memprediksi spesies 10, 28, 156, 10 dan 43.

- Decision Tree (27%)

Maksud arti score 27% pada decision tree adalah presentasi hasil dari perhitungan dataset. Dari set tentang burung pipit. Confusion matrix memberi tau hal-hal yang diharapkan, artinya, burung-burung yang terlihat mirip saling bingung satu sama lain.

- SVM (29%)

Maksud arti score 29% dari SVM adalah hasil pendekatan jaringan saraf. Di sini, akurasinya adalah 27%, yang kurang dari akurasi 44% sebelumnya. Oleh karena itu, dessicion tree menjadi lebih buruk. Jika kita menggunakan Support Vector Machine (SVM), yang merupakan neural pendekatan jaringan, outputnya 29%. Jadi 29% pada SVM merupakan hasil outputannya.

Hasil tersebut didapat dari hasil validasi silang untuk memastikan bahwa membagi training test dengan cara yang berbeda. Sehingga didapat outputnya 44% untuk hutan acak, 27% untuk pohon keputusan, dan 29% untuk SVM.

5. Confusion Matrix Dan Ilustrasinya

- Cara Membaca Confusion Matrix :

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

$$a = 4$$

$$b = 1$$

$$c = 1$$

$$d = 2$$

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

$$\text{Recall} = 2/(1+2) = 0,66$$

$$\text{Precision} = 2/(1+2) = 0,66$$

$$\text{Accuracy} = (4+2)/(4+1+1+2) = 0,75$$

$$\text{Error Rate} = (1+1)/(4+1+1+4) = 0,2$$

Ilustrasi Confusion Matrix :

6. Voting Random Forest Dan Ilustrasi Gambarnya.

- Pengertian Voting pada Random Forest :

Metode ensemble dapat mencapai akurasi tinggi dengan membangun beberapa pengklasifikasi dan menjalankan masing-masing secara mandiri.

Definisi	IPK	Psikologi	Wawancara
P1	Bagus	Tinggi	Baik
P2	Bagus	Sedang	Baik
P3	Bagus	Sedang	Buruk
P4	Cukup	Rendah	Buruk
P5	Cukup	Tinggi	Baik
P6	Cukup	Sedang	Baik
P7	Cukup	Sedang	Buruk
P8	Kurang	Rendah	Buruk
P9	Kurang	Tinggi	Baik
P10	Kurang	Sedang	Buruk
P11	Kurang	Rendah	Baik

Figure 3.7: Confussion Matrik

Ketika classifier membuat sebuah keputusan, kamu dapat memanfaatkan yang terbaik keputusan umum dan rata-rata. Jika kita menggunakan metode yang paling umum, itu disebut voting.

- Ilustrasi Gambar Voting Random Forest :

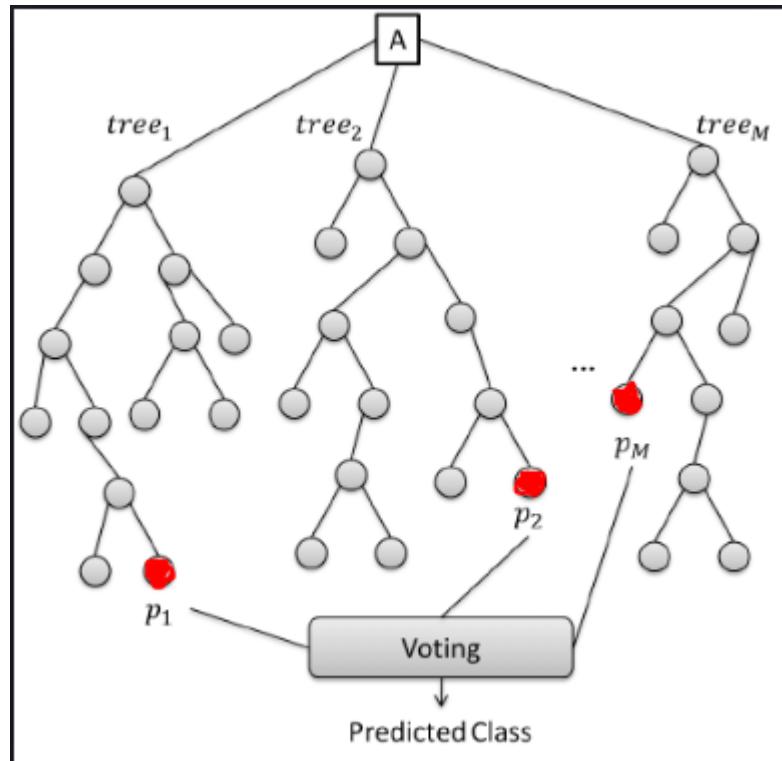


Figure 3.8: Voting Random forest

3.4.2 Praktek

1. Aplikasi Sederhana Menggunakan Pandas

```
1 import pandas as pd
2 ufo = pd.read_csv('http://bit.ly/uforeports')
3 type(ufo)
4 ufo.head()
```

Figure 3.9: Aplikasi Pandas

Penjelasan kodingan :

- Memanggil library.
- Membaca dari file pandas.
- Menampilkan hasil

Sehingga menghasilkan :

```
In [24]: import pandas as pd
...: ufo = pd.read_csv('http://bit.ly/uforeports')
...: type(ufo)
...: ufo.head()
Out[24]:
   City Colors Reported ... State      Time
0    Ithaca      NaN     ...    NY 6/1/1930 22:00
1  Willingboro  NaN     ...    NJ 6/30/1930 20:00
2   Holyoke      NaN     ...    CO 2/15/1931 14:00
3   Abilene      NaN     ...    KS 6/1/1931 13:00
4 New York Worlds Fair  NaN     ...    NY 4/18/1933 19:00
[5 rows x 5 columns]
In [25]:
```

Figure 3.10: Hasil Pandas

2. Aplikasi Sederhana Menggunakan Numpy

```
:> import numpy as np
:> a = np.array([2,3,4])
:> a
:> a = np.arange(1, 12, 2)
:> a
:> a = np.linspace(1, 12, 6)
:> a
```

Figure 3.11: Aplikasi Numpy

Penjelasan kodingan :

- Memanggil library
- Membuat variable dengan value linspace
- Menampilkan hasil value

Sehingga menghasilkan :

```
In [27]: import numpy as np
...: a = np.array([2,3,4])
...: a
...: a = np.arange(1, 12, 2)
...: a
...: a = np.linspace(1, 12, 6)
...: a
Out[27]: array([ 1.,  3.2,  5.4,  7.6,  9.8, 12.])
```

Figure 3.12: Hasil Numpy

3. Aplikasi Sederhana Menggunakan Matplotlib

```
:> import matplotlib.pyplot as plt
:> import numpy as np
:>
:> x = np.linspace(0, 10, 100)
:> plt.plot(x, np.sin(x))
:> plt.plot(x, np.cos(x))
:>
:> plt.show()
```

Figure 3.13: Aplikasi Matplotlib

Penjelasan kodingan :

- (a) Memanggil library
- (b) Membuat variable yang berisi bahasa pemrograman
- (c) Membuat variable dengan value
- (d) Membuat variable untuk menentukan garis
- (e) Membuat garis koordinat
- (f) Menampilkan hasil

Sehingga menghasilkan :

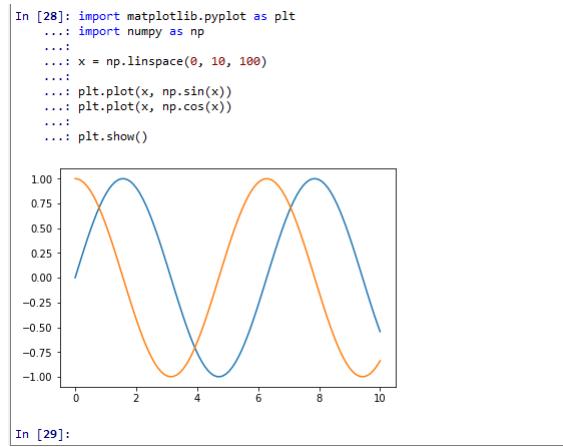


Figure 3.14: Hasil Matplotlib

4. Program Klasifikasi Random Forests

- Yang pertama dataset akan dibaca.

```
In [1]: import pandas as pd
...
...: # some Lines have too many fields (?), so skip bad lines
...: imgatt = pd.read_csv('D:/KULIAH/SEMESTER 6/KECERDASAN BUITAN/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.

In [2]:
```

Figure 3.15: Membaca Data File

- Selanjutnya sebagian data awal akan dilihat dengan menggunakan listing.

```

In [2]: imgatt.head()
Out[2]:
   imgid  attid  present
0       1      1        0
1       1      2        0
2       1      3        0
3       1      4        0
4       1      5        1

```

Figure 3.16: Melihat Data Sebagian

```

In [3]: imgatt.shape
Out[3]: (3677856, 3)
In [4]:

```

Figure 3.17: Melihat Jumlah Data

- Selanjutnya jumlah data dilihat dengan menggunakan listing.
- Lalu atribut diubah menjadi kolom dengan menggunakan perintah pivot.

```

In [4]: imgatt2 = imgatt.pivot(index='imgid', columns='attid', values='present')
In [5]:

```

Figure 3.18: Mengubah menjadi kolom

- Selanjutnya atribut yang telah diubah, sebagian data awalnya akan dilihat dengan menggunakan listing kembali.

```

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

```

Figure 3.19: Lihat sebagian data awal

- Selanjutnya atribut yang telah diubah, jumlah data dilihat dengan menggunakan listing kembali.

```

In [6]: imgatt2.shape
Out[6]: (11788, 312)
In [7]: |

```

Figure 3.20: Melihat jumlah data

- Lalu mengelompokkan burung kedalam spesies yang sama dengan dua kolom imgid dan label.

```
In [7]: imglabels = pd.read_csv("D:/KULIAH/SEMESTER 6/KECERDASAN BUATAN/CUB_200_2011/
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.
In [8]: |
```

Figure 3.21: Mengelompokkan burung

```
In [8]: imglabels.head()
Out[8]:
      label
imgid
1        1
2        1
3        1
4        1
5        1
In [9]: |
```

Figure 3.22: Melalukan pivot

- Lalu melakukan pivot dimana imgid menjadi index.
- Selanjutnya imgid, sebagian data awalnya akan dilihat dengan menggunakan listing untuk mengecek data.

```
In [9]: imglabels.shape
Out[9]: (11788, 1)
In [10]: |
```

Figure 3.23: Melihat data awal imgid

- Selanjutnya imgid, jumlah data dilihat dengan menggunakan listing untuk mengecek data.

```
In [10]: df = imgatt2.join(imglabels)
...: df = df.sample(frac=1)
In [11]: |
```

Figure 3.24: Melihat jumlah data imgid

- Lalu melakukan join karena isi datanya adalah sama di antara dua data. Sehingga mendapatkan data ciri labelnya sehingga bisa dikategorikan.
- Kemudian label yang didepan di drop dan berikan label pada data yang telah dilakukan join dengan perintah listing.
- Lalu cek kembali isinya dengan perintah listing.

```
In [11]: df_att = df.iloc[:, :312]
...: df_label = df.iloc[:, 312:]
In [12]:
```

Figure 3.25: Data ciri label dari join

```
In [12]: df_att.head()
Out[12]:
   1    2    3    4    5    6    7    ...   306   307   308   309   310   311   312
imgid
3218  0    0    0    0    0    0    0    ...   1     1    0    0    0    1    0
4781  0    1    0    0    0    0    0    ...   0     0    0    0    0    1    0
7610  0    0    0    0    0    0    0    ...   0     0    0    0    0    0    1
2327  0    0    0    0    0    0    0    ...   1     0    0    0    0    0    1
2462  0    0    0    0    0    0    0    ...   0     0    0    1    0    0    0
[5 rows x 312 columns]
In [13]: |
```

Figure 3.26: Mengubah menjadi kolom

```
In [13]: df_label.head()
Out[13]:
   label
imgid
3218    56
4781    82
7610   138
2327    41
2462    43
In [14]: |
```

Figure 3.27: Melihat isi data frame

- Kemudian data dibagi menjadi dua bagian, dimana 8000 row pertama merupakan data training dan sisanya adalah data testing.

```
In [14]: df_train_att = df_att[:600]
...: df_train_label = df_label[:600]
...: df_test_att = df_att[600:]
...: df_test_label = df_label[600:]
...: df_train_label = df_train_label['label']
...: df_test_label = df_test_label['label']
In [15]: |
```

Figure 3.28: Membagi data

- Kelas random forest selanjutnya dipanggil dengan RandomForestClassifier, dengan banyak kolom yang telah ditentukan oleh max feature.

```
In [15]: from sklearn.ensemble import RandomForestClassifier
...: clf = RandomForestClassifier(max_features=20, random_state=0, n_estimators=100)
In [16]: |
```

Figure 3.29: Kelas Random Forest

- Kemudian untuk membangun random forest dilakukan perintah fitting dengan maksimum fitur sebanyak 50.

```
In [16]: clf.fit(df_train_att, df_train_label)
Out[16]:
RandomForestClassifier(bootstrap=True, class_weight=None, criterion='gini',
                       max_depth=None, max_features=20, 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)

In [17]: |
```

Figure 3.30: Membangun Random forest

```
In [17]: print(clf.predict(df_train_att.head()))
[ 56  82 130  41  43]
```

Figure 3.31: Melihat hasil

- Kemudian lihat hasilnya dengan perintah predict.
- Lalu akan terlihat hasil score dari klasifikasi.

```
In [18]: clf.score(df_test_att, df_test_label)
Out[18]: 0.21290668573471577

In [19]: |
```

Figure 3.32: Lihat hasil score

5. Program Klasifikasi Confusion Matrix

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

```
In [19]: from sklearn.metrics import confusion_matrix
...: pred_labels = clf.predict(df_test_att)
...: cm = confusion_matrix(df_test_label, pred_labels)

In [20]: |
```

Figure 3.33: Memetakan ke confusion matrix

- Lalu melihat hasilnya.

```
In [20]: cm
Out[20]:
array([[ 7,  3,  0, ...,  0,  0,  0],
       [ 4, 26,  0, ...,  0,  0,  0],
       [ 1,  2,  0, ...,  0,  0,  0],
       ...,
       [ 0,  0,  0, ...,  5,  0,  0],
       [ 0,  0,  0, ...,  1,  0,  0],
       [ 0,  0,  0, ...,  0,  0, 19]], dtype=int64)

In [21]: |
```

Figure 3.34: Melihat hasil

- Kemudian dilakukan perintah plot.

```
In [21]: import matplotlib.pyplot as plt
...: import iterools
...: 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 iterools.product(range(cm.shape[0]), range(cm.shape[1])):
...:         plt.text(j, i, format(cm[i, j], fmt),
...:                  horizontalalignment="center",
...:                  verticalalignment="center")
```

Figure 3.35: Melakukan Plot

```
In [22]: birds = pd.read_csv("D:/KULIAH/SEMESTER 6/KECERDASAN BUATAN/CUB_200_2011/
classes.txt",
...:                         sep='\s+', header=None, usecols=[1], names=['birdname'])
...: birds = birds['birdname']
...: birds
Out[22]:
0          001.Black_footed_Albatross
1          002.Laysan_Albatross
2          003.Sooty_Albatross
3          004.Groove_billed_Ani
4          005.Crested_Auklet
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
...:          ...: 117.Wirebird
```

Figure 3.36: Plotting nama data

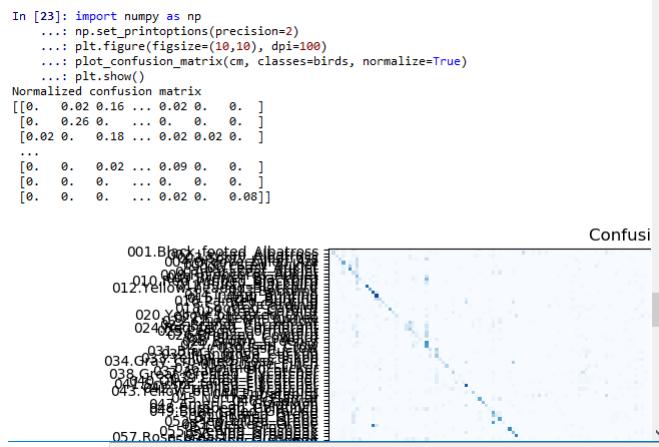


Figure 3.37: Melakukan perintah plot

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

6. Program Klasifikasi SVM dan Decision Tree

(a) Program Decision Tree

Mengklasifikasikan dataset yang sama menggunakan decision tree.

```
In [24]: from sklearn import tree
...: clftree = tree.DecisionTreeClassifier()
...: clftree.fit(df_train_att, df_train_label)
...: clftree.score(df_test_att, df_test_label)
Out[24]: 0.11807293528780836
In [25]: |
```

Figure 3.38: Klasifikasi menggunakan decision tree

(b) Program Klasifikasi SVM

Mengklasifikasikan dataset yang sama menggunakan SVM.

```
In [25]: 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[25]: 0.005809796210225242
In [26]: |
```

Figure 3.39: Klasifikasi menggunakan SVM

7. Program Cross Validation

- Melakukan pengecekan cross validation untuk random forest.

```
In [26]: 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))
C:\ProgramData\Anaconda3\lib\site-packages\sklearn\model_selection\_split.py:652: Warning:
The least populated class in y has only 1 members, which is too few. The minimum number of
members in any class cannot be less than n_splits=5.
% (min_groups, self.n_splits)), Warning)
Accuracy: 0.27 (+/- 0.13)
```

Figure 3.40: Pengecekan cross validation random forest

- Melakukan pengecekan cross validation untuk decision tree.
- Melakukan pengecekan cross validation untuk SVM.

8. Program Pengamatan Komponen Informasi

```
In [27]: scoretree = cross_val_score(clftree, df_train_att, df_train_label, cv=5)
...:     ...: print("Accuracy: %0.2f (+/- %0.2f)" % (scoretree.mean(), scoretree.std() * 2))
C:\ProgramData\Anaconda3\lib\site-packages\sklearn\model_selection\_split.py:652: Warning:
The least populated class in y has only 1 members, which is too few. The minimum number of
members in any class cannot be less than n_splits=5.
% (min_groups, self.n_splits)), Warning)
Accuracy: 0.11 (+/- 0.09)
```

Figure 3.41: Pengecekan cross validation decision tree

```
...: print("Accuracy: %0.2f (+/- %0.2f)" % (scoretree.mean(), scoretree.std() * 2))
C:\ProgramData\Anaconda3\lib\site-packages\sklearn\model_selection\_split.py:652: Warning:
The least populated class in y has only 1 members, which is too few. The minimum number of
members in any class cannot be less than n_splits=5.
% (min_groups, self.n_splits)), Warning)
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)
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: 0.03 (+/- 0.04)
```

Figure 3.42: Pengecekan cross validation SVM

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

```
The least populated class in y has only 1 members, which is too few. The minimum number of
members in any class cannot be less than n_splits=5.
% (min_groups, self.n_splits)), Warning)
Max features: 9, num estimators: 18, accuracy: 0.17 (+/- 0.10)
C:\ProgramData\Anaconda3\lib\site-packages\sklearn\model_selection\_split.py:652: Warning:
The least populated class in y has only 1 members, which is too few. The minimum number of
members in any class cannot be less than n_splits=5.
% (min_groups, self.n_splits)), Warning)
Max features: 9, num estimators: 22, accuracy: 0.17 (+/- 0.08)
C:\ProgramData\Anaconda3\lib\site-packages\sklearn\model_selection\_split.py:652: Warning:
The least populated class in y has only 1 members, which is too few. The minimum number of
members in any class cannot be less than n_splits=5.
% (min_groups, self.n_splits)), Warning)
Max features: 9, num estimators: 26, accuracy: 0.21 (+/- 0.14)
C:\ProgramData\Anaconda3\lib\site-packages\sklearn\model_selection\_split.py:652: Warning:
The least populated class in y has only 1 members, which is too few. The minimum number of
members in any class cannot be less than n_splits=5.
% (min_groups, self.n_splits)), Warning)
Max features: 9, num estimators: 30, accuracy: 0.18 (+/- 0.10)
C:\ProgramData\Anaconda3\lib\site-packages\sklearn\model_selection\_split.py:652: Warning:
The least populated class in y has only 1 members, which is too few. The minimum number of
members in any class cannot be less than n_splits=5.
% (min_groups, self.n_splits)), Warning)
Max features: 9, num estimators: 34, accuracy: 0.19 (+/- 0.12)
C:\ProgramData\Anaconda3\lib\site-packages\sklearn\model_selection\_split.py:652: Warning:
The least populated class in y has only 1 members, which is too few. The minimum number of
members in any class cannot be less than n_splits=5.
% (min_groups, self.n_splits)), Warning)
Max features: 9, num estimators: 38, accuracy: 0.21 (+/- 0.12)
```

In [30]:

Figure 3.43: Pengamatan Komponen

- Melakukan plot informasi agar bisa dibaca.

3.4.3 Penanganan Error

1. Skrinsut Error

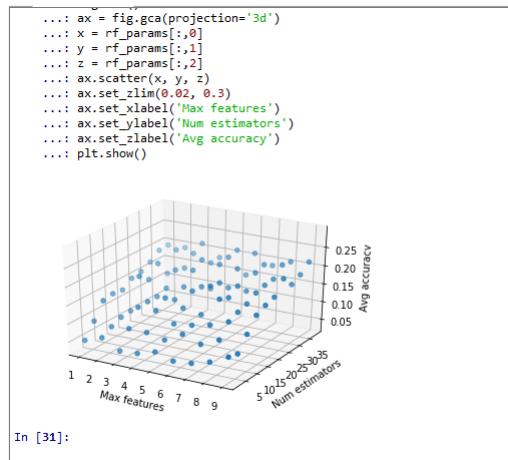


Figure 3.44: Plot informasi

```
FileNotFoundException: File b'data/CUB_200_2011/attributes/image_attribute_labels.txt' does not exist
```

Figure 3.45: Skrinsut Error

2. Tuliskan kode eror dan jenis errornya

Kode Error = FileNotFoundError: File b'data/CUB 200 2011/attributes/image attribute labels . txt' does not exist

Jenis Error = File not found

3. Solusi Pemecahan Masalah Error

Solusi dari error yang terjadi pada nomor 1 adalah perbaiki alamat direktoriya sebagai berikut :

```
# some lines have too many fields (?), so skip bad lines
imgatt = pd.read_csv('D:/KULIAH/SEMESTER 6/KECERDASAN BUATAN/CUB_200_2011/attributes/image_attribute_labels.txt',
                     sep='\t+', header=None, error_bad_lines=False, warn_bad_lines=False,
                     usecols=[0,1,2], names=['imgid', 'atid', '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 3.46: Penyelesaian

Sehingga didapat hasil seperti berikut :

```

In [1]: import pandas as pd
...
...: # some Lines have too many fields (?), so skip bad Lines
...: imgatt = pd.read_csv('D:/KULIAH/SEMESTER 6/KECERDASAN BUATAN/CUB_200_2011/
...: attributes/image_attribute_labels.txt',
...:                     sep='\s+', header=None, error_bad_lines=False,
...: warn_bad_lines=False,
...:                     usecols=[0,1,2], names=['imgid', 'attid', 'present'])
...: # 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.

In [2]:

```

Figure 3.47: Hasil

3.5 Andi Muhammad Aslam/1164064

1. Random Forest merupakan algoritma yang digunakan terhadap klasifikasi data dalam jumlah yang besar. Klasifikasi pada random forest dilakukan dengan penggabungan dicision tree dengan melakukan training terhadap sempel data yang dimiliki. Pembentukan decision tree menggunakan sample data berupa variable secara acak lalu menjalankan klasifikasi pada semua tree yang terbentuk. Random forest berupa Decision Tree agar dapat melakukan proses seleksi. Decision tree yang di buat dibagi secara strategis dari data pada kelas yang sama. Pemecahan digunakan untuk membagi data berdasarkan jenis atribut yang digunakan.. ??
2. Download dataset terdahulu kemudian buka software spyder untuk melihat isi dataset. Data yang di download berupa extensi file bernama .txt yang terdapat class dari field. Contohnya pada data jenis burung memiliki file index dan angka, dimana index berisi angka yang memiliki makna berupa jenis burung atau bahkan nama burung sedangkan field memiliki isi nilai berupa 0 dan 1 yang dimana sifatnya boolean, Ya dan Tidak. Hal ini dikarenakan komputer hanya dapat membaca bilangan biner maka dari itu field yang di isikan berupa angka. Artinya angka 0 berarti tidak dan angka 1 berarti Ya.
3. Cross validation adalah metode statistik yang digunakan untuk memperkirakan keterampilan model pembelajaran mesin. Ini biasanya digunakan dalam pembelajaran mesin yang diterapkan untuk membandingkan dan memilih model untuk masalah pemodelan prediktif yang diberikan karena mudah dipahami, mudah diimplementasikan, dan menghasilkan estimasi keterampilan yang umumnya memiliki bias lebih rendah daripada metode lainnya.

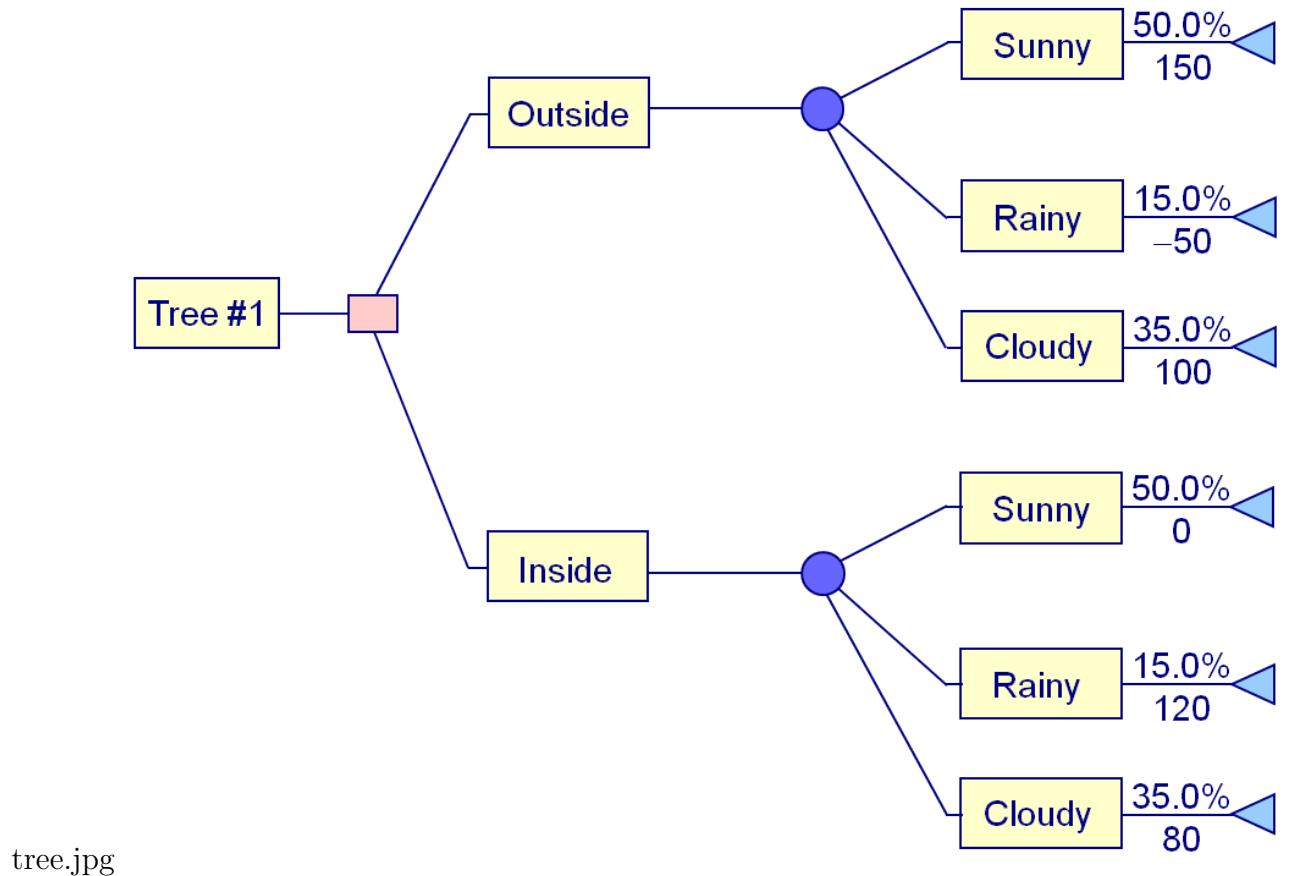


Figure 3.48: Random Forest.

4. Penjelasan Score

- Pada score 44% pada random forest berupa hasil akurasi.
- Pada score 27% pada decision tree adalah presentasi hasil dari perhitungan dataset.
- Pada score 29% dari SVM adalah hasil pendekatan neural network.
- Hasil tersebut didapat dari hasil validasi silang untuk memastikan bahwa membagi training test dengan cara yang berbeda. Sehingga dapat diketahui hasil output yaitu 44% untuk hutan, 27% untuk pohon keputusan, dan 29% untuk SVM.

5. Untuk membaca confusion matriks dapat menggunakan source code berikut :

```

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()
```

Dimana numpy dapat mengelola data yang berhubungan pada matrix. Pada perintah code tersebut digunakan dalam melakukan read pada dataset burung dengan menggunakan metode confusion matrix. Dalam confusion matrix memiliki 4 istilah yaitu True Positive yang merupakan data posotif yang terditeksi benar, True Negatif yang merupakan data negatif akan tetapi terditeksi benar, False Positif merupakan data negatif namun terditeksi sebagai data positif, False Negatif merupakan data posotif namun terditeksi sebagai data negatif. Adapun contoh hasil read dataset menggunakan confusion matrix dapat dilihat pada figure ??

- Untuk mengetahui confusion matriks kita dapat melihat contoh klasifikasi dari biner berikut ini :

	Predicted: NO	Predicted: YES
Actual: NO	TN = ??	FP = ??
Actual: YES	FN = ??	TP = ??

Figure 3.49: Tabel Confusion Matriks

- Voting merupakan proses pemilihan dari tree yang dimana akan dimunculkan hasilnya dan disimpulkan menjadi informasi yang pasti.

Pada figure Voting terdapat Decision Tree yang terbagi menjadi 3 Branch yaitu tree 1, tree 2, dan tree 3. Pada tree tersebut akan dilakukan proses voting. Pada masing-masing tree tersebut memiliki data-data yang berbeda, yang di

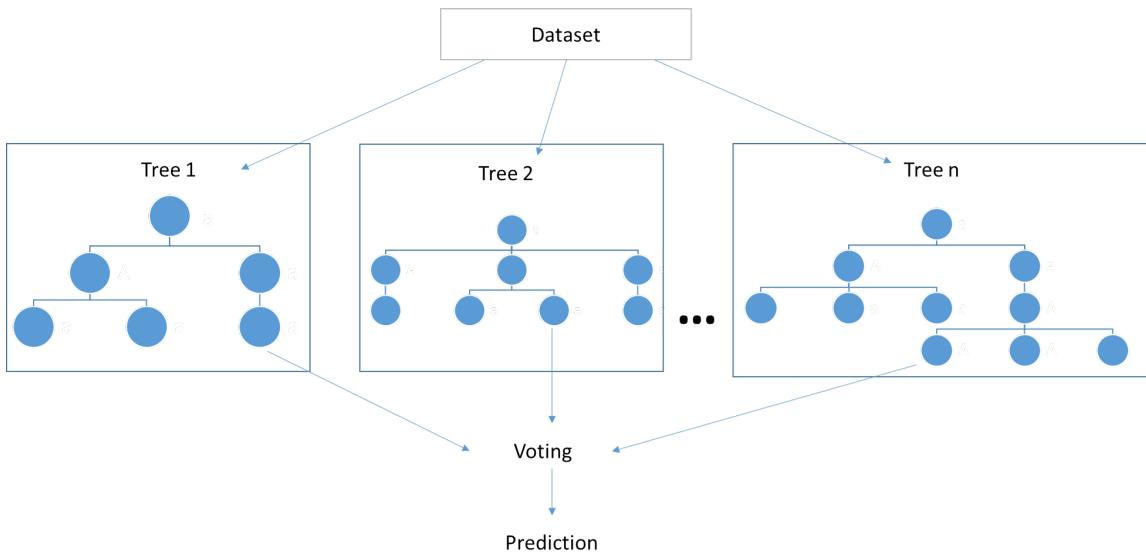


Figure 3.50: Voting

mana data tersebut akan di pilih dengan cara voting. Hasil voting dari setiap tree tersebut menunjukkan data pada setiap tree. Di sini kita dapat menghitung akurasi dengan menambahkan angka secara diagonal, sehingga ini semua adalah contoh yang diklasifikasikan dengan benar, dan membagi jumlah tersebut dengan jumlah semua angka dalam matriks.

3.5.1 Praktek

1. Aplikasi Sederhana Pandas dan Penjelasan Code

- Pandas:
 - Penjelasan Baris 1 : import pandas as Meong merupakan import dari library pandas dari python menggunakan inisial Meong.
 - Penjelasan Baris 2 : variabel a sebagai pendefinisian data yang dibuat pada baris pertama yaitu Meong, sedangkan Series merupakan sebuah array satu dimensi yang memiliki label dan digunakan untuk menyimpan data yang beragam seperti integer, string, float, dan lain sebagainya. angka pada Series yang saya gunakan yaitu ([10,20,30,40,np.nan,50]) untuk menampilkan secara default, pandas secara otomatis memberi index pada setiap baris dari series.
 - Penjelasan Baris 3 : fungsi print(am) untuk mencetak atau menampilkan objek

```

In [33]: import pandas as Meong
...: am = Meong.Series([10,20,30,40,np.nan,50])
...: print (am)
0    10.0
1    20.0
2    30.0
3    40.0
4    NaN
5    50.0
dtype: float64

In [34]:

```

Figure 3.51: Pandas

- Hasil:
- Aplikasi Sederhana Numpy dan Penjelasan Code
 - Code Numpy:
 - * Penjelasan Baris 1 : import numpy as np. untuk import library numpy dari python menggunakan inisial Kucing.
 - * Penjelasan Baris 2 : matrix one np.eye3. matrix dapat dikatakan sebagai list dua dimensi dimana suatu list berisi list lagi. sedangkan eye 3 untuk menghasilkan matriks identitas dengan dimensi yang ditetapkan.
 - * Penjelasan Baris 3 : matrix one. matrix pada baris ketiga ini untuk menampilkan objek numpy.
 - Hasil:

```

In [34]: import numpy as Kucing
...: matrix_one = np.eye(3)
...: matrix_one
Out[34]:
array([[1., 0., 0.],
       [0., 1., 0.],
       [0., 0., 1.]])

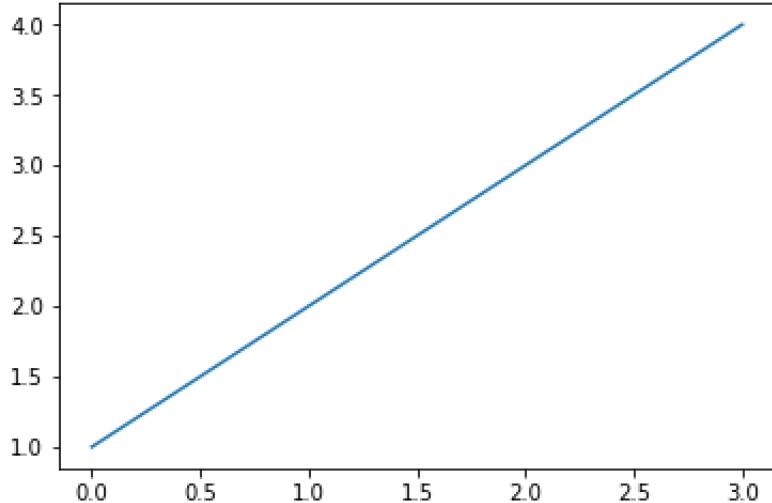
```

In [35]:

Figure 3.52: Numpy

- Aplikasi Sederhana Matplotlib dan Penjelasan Code
 - * Code Matplotlib:
 - Penjelasan Baris 1 : import matplotlib.pyplot as plt merupakan modul python untuk menggambarkan plot 2D.
 - Penjelasan Baris 2 : plt.plot([1,2,3,4]) untuk menentukan angka-angka pada gambar plot 2D tersebut

- Penjelasan Baris 3 : untuk menampilkan matplotlib berupa gambar plot 2D
- * Hasil:



In [36]:

Figure 3.53: Matplotlib

- Program Klasifikasi Random Forest dan Penjelasan :
 - * Code Random Forest 1 :

```
In [45]: import pandas as pd
...
...: # some Lines have too many fields (?), so skip bad Lines
...: imgatt = pd.read_csv("D:/Tugas Kuliah/Semester 6/KECERDASAN BUATAN/git/
hust/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'])
```

Figure 3.54: Gambar1

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

```
In [46]: imgatt.head()
Out[46]:
   imgid  attid  present
0      1      1      0
1      1      2      0
2      1      3      0
3      1      4      0
4      1      5      1
```

Figure 3.55: Gambar2

```
In [47]: imgatt.shape
Out[47]: (3677856, 3)

In [48]:
```

Figure 3.56: Gambar3

- 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 Random Forest 4 :

```
In [48]: imgatt2 = imgatt.pivot(index='imgid', columns='attid',
values='present')

In [49]:
```

Figure 3.57: Gambar 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 Random Forest 5 :
 - Penjelasan Pada gambar di atas menampilkan hasil dari variabel imgatt2.head. Dimana dataset nya ada 5 baris dan 312 kolom.
- * Code Random Forest 6 :
 - Penjelasan Pada gambar di atas menampilkan jumlah dari baris dan kolom dari variabel imgatt2. Dimana 11788 adalah baris dan 312 adalah kolom.
- * Code Random Forest 7 :
 - Penjelasan Pada gambar di atas menunjukkan load dari jawa-bannya yang berisi ” apakah burung tersebut (subjek pada dataset) termasuk dalam spesies yang mana ?. Kolom yang

```

In [49]: imgatt2.head()
Out[49]:
   attid  1    2    3    4    5    6    7    ...    306   307   308   309   310   311   312
   ...
1        0    0    0    0    1    0    0    ...    0    0    1    0    0    0    0
2        0    0    0    0    0    0    0    ...    0    0    0    0    0    0    0
3        0    0    0    0    0    1    0    0    ...    0    0    1    0    0    1    0
4        0    0    0    0    0    1    0    0    ...    1    0    0    1    0    0    0
5        0    0    0    0    0    1    0    0    ...    0    0    0    0    0    0    0

[5 rows x 312 columns]

```

Figure 3.58: Gambar 5

```

In [50]: imgatt2.shape
Out[50]: (11788, 312)

```

In [51]:

Figure 3.59: Gambar 6

digunakan adalah imgid dan label, kemudian melakukan pivot yang mana imgid menjadi index yang artinya unik sehubungan dengan dataset yang telah dieksekusi.

* Code Random Forest 8 :

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

* Code Random Forest 9 :

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

* Code Random Forest 10 :

- Penjelasan Pada gambar diatas dikarenakan isinya sama, maka bisa melakukan join antara dua data yang diesekusi (yaitu ada imgatt2 dan imglabels), sehingga pada hasilnya akan didapatkan data ciri dan data 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 Random Forest 11 :

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

```

In [51]: imglabels = pd.read_csv("D:/Tugas Kuliah/Semester 6/KECERDASAN BUATAN/
git/hust/CUB_200_2011/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.

```

In [52]:

Figure 3.60: Gambar 7

In [51]:	imglabels.head()
	Out[51]:
	label
	imgid
1	1
2	1
3	1
4	1
5	1

Figure 3.61: Gambar 8

* Code Random Forest 12 :

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

* Code Random Forest 13 :

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

* Code Random Forest 14 :

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

* Code Random Forest 15 :

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

* Code Random Forest 16 :

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

```
In [53]: imglabels.shape  
Out[53]: (11788, 1)
```

```
In [54]:
```

Figure 3.62: Gambar 9

```
In [54]: df = imgatt2.join(imglabels)
```

```
...: df = df.sample(frac=1)
```

```
In [55]:
```

Figure 3.63: Gambar 10

- * Code Random Forest 17 :
 - Penjelasan Pada gambar di atas menunjukkan hasil dari cetakan variabel dftrainatt.head.
- * Code Random Forest 18 :
 - Penjelasan Pada gambar di atas merupakan hasil dari variabel dftestatt da dftsetlabel. Dimana hasilnya dapat dilihat dari pada gambar di atas
- Program Aplikasi Confusion Matrix dan Penjelasan Keluarannya :
 - * Code Confusion Matrix 1 :
 - Penjelasan Pada gambar di atas merupakan kodingan untuk import confusiion matrik dari random forest. untuk hasilnya dapat dilihat dari gambar.
 - * Code Confusion Matrix 2 :
 - Penjelasan Pada gambar di atas merupakan tampilan dari variabel cm.
 - * Code Confusion Matrix 3 :
 - Penjelasan Pada gambar di atas merupakan perintah untuk plot. Dan untuk hasilnya terpadat pada gambar di atas.
 - * Code Confusion Matrix 5 :
 - Penjelasan Pada gambar di atas merupakan perintah plot dari gambar sebelumnya.
- Program Klasifikasi SVM dan Decision Tree Beserta Penjelasan Keluarannya :
 - * Code SVM :

```
In [55]: df_att = df.iloc[:, :312]
...: df_label = df.iloc[:, 312:]

In [56]:
```

Figure 3.64: Gambar 11

```
In [56]: df_att.head()
Out[56]:
   1    2    3    4    5    6    7    ...   306   307   308   309   310   311   312
imgid
7270    0    0    0    0    0    0    0 ...    0    0    0    0    0    1    0
10635    0    0    0    0    0    0    1 ...    0    0    0    1    0    0    0
2752     0    1    0    0    0    0    0 ...    0    0    0    0    0    0    0
2416     0    0    0    0    0    0    0 ...    0    0    0    0    0    1    0
8718     0    0    0    0    0    0    1 ...    0    0    1    0    0    1    0
[5 rows x 312 columns]
```

Figure 3.65: Gambar 12

- 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.
- 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 Decision 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.

```
In [57]: df_label.head()
Out[57]:
label

```

In [58]:

Figure 3.66: Gambar 13

```
In [58]: 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']
```

In [59]:

* 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

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

Figure 3.67: Gambar 15

```
Out[16]:
RandomForestClassifier(bootstrap=True, class_weight=None, criterion='gini',
max_depth=None, max_features=50, max_leaf_nodes=None,
min_impurity_decrease=0.0, min_impurity_split=None,
min_samples_leaf=1, min_samples_split=2,
min_weight_fraction_leaf=0.0, n_estimators=100, n_jobs=None,
oob_score=False, random_state=0, verbose=0, warm_start=False)
In [16]: clf.fit(df_train_att, df_train_label)Out[17]:
RandomForestClassifier(bootstrap=True, class_weight=None, criterion='gini',
max_depth=None, max_features=50, max_leaf_nodes=None,
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)

In [17]:
```

Figure 3.68: Gambar 16

```
In [17]:
In [18]: print(clf.predict(df_train_att.head()))
[183 57 139 108 28]
In [19]:
```

Figure 3.69: Gambar 17

```
In [19]: clf.score(df_test_att, df_test_label)
Out[19]: 0.44720168954593453
```

```
In [20]:
```

Figure 3.70: Gambar 18

```
In [20]: from sklearn.metrics import confusion_matrix
...: pred_labels = clf.predict(df_test_att)
...: cm = confusion_matrix(df_test_label, pred_labels)
```

```
In [21]:
```

```
In [21]: cm
Out[21]:
array([[ 9,  1,  0, ...,  0,  0,  0],
       [ 0, 15,  0, ...,  0,  0,  0],
       [ 3,  0,  7, ...,  0,  0,  0],
       ...,
       [ 0,  0,  0, ...,  2,  0,  0],
       [ 0,  0,  0, ...,  0,  5,  0],
       [ 0,  0,  0, ...,  0,  0, 12]], dtype=int64)
```

```
In [22]:
```

Figure 3.71: Gambar 20

```

In [22]: 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)

```

Figure 3.72: Gambar 21

```

In [27]: import numpy as np
.... np.set_printoptions(precision=2)
.... plt.figure(figsize=(10,10), dpi=100)
.... plot_confusion_matrix(cm, classes=birds, normalize=True)
.... plt.show()
Normalized confusion matrix
[[0.6  0.07 0.   ... 0.   0.   0.   ]
 [0.   0.58 0.   ... 0.   0.   0.   ]
 [0.18 0.   0.41 ... 0.   0.   0.   ]
 ...
 [0.   0.   0.   ... 0.1  0.   0.   ]
 [0.   0.   0.   ... 0.   0.29 0.   ]
 [0.   0.   0.   ... 0.   0.   0.8  ]]

```

Figure 3.73: Gambar 23

```

In [28]: from sklearn import tree
.... clftree = tree.DecisionTreeClassifier()
.... clftree.fit(df_train_att, df_train_label)
.... clftree.score(df_test_att, df_test_label)
Out[28]: 0.2663674762407603

In [29]:

```

Figure 3.74: SVM

```

In [27]: 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[27]: 0.27613516367476243

In [28]: |

```

Figure 3.75: Decission Tree

```
In [28]: 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))
C:\ProgramData\Anaconda3\lib\site-packages\sklearn\model_selection\_split.py:652:
Warning: The least populated class in y has only 1 members, which is too few. The
minimum number of members in any class cannot be less than n_splits=5.
% (min_groups, self.n_splits)), Warning)
Accuracy: 0.25 (+/- 0.12)
```

In [29]:

Figure 3.76: Cross Validation 1

```
In [29]: scorestree = cross_val_score(clftree, df_train_att, df_train_label, cv=5)
...: print("Accuracy: %0.2f (+/- %0.2f)" % (scorestree.mean(), scorestree.std() *
2))
C:\ProgramData\Anaconda3\lib\site-packages\sklearn\model_selection\_split.py:652:
Warning: The least populated class in y has only 1 members, which is too few. The
minimum number of members in any class cannot be less than n_splits=5.
% (min_groups, self.n_splits)), Warning)
Accuracy: 0.13 (+/- 0.06)
```

In [30]:

Figure 3.77: Cross Validation 2

```
In [30]: scoressvm = cross_val_score(clfsvm, df_train_att, df_train_label, cv=5)
...: print("Accuracy: %0.2f (+/- %0.2f)" % (scoressvm.mean(), scoressvm.std() * 2))
C:\ProgramData\Anaconda3\lib\site-packages\sklearn\model_selection\_split.py:652:
Warning: The least populated class in y has only 1 members, which is too few. The
minimum number of members in any class cannot be less than n_splits=5.
% (min_groups, self.n_splits)), Warning)
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
```

Figure 3.78: Cross Validation 3

```

In [31]: max_features_opts = range(1, 10, 1)
....: n_estimators_opts = range(2, 40, 4)
....: 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))
C:\ProgramData\Anaconda3\lib\site-packages\sklearn\model_selection\_split.py:652: Warning: The least populated class in y has only 1 members, which is too few. The minimum number of members in any class cannot be less than n_splits=5.
% (min_groups, self.n_splits)), Warning)
C:\ProgramData\Anaconda3\lib\site-packages\sklearn\model_selection\_split.py:652: Warning: The least populated class in y has only 1 members, which is too few. The

```

Figure 3.79: Program Pengamatan Komponen Informasi 1

```

In [32]: 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(0.02, 0.3)
....: ax.set_xlabel('Max features')
....: ax.set_ylabel('Num estimators')
....: ax.set_zlabel('Avg accuracy')
....: plt.show()

```



Figure 3.80: Program Pengamatan Komponen Informasi 2

```

In [59]: from sklearn.ensemble import RandomForestClassifier
....: clf = RandomForestClassifier(max_features=50, random_state=0,
n_estimators=100)

```

Figure 3.81: Error

Chapter 4

Experiment and Result

brief of experiment and result.

4.1 Experiment

Please tell how the experiment conducted from method.

4.2 Result

Please provide the result of experiment

4.3 Aip Suprapto Munari/1164063

4.3.1 Teori

1. Klasifikasi teks

Klasifikasi teks atau kategorisasi teks merupakan proses yang secara otomatis menempatkan dokumen teks ke dalam suatu kategori berdasarkan isi dari teks tersebut.

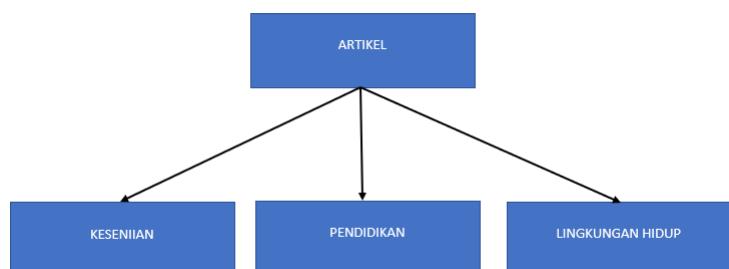


Figure 4.1: Aip-Klasifikasi teks

2. Klasifikasi Bunga tidak dapat penggunaan machine learning

Dikarenakan masalah dari input yang serupa namun output yang berbeda noise, yang dimaksud dengan noise adalah contoh pada output yang direkam bukan seperti perkiraan.

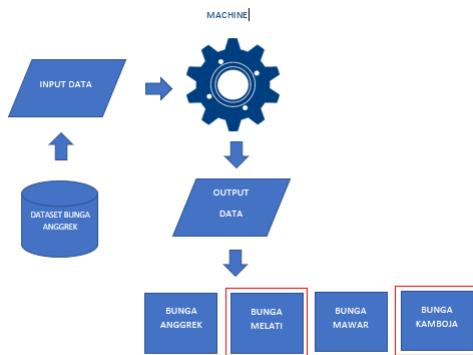


Figure 4.2: Aip-Klasifikasi bunga

3. Teknik pembelajaran mesin pada teks YouTube

Teknik Machine Learning pada YouTube memperhatikan apa saja yang menarik perhatian para penggunanya. Ketika kita sedang menonton di YouTube, pada sebelah kanan terdapat 'Up Next' yang menampilkan beberapa video serupa yang sedang ditonton. Dan ketika mengklik salah satu video dari baris tersebut, maka YouTube akan mengingatnya dan menggunakan kata yang tertera sebagai referensi.

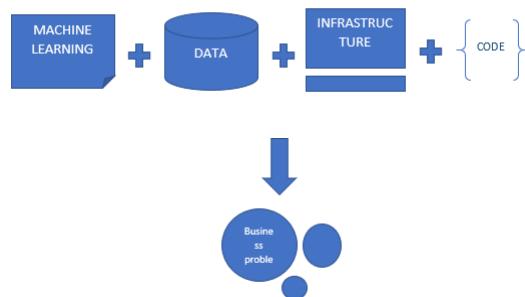


Figure 4.3: Aip-Teknik YouTube

4. Vectorisasi Data

- Vectorisasi Data merupakan pemecahan serta pembagian data kemudian dilakukan perhitungan datanya.

5. Bag of word

Bag of Words adalah metode untuk mengekstraksi fitur dari dokumen teks.

1. I love dogs.

2. I hate dogs and knitting.

3. Knitting is my hobby and my passion.

Figure 4.4: Aip-Bag of Word

6. TF-IDF

TF-IDF merupakan istilah beberapa frekuensi dokumen terbalik, adalah ukuran penilaian yang banyak digunakan dalam pengambilan informasi (IR) atau peringkasan.

Term (t)	df	idf
Akhir	1	$\log(4/1)=0.602$
Awal	1	$\log(4/1)=0.602$
Belajar	2	$\log(4/2)=0.301$
Dokumen	1	$\log(4/1)=0.602$
Frekuensi	1	$\log(4/1)=0.602$
Hitung	3	$\log(4/3)=0.125$
Idf	4	$\log(4/4)=0$
Kita	1	$\log(4/1)=0.602$
Langkah	2	$\log(4/2)=0.301$
Muncul	1	$\log(4/1)=0.602$
Saya	1	$\log(4/1)=0.602$
Term	1	$\log(4/1)=0.602$
Tf	4	$\log(4/4)=0$

Figure 4.5: Aip-TF IDF

4.4 BAGIAN PRAKTEK

4.5 Aip Suprapto Munari /1164063

4.5.1 Aplikasi Sederhana Menggunakan Pandas

Disini saya akan menggunakan Dataset dari <https://data.world/alexandra/generic-food-database> dan akan mengambil Data Dummy sebanyak 500 records dan membuat Dataframe baru.

```

import pandas as pd
gf = pd.read_csv('D:/KULIAH/SEMESTER 6/KECERDASAN BUATAN/PRAKTEK/Python-Artificial
df = pd.DataFrame(gf, columns = ['FOOD NAME', 'SCIENTIFIC NAME', 'GROUP', 'SUB GR
dummy = pd.get_dummies (df['test preparation course'])
dummy.head()

df = df.join(dummy)

```

Maksud dari kodingan diatas yaitu :

1. Baris pertama impor librari pandas dengan inisiasi pd
2. Definisikan variabel gf (generic-food) untuk membaca file csv dengan pandas
3. variabel df akan menggunakan function pd dataframe untuk membuat datafarme di pandas dari file CSV yang tadi.
4. Mendefinisikan variabel dummy untuk mengubah data categorical menjadi integer. dibawah merupakan data sebelum di Dummy.

Index	FOOD NAME	SCIENTIFIC NAME	GROUP	SUB GROUP
0	nan	nan	nan	nan
1	nan	nan	nan	nan
2	nan	nan	nan	nan
3	nan	nan	nan	nan
4	nan	nan	nan	nan
5	nan	nan	nan	nan
6	nan	nan	nan	nan
7	nan	nan	nan	nan
8	nan	nan	nan	nan
9	nan	nan	nan	nan
10	nan	nan	nan	nan
11	nan	nan	nan	nan
12	nan	nan	nan	nan
13	nan	nan	nan	nan

Figure 4.6: Dataset Original Aip

5. Atribut atau kolom yang ingin di Dummy yaitu test preparation course. Dalam test memunculkan index dan nama keseluruhan kolom.
6. kemudian df akan melakukan join dengan dataframe dummy.

Index	ENTIFIC NAME_GF
0	Angelica,Ang...
1	Savoy cabbage,Bras...
2	Silver linden,Tilia...
3	Kiwi,Actinid...
4	Allium (Onion),Alli...
5	Garden onion,Allium...
6	Leek,Allium porrum,Veget...
7	Garlic,Allium sativum,Herb...
8	Chives,Allium schoenoprasu...
9	Lemon verbena,Aloy...
10	Cashew nut,Anacardi...
11	Pineapple,An...
12	Dill,Anethum graveolens,H...
13	Custard apple,Annona...

Figure 4.7: Dataset Dummy Aip

4.5.2 Memecah DataFrame Menjadi 2 Dataframe

Dari dataframe tersebut dipecah menjadi dua dataframe yaitu 450 row pertama dan 50 row sisanya

```
gf_train= gf[:450]
gf_test= gf[451:]
```

1. gf train akan mendefinisikan dataframe untuk train dengan 450 data pertama
2. gf test mendefinisikan dataframe untuk test untuk data setelah 451. Hasilnya seperti berikut :

gf_test	DataFrame	(50, 1)
gf_train	DataFrame	(450, 1)

Figure 4.8: Split DataFrame Aip

4.5.3 Vektorisasi Dan Klasifikasi Dari Data Youtube05-Shakira Dengan Decision Tree

1. Ini hasil dari impor dataset
2. Ini hasil Setelah di Klasifikasikan dengan Decision Tree
3. Dalam in 13 impor Tree dari Sklearn. Dan mendefinisikan variabel clf untuk memanggil Decision Tree Classifier dan melakukan fit atau pengujian.

```
In [13]: import pandas as pd
...: d = pd.read_csv("D:/KULIAH/SEMESTER 6/KECERDASAN BUATAN/PRAKTEK/Python-Artificial-Intelligence-Projects-for-Beginners-master/Chapter03/Youtube05-Shakira.csv")
In [14]: from sklearn.feature_extraction.text import CountVectorizer
...: vectorizer = CountVectorizer()
```

Figure 4.9: Dataset Youtube05-Shakira Aip

```
In [13]: from sklearn import tree
...: clf = tree.DecisionTreeClassifier(criterion="entropy", max_depth=5)
...: clf = clf.fit(d_train_att, d_train_label)

In [14]: clf.predict(d_test_att)
Out[14]:
array([1, 0, 1, 1, 1, 0, 0, 0, 1, 0, 0, 0, 1, 0, 1, 0, 0, 0, 1, 0,
       0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 0, 1,
       0, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0,
       0, 0, 0], dtype=int64)

In [15]: clf.score(d_test_att, d_test_label)
Out[15]: 0.8428571428571429
```

Figure 4.10: Dataset Youtube05-Shakira Aip

4. Dalam In 14 menggunakan prediksi untuk clf dengan function predict untuk memprediksi test. Dan hasilnya muncul dalam bentuk array.
5. clf score memunculkan akurasi prediksi yang dilakukan terhadap clf.

4.5.4 Vektorisasi Dan Klasifikasi Dari Data Youtube05-Shakira Dengan SVM

```
In [19]: from sklearn import svm
...: clfsvm = svm.SVC()
...: clfsvm.fit(d_train_att, d_train_label)
...: clfsvm.score(d_test_att, d_test_label)
C:\ProgramData\Anaconda3\lib\site-packages\sklearn\svm\base.py
features. Set gamma explicitly to 'auto' or 'scale' to avoid t
"avoid this warning.", FutureWarning)
Out[19]: 0.7571428571428571
```

Figure 4.11: Dataset Youtube05-Shakira SVM Aip

Dari Gambar diatas dapat dijelaskan bahwa :

1. Impor SVM dari sklearn
2. Melakukan fit dari d train att dan d train label atau disebut dengan pengujian
3. Mendefinisikan variabel clf untuk melakukan prediksi dataset Youtube05-Shakira dengan SVM. Dan akan muncul hasil prediksinya

4.5.5 Vektorisasi Dan Klasifikasi Dari Data Youtube05-Shakira Dengan Decision Tree 2

Maksud dari codingan diatas yaitu, mengklasifikasikan Dataset Youtube05-Shakira dengan Decision Tree dengan melakukan prediksi menggunakan function test pada d test att, dan memberikan akurasi prediksi menggunakan prediksi score.

```

In [20]: from sklearn import tree
...: clf = tree.DecisionTreeClassifier(criterion="entropy", max_depth=5)
...: clf = clf.fit(d_train_att, d_train_label)
...: clf.predict(d_test_att)
...: clf.score(d_test_att,d_test_label)
Out[20]: 0.8428571428571429

```

Figure 4.12: Dataset Youtube05-Shakira Aip

4.5.6 Plotting Confusion Matrix

Berikut adalah skript dari plotting confusion matrix dari contoh yang ada pada bagian teori

```

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')

import numpy as np
np.set_printoptions(precision=2)
plt.figure(figsize=(60,60), dpi=300)
plot_confusion_matrix(cm, classes=clf, normalize=True)
plt.show()

```

Hasilnya adalah sebagai berikut : Dari gambar dapat dijelaskan bahwa data array

```

In [21]: import numpy as np
...: np.set_printoptions(precision=2)
...: plt.figure(figsize=(10,10), dpi=100)
...: plot_confusion_matrix(cm, classes=clf, normalize=True)
...: plt.savefig()
Normalized confusion matrix
[[0.98 0.02]
 [0.18 0.82]]

```

Figure 4.13: Confusion Matrix Aip

merupakan data asli dan data prediksi yang dilakukan dengan Random Forest. Dengan melakukan normalisasi data confusion matrix.

4.5.7 Menjalankan Program Cross Validation

```

In [11]: from sklearn.model_selection import cross_val_score
...: scores = cross_val_score(clf,d_train_att,d_train_label,cv=5)
...:
...: shakirarata2=scores.mean()
...: shakirasd=scores.std()

```

Figure 4.14: Cross Validation Aip

Gambar diatas akan dijelaskan seperti berikut :

1. Dari sklearn mengimpor Cross Validation
2. Variabel scores akan melakukan cross validation pada variabel clf, d train att , dan d train label

3. Variabel shakirarata2 akan menghitung nilai rata rata dari variabel scores tadi menggunakan function mean
4. shakirasd Menghitung standar deviasi dari data yang diberikan. Hasilnya seperti berikut :

```
In [13]: from sklearn import tree
...: clf = tree.DecisionTreeClassifier(criterion="entropy", max_depth=5)
...: clf = clf.fit(d_train_att, d_train_label)

In [14]: clf.predict(d_test_att)
Out[14]:
array([1, 0, 1, 1, 1, 0, 0, 0, 0, 1, 0, 0, 1, 0, 0, 1, 0, 0, 0, 1, 0,
       0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 1, 1, 1, 1, 0, 1,
       0, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0,
       0, 0, 0], dtype=int64)

In [15]: clf.score(d_test_att,d_test_label)
Out[15]: 0.8428571428571429
```

Figure 4.15: Hasil Cross Validation Aip

4.5.8 Program Pengamatan Komponen Informasi

```
In [22]: max_features_opts = range(1, 10, 1)
...: n_estimators_opts = range(2, 40, 4)
...: 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
```

Figure 4.16: Program Komponen Informas Aip

Dari gambar diatas dapat dijelaskan bahwa :

1. Max featuresnya dari range 1 sampai 10
2. n estimators dengan range 20 sampai 40
3. Variabel rf params berisikan function np empty dimana akan membuat array baru berisikan tipe yang didefinisikan dengan random value
4. Mendefinisika i dimulai dari angka 0 dimana max features dan n estimators menggunakan klasifikasi randomforestclassifier menggunakan data prediksi
5. Mendefinisikan rfparams untuk max features , n estimators, nilai rata dan std

4.6 Penanganan Error

AIP SUPRAPTO MUNARI 1164063

4.6.1 Error Index

1. Berikut ini merupakan eror yang didapatkan saat menjalankan program diatas

```
| KeyError: 'generic-food'
```

Figure 4.17: Error Key Aip

2. Pada gambar diatas kode erornya adalah KeyEror. Eror ini terjadi karena keyword yang dimasukan tidak ada.
3. Solusi yang bisa dilakukan untuk mengatasi eror tersebut adalah sebagai berikut

:

```
dummy = pd.get_dummies (df['generic-food'])
dummy.head()
```

Figure 4.18: Error Key Aip

- Pada gambar diatas Dataset Generic-Food tidak terdapat atribut COLOR, maka dari itu kita harus merubahnya dengan atribut yang terdapat di dataset tersebut. Mari kita gunakan atribut SCIENTIFIC NAME. Ubah skrip menjadi seperti berikut

```
dummy = pd.get_dummies (df['COLOR'])
dummy.head()
```

Figure 4.19: Error Key Aip

•

- Maka ketika di run akan muncul data dummy nya seperti berikut

4.7 Andi Muhammad Aslam/1164064

4.7.1 Teori

1. Klasifikasi teks

Klasifikasi merupakan kata serapan dari bahasa Belanda, classificatie, yang sendirinya berasal dari bahasa Prancis classification. Istilah ini menunjuk kepada sebuah metode untuk menyusun data secara sistematis atau menurut beberapa aturan atau kaidah yang telah ditetapkan. Di dalam KBBI, klasifikasi adalah

Index
0
1
2
3
4
5
6
7
8
9
10
11
12
13

Figure 4.20: Error Key Aip

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 persamaan dan perbedaan.

2. Klasifikasi Bunga tidak bisa menggunakan machine learning

Machine Learning tidak dapat mengklasifikasikan bunga, Karena 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 kata lain, 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 petalnya(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.

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.

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).

4.8 Andi Muhammad Aslam/1164064

4.8.1 Praktek Program

1. Pada penjelasan kali ini saya akan membuat data dummy dengan menggunakan format csv, data di ambil dari web UCI Machine Learning

Repository dengan nama file Youtube02-KatyPerry.csv

Penjelasan pada gambar G1 yaitu mengimport librari pandas dimana kita mengalaskan praktek sebagai pandas. Pandas berguna untuk mengelola dataframe = matrix = tabel kemudian memanggil nama alias untuk membaca format csvnya.

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

Dapat dilihat pada gambar ??

Penjelasan pada gambar G2 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 KetyPerry pada gambar G3 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.

Pada gambar G3 merupakan hasil output content dimana terdapat 448 baris/data yang mempunyai 1602 kata-kata yang digunakan pada komentar yang ada di content tersebut.

Pada gambar G4 maksud dari outputannya merupakan dataframe kata-kata tadi yang berjumlah 1602 kata orang yang komen pada data KetyPerry.

4. Mengklasifikasikan dari data vektorisasi dengan menggunakan klasifikasi svm(support vektorisasi machine) dapat dilihat pada gambar G6.

Jadi pada gambar G6 merupakan hasil dari memprediksi data score vektorisasi dengan svm menggunakan metode fit dimana digunakan untuk data training atau data pelatihannya.

5. Mengklasifikasikan dari data vektorisasi dengan menggunakan klasifikasi Decision Tree dapat dilihat pada gambar G7.

Jadi pada gambar G7 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 G8.

Pada gambar G8 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.

7. Menjalankan program cross validation pada data vektorisasi dapat dilihat pada gambar G9.

Pada gambar G9 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 G10

Pada gambar G10 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.

4.8.2 Penanganan Eror

1. Pada gambar G11 merupakan ScreeShootan dari data yang eror.

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\__i
    projection_class = get_projection_class(projection)
```

```
File "C:\ProgramData\Anaconda3\lib\site-packages\matplotlib\projections\__i
    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()
```

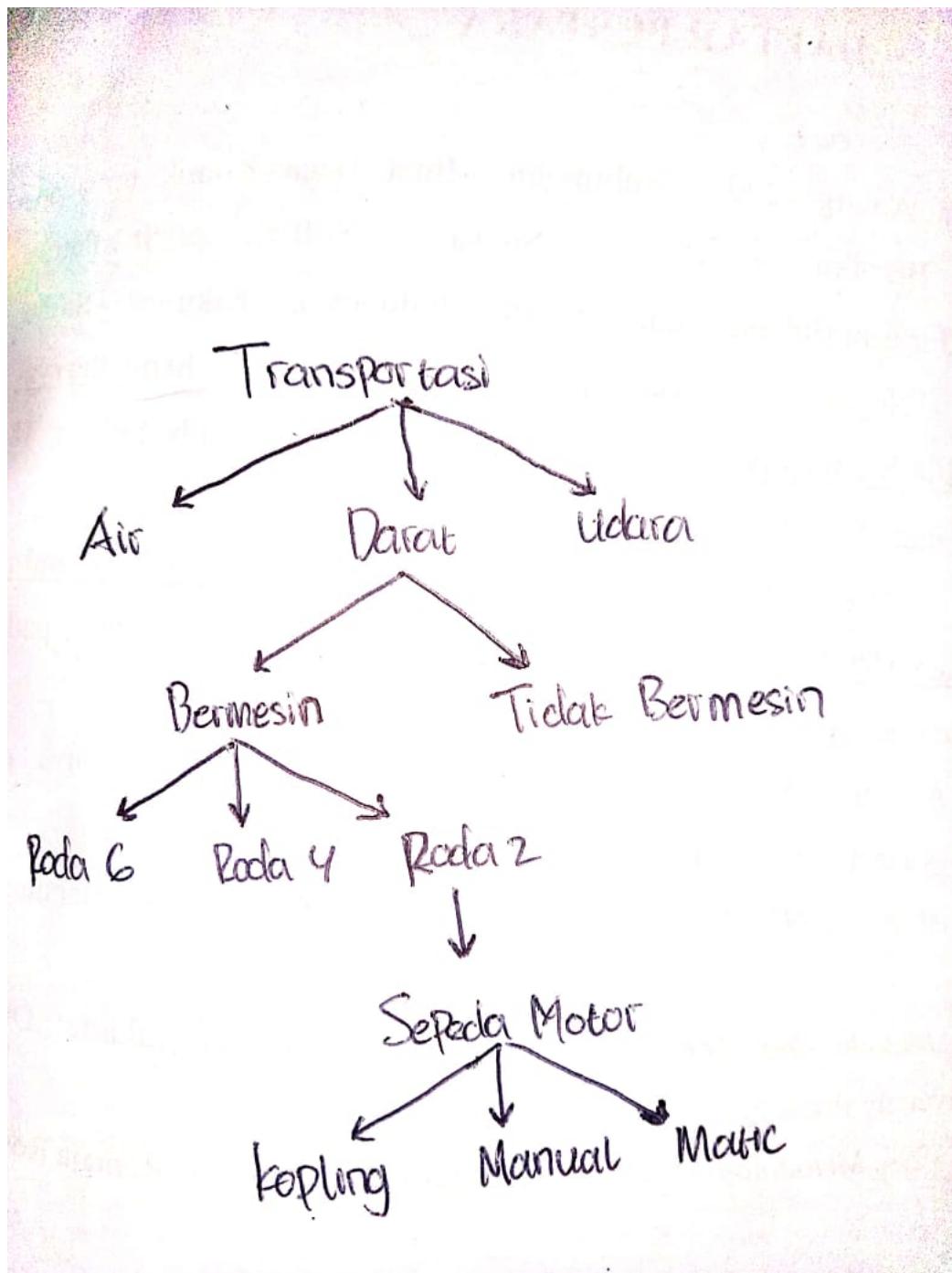


Figure 4.21: Klasifikasi teks



Figure 4.22: Klasifikasi bunga

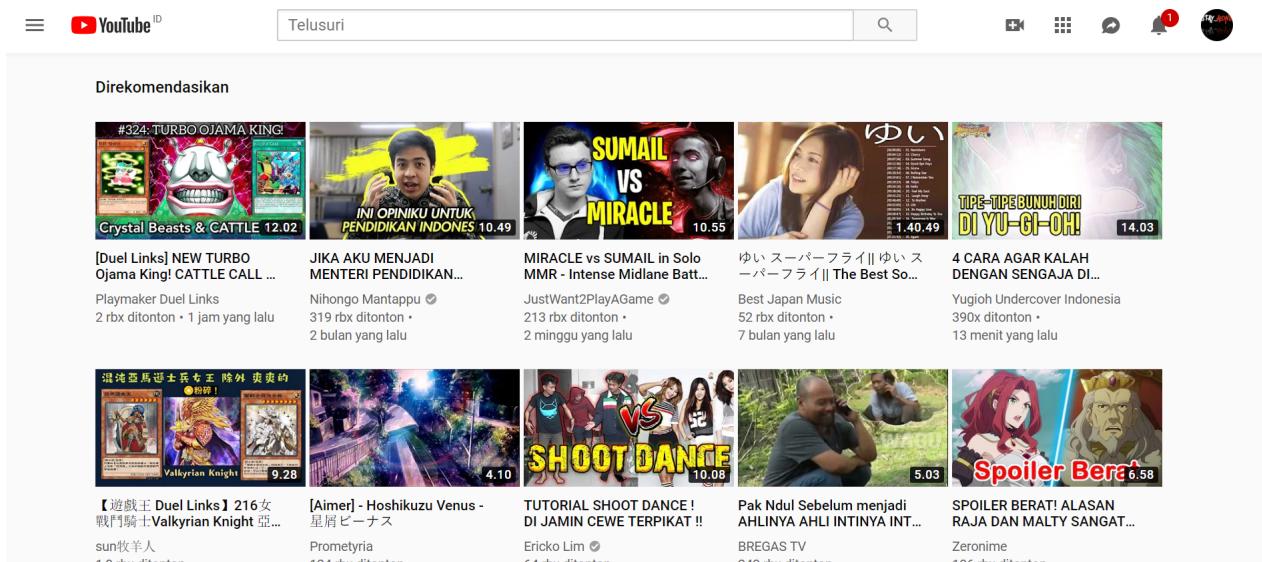


Figure 4.23: Teknik YouTube

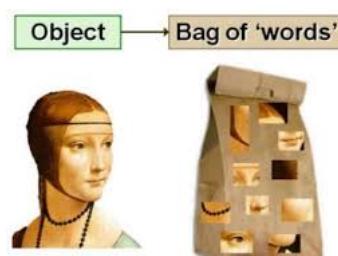


Figure 4.24: Bag of Word

Q	tf			$\frac{df}{D}$	IDF	IDF+1	W = tf * (IDF+1)		
	d1	d2	d3				d1	d2	d3
gold	1	0	1	2	1.5	0.176	1.176	1.176	0
silver	0	2	0	1	3	0.477	1.477	0	2.954
truck	1	1	1	2	1.5	0.176	1.176	0	1.176
						sum(d1)	sum(d2)	sum(d3)	
Nilai Bobot setiap Dokumen =							1.176	4.130	2.352

Figure 4.25: 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\
```

Figure 4.26: Data Dummy 500 Data

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

Variable explorer File explorer Help

IPython console

Console 1/A ✖

```
In [118]: yeay = andi.sample(frac=1)

In [119]: andi_train=yeay[:300]
...: andi_test=yeay[300:]

In [120]: |
```

Figure 4.27: 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 4.28: Vektorisasi dan Klasifikasi Data

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

Figure 4.29: Data Content

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

Figure 4.30: 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 4.31: Klasifikasi SVM Dari Data Vektorisasi

```
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
```

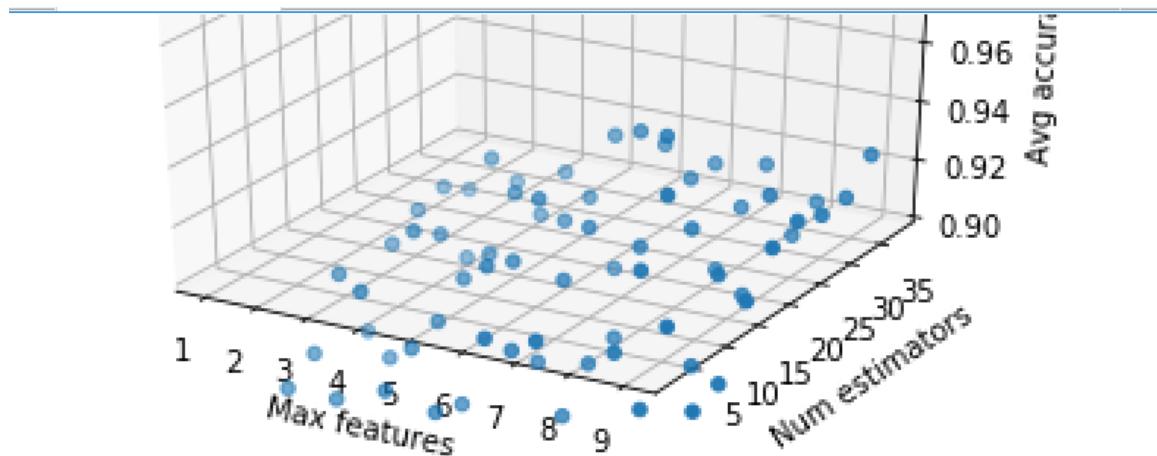
Figure 4.32: Klasifikasi Decision Tree Dari Data Vektorisasi

```
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 4.33: Plot Confusion Matrix Menggunakan Matplotlib

```
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 4.34: Program Cross Validation Pada Data Vektorisasi



```
In [180]: |
```

Figure 4.35: 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 4.36: Eror matplotlib.pyplot

Chapter 5

Conclusion

brief of conclusion

5.1 Conclusion of Problems

Tell about solving the problem

5.2 Conclusion of Method

Tell about solving using method

5.3 Conclusion of Experiment

Tell about solving in the experiment

5.4 Conclusion of Result

tell about result for purpose of this research.

5.5 Andi Muh Aslam/1164064

5.5.1 Teori

1. Jelaskan Kenapa Kata-Kata harus dilakukan vektorisasi lengkapi dengan ilustrasi gambar.

Kata-kata harus dilakukan vektorisasi untuk mengukur nilai kemunculan suatu kata agar dapat di prediksi atau untuk menentukan bobot suatu kata.

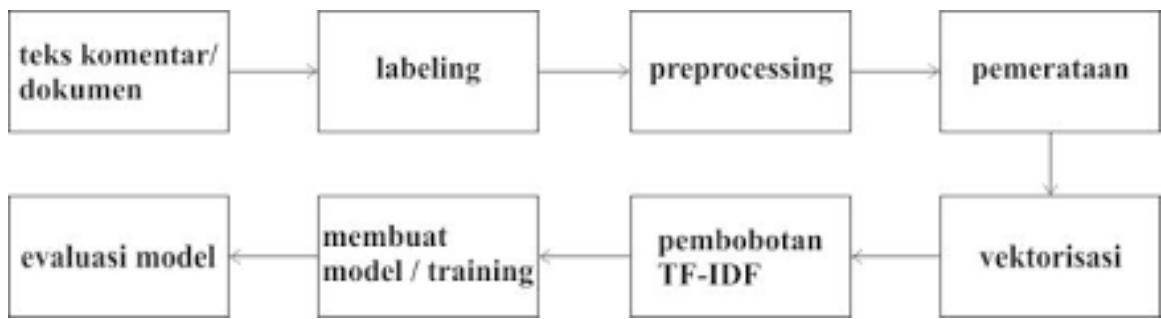


Figure 5.1: Ilustrasi Soal No.1

Untuk ilustrasinya dapat dilihat pada gambar

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

Dimensi dari vektor dataset google dapat mencapai 300 karena dimensi pada vektor agar dapat membandingkan bobot dari kata tersebut, misalkan terdapat kata Jaket dan Tas pada data set google setiap kata tersebut dibuat dimensi vektor senilai 300 kata Jaket dan 300 kata Tas, agar dapat membandingkan bobot dari kesamaan kata Jaket dan Tas.

Untuk ilustrasinya dapat dilihat pada gambar

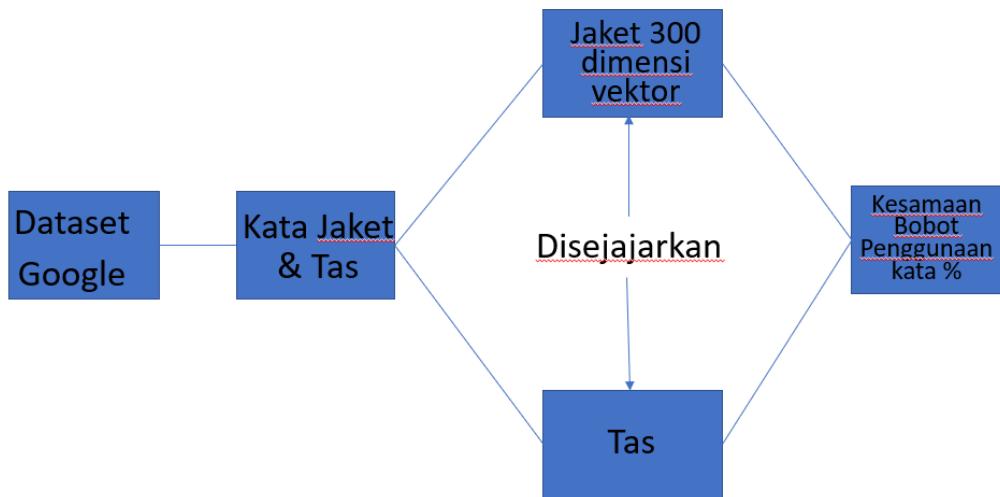


Figure 5.2: Ilustrasi Soal No. 2

3. Jelaskan Konsep vektorisasi untuk kata . dilengkapi dengan ilustrasi atau gambar.

Konsep dari vektorisasi kata yaitu agar dapat mengetahui kata tengah pada kalimat utama, Contoh (Subscribe channel ini dan Like yah Guys). Kata tengah dari contoh tersebut merupakan (dan) yang memiliki bobot sebagai kata tengah dari kalimat. Hal ini sangat berkaitan dengan dimensi vektor pada dataset google karena memiliki nilai atau bobot kata tengah.

Untuk ilustrasinya dapat dilihat pada gambar



Figure 5.3: Ilustrasi Soal No. 3

4. Jelaskan Konsep vektorisasi untuk dokumen. dilengkapi dengan ilustrasi atau gambar.

Konsep vektorisasi pada dokumen mesin akan membaca kata-kata terlebih dahulu pada semua kalimat yang ada di dalam dokumen dan nanti kalimat yg ada di dalam dokumen akan dipecah menjadi kata-kata.

Untuk ilustrasinya dapat dilihat pada gambar



Figure 5.4: Ilustrasi Soal No. 4

5. Jelaskan apa mean dan standar deviasi, lengkapi dengan ilustrasi atau gambar.

Mean merupakan nilai rata-rata yang tingkat akurasinya tinggi atau nilai tersebut sering muncul. Standar deviasi mengukur bagaimana nilai-nilai data tersebar. Bisa juga didefinisikan sebagai, rata-rata jarak penyimpangan titik-titik data diukur dari nilai rata-rata data tersebut.

Untuk ilustrasinya dapat dilihat pada gambar

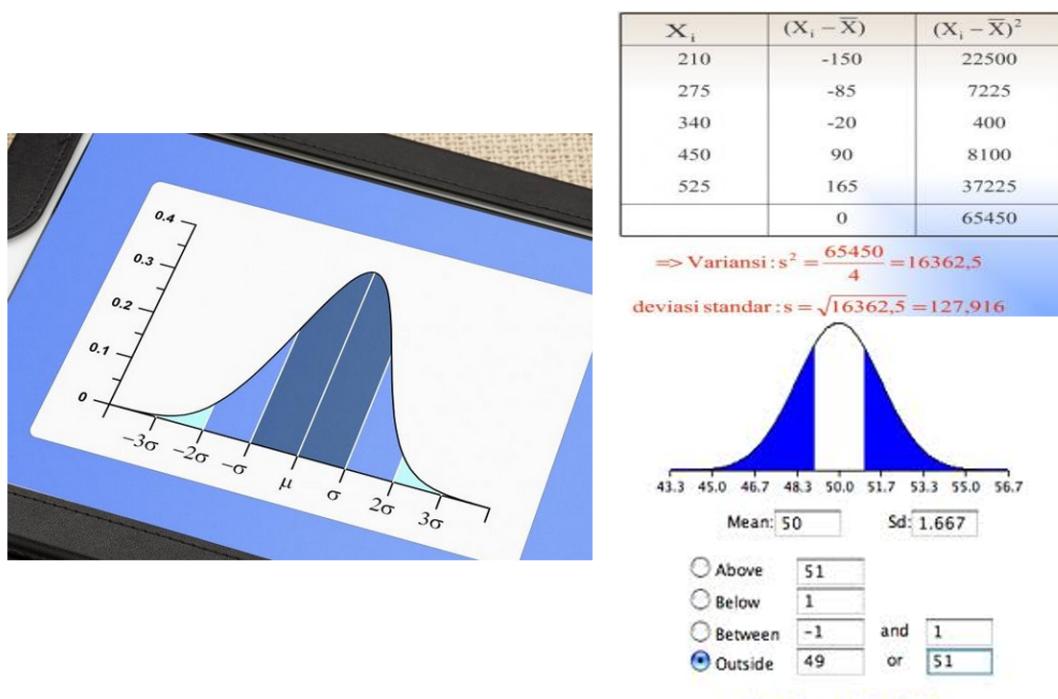


Figure 5.5: Ilustrasi Soal No. 5

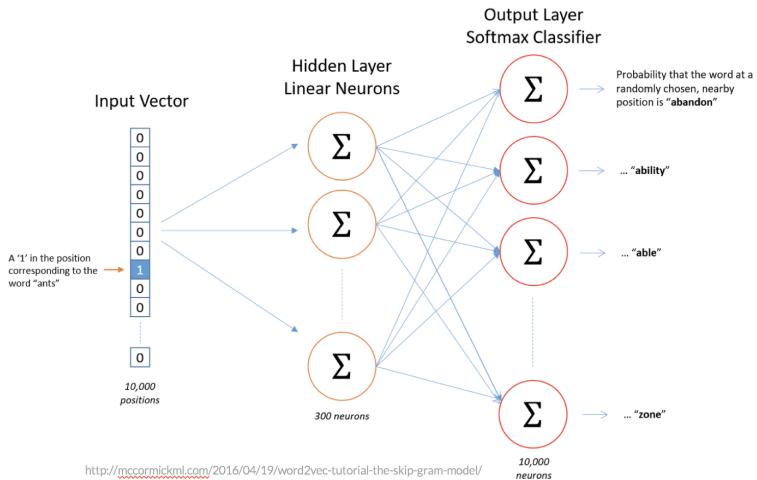
6. Jelaskan Apa itu Skip-Gram sertakan contoh ilustrasi.

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

Untuk ilustrasinya dapat dilihat pada gambar

5.5.2 Praktek Program

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



“Skip-Gram”
With one-hot encoded centre word, we can predict context words.

Hidden layer creates embeddings

Figure 5.6: Ilustrasi Soal No. 6

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.

```
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']
```

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

Figure 5.10: 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.

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

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

Figure 5.11: 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.

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

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

Figure 5.14: 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']
```

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

Figure 5.17: 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.

```
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.

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

Figure 5.18: Similariti Pada Kata Motor dan Cycle

```
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. Jelaskan dengan 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

5.6 Aip Suprapto Munari/1164063

5.6.1 Teori

1. Mengapa Kata-Kata Harus di Vektorisasi

Kata harus divektorisasi karena mesin hanya mampu membaca data dalam tipe/bentuk angka. Oleh sebab itu, diperlukannya vektorisasi kata untuk mampu membaca data tersebut.

- Gambar :

Penjelasan : Berdasarkan pengertian diatas, ada beberapa contoh yang bisa diterapkan. Untuk salah satu contoh dari klasifikasi data sendiri dapat diliat pada gambar berikut ??.

2. Mengapa Dimensi Dari Vektor Dataset Google Bisa Sampai 300

Setiap nilai dalam vektor 300 dimensi yang terkait dalam sebuah kata "dioptimalkan" dalam beberapa hal untuk menangkap aspek yang berbeda. Dengan

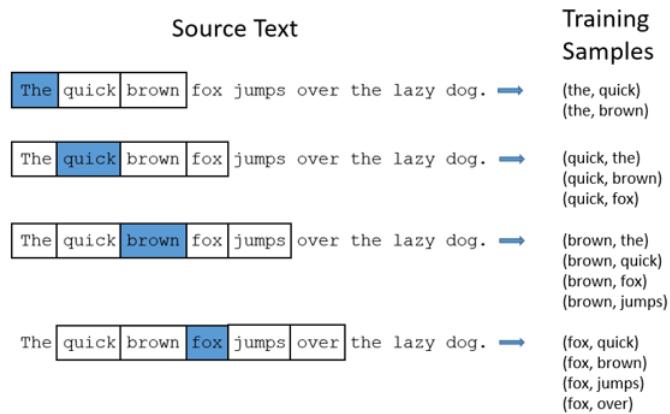


Figure 5.22: Vektorisasi Kata Aip

kata lain masing-masing dari 300 nilai sesuai dengan beberapa fitur abstrak kata. Menghapus kombinasi nilai-nilai ini secara acak akan menghasilkan vektor yang mungkin kurang informasi penting tentang kata tersebut dan mungkin tidak lagi berfungsi sebagai representasi yang baik dari kata itu. Atau singkat cerita mungkin ada lebih dari 3 miliar kata-kata dan kalimat atau data yang tidak mungkin disimpan dalam 1 dimensi vektor maka disimpan menjadi 300 dimensi vektor untuk mengurangi kegagalan memori.

- Gambar :

Penjelasan : Berdasarkan pengertian diatas, ada beberapa contoh yang bisa diterapkan. Untuk salah satu contoh dari klasifikasi data sendiri dapat diliat pada gambar berikut ??.



Figure 5.23: Google Dataset Aip

3. Konsep Vektorisasi Kata

Konsep vektorisasi data merupakan kata-kata yang di inputkan pada mesin learning. Dan outputnya berupa kata-kata atau keyword dari pencarian yang telah dilakukan sebelumnya. Contohnya pada saat kita melakukan pencarian

di youtube maupun pencarian google. Maka akan muncul hasil dari pencarian dari kata-kata yang telah kita cari atau input.

- Gambar :

Penjelasan : Berdasarkan pengertian diatas, ada beberapa contoh yang bisa diterapkan. Untuk salah satu contoh dari klasifikasi data sendiri dapat diliat pada gambar berikut ??.

Source Text	Training Samples
The quick brown fox jumps over the lazy dog.	(the, quick) (the, brown)
The quick brown fox jumps over the lazy dog.	(quick, the) (quick, brown)
The quick brown fox jumps over the lazy dog.	(brown, the) (brown, quick) (brown, fox) (brown, jumps)
The quick brown fox jumps over the lazy dog.	(fox, quick) (fox, brown) (fox, jumps) (fox, over)

Figure 5.24: Vektorisasi Kata Aip

4. Konsep Vektorisasi Dokumen

Konsep vektorisasi dokumen yaitu mesin akan membaca terlebih dahulu semua kalimat yang berada pada dalam dokumen dan nanti kalimat tersebut akan dipecah menjadi kata-kata.

- Gambar :

Penjelasan : Berdasarkan pengertian diatas, ada beberapa contoh yang bisa diterapkan. Untuk salah satu contoh dari klasifikasi data sendiri dapat diliat pada gambar berikut ??.

Document	D1	D2	D3	D4	D5
Indonesian	1		2	1	3
Culture		1	2		
Sport		1	1	3	2
Archipelago	1	1			
diverse					

Figure 5.25: Vektorisasi Dokumen Aip

5. Mean dan Standar Deviasi

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

$$\begin{aligned} \text{Mean} &= \frac{\text{Jumlah seluruh nilai}}{\text{Banyaknya nilai}} \\ &= \frac{81}{5} \\ &= 17 \end{aligned}$$

Figure 5.26: Mean Aip

data seluruh individu dalam kelompok itu, kemudian dibagi dengan jumlah individu yang ada pada kelompok tersebut. Standar deviasi adalah nilai statistik yang digunakan untuk menentukan bagaimana sebaran data dalam sampel, dan seberapa dekat titik data individu ke mean atau rata-rata nilai sampel. Sebuah standar deviasi dari kumpulan data sama dengan nol menunjukkan bahwa semua nilai-nilai dalam himpunan tersebut adalah sama. Sebuah nilai deviasi yang lebih besar akan memberikan makna bahwa titik data individu jauh dari nilai rata-rata.

- Gambar :

Penjelasan : Berdasarkan pengertian diatas, ada beberapa contoh yang bisa diterapkan. Untuk salah satu contoh dari klasifikasi data sendiri dapat diliat pada gambar berikut ??.

$$\begin{aligned} \sigma &= \sqrt{\frac{\sum(X - \mu)^2}{N}}; \text{untuk populasi} \\ s &= \sqrt{\frac{\sum(X - \bar{X})^2}{n-1}}; \text{untuk sampel} \end{aligned}$$

Figure 5.27: Standar Deviasi Aip

6. Skip Gram

Skip-Gram mencoba memprediksi vektor kata-kata yang ada di konteks diberikan vektor kata tertentu. Skip-gram membuat sepasang kata target dan konteks sebagai sebuah instance.

- Gambar :

Penjelasan : Berdasarkan pengertian diatas, ada beberapa contoh yang bisa diterapkan. Untuk salah satu contoh dari klasifikasi data sendiri dapat diliat pada gambar berikut ??.

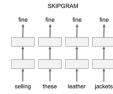


Figure 5.28: Skip-Gram Aip

5.7 PRAKTEK PROGRAM

5.8 Aip Suprapto Munari/1164063

5.8.1 Mencoba Dataset

5.8.1.1 Vektor

- Pada gambar diatas dapat dilihat bahwa vektor memiliki array sebanyak 300 dimensi. Untuk identitas sektor satu adalah 0.080

```
In [37]: gmodel['love']
Out[37]:
array([ 0.10302734, -0.15234375,  0.02587891,  0.16503906, -0.16503906,
       0.06689453,  0.29296875, -0.26367188, -0.140625 ,  0.20117188,
     -0.02624512, -0.08203125, -0.02770996, -0.04394531, -0.23535156,
     0.16992188,  0.12890625,  0.15722656,  0.00756836, -0.06982422,
     -0.03857422,  0.07958984,  0.22549219, -0.14355469,  0.16796875,
     -0.03515625,  0.05517578,  0.10693359,  0.11181641, -0.16308594,
     -0.11181641,  0.13964844,  0.01556396,  0.12792969,  0.15429688,
     0.07714844,  0.26171875,  0.08642578, -0.02514648,  0.33398438,
     0.18652344, -0.20996694,  0.07880078,  0.02600098, -0.10644531,
     -0.10253906,  0.12304688,  0.04711914,  0.02289473,  0.05834961]
```

Figure 5.29: Vektor Love Aip

- Pada gambar diatas untuk vektor faith dapat dilihat memiliki nilai 0.049 , untuk similaritasnya cukup mendekati vektor love dimana faith dapat dikategorikan dalam satu kategori dengan love.

```
In [38]: gmodel['faith']
Out[38]:
array([ 0.26367188, -0.04150391,  0.1953125 ,  0.13476562, -0.14648438,
       0.11962891,  0.04345703,  0.10351562,  0.12287031,  0.13476562,
       0.06646062,  0.18945312, -0.16601562,  0.21679688, -0.27148438,
       0.3203125 ,  0.10444921,  0.36132812, -0.1953125 , -0.18164062,
     0.15332031, -0.10839844,  0.10253906, -0.01367188,  0.23144531,
     -0.05957031, -0.22949219, -0.00604248,  0.26171875,  0.10302734,
     0.1328125 ,  0.21484375,  0.01135254,  0.02111816,  0.18554688,
     0.04125977,  0.12011719,  0.17480469, -0.22167969, -0.13476562,
     0.3125 ,  0.06640625, -0.17675781, -0.01708894, -0.1640625 ,
     -0.02819824,  0.01257324, -0.09521484, -0.18066406, -0.140625 ])
```

Figure 5.30: Vektor Faith Aip

- Vektor fall hanya memiliki nilai yaitu 0.046, dimana mesin memahami bahwa fall terdapat dalam satu kategori yang sama dengan faith.
- Vektor sick memiliki nilai identitas 0.085 dimana mendekati love.
- Vektor clear memiliki nilai identitas -0,025 dan tidak mendekati nilai dari vektor manapun sehingga tidak dapat dijadikan dalam satu kategori.

```
In [39]: gmodel['fall']
Out[39]:
array([-0.04272461,  0.10742188, -0.09277344,  0.16894531, -0.1328125,
-0.10693359,  0.04321289,  0.01904297,  0.14648438,  0.15039062,
-0.08691406,  0.04492188,  0.0145874,  0.08691406, -0.19824219,
-0.11035156,  0.01092529, -0.08300781, -0.0189209, -0.1953125,
-0.1015625,  0.13671875,  0.09228516, -0.12109375,  0.12695312,
0.03417969,  0.2109375,  0.01977539,  0.125,  0.01544189,
-0.26953125, -0.0089877, -0.07763672, -0.15527344, -0.03393555,
0.04199219, -0.29682812, -0.18554688,  0.08496094, -0.02087402,
```

Figure 5.31: Vektor Fall Aip

```
In [40]: gmodel['sick']
Out[40]:
array([ 1.82617188e-01,  1.49414062e-01, -4.05273438e-02,  1.64062500e-01,
-2.59765625e-01,  3.22256525e-01,  1.73828125e-01, -1.47460938e-01,
1.01074219e-01,  5.46875000e-02,  1.66992188e-01, -1.68945312e-01,
2.24304199e-03,  9.66796875e-02, -1.66015625e-01, -1.12304688e-01,
1.66015625e-01,  1.79687500e-01,  5.92841016e-03,  2.45117188e-01,
8.74023438e-02, -2.56347656e-02,  3.41796875e-01,  4.98046875e-02,
1.78710938e-01, -9.91821289e-04,  8.88671875e-02, -1.95312500e-01,
1.81640625e-01, -2.65625000e-01, -1.45507812e-01,  1.00585938e-01,
9.42382812e-02, -3.17500000e-02,  1.98374600e-02, -6.30648438e-02,
```

Figure 5.32: Vektor Sick Aip

```
In [41]: gmodel['clear']
Out[41]:
array([-2.44148625e-04, -1.02050781e-01, -1.49414062e-01, -4.24884688e-02,
-1.67968750e-01, -1.464434375e-01,  1.76757812e-01,  1.46484750e-01,
2.25562500e-01, -1.75390000e-01, -0.87280000e-01,  0.85390000e-01,
```

Figure 5.33: Vektor Clear Aip

- Untuk vektor shine 0.35 mendekati vektor Faith dan fall.

```
In [42]: gmodel['shine']
Out[42]:
array([-0.12402344,  0.25976562, -0.15917969, -0.27734375,  0.30273438,
-0.13907734,  0.15871875, -0.25390000, -0.87280000,  0.85390000,
```

Figure 5.34: Vektor Shine Aip

- Vektor bag memiliki i=nilai identitas 0.026 yang mendekati dengan vektor fall dan faith. Sehingga mesin memahami bahwa mungkin saja kedua vektor tersebut berada dalam satu kategori.

```
In [43]: gmodel['bag']
Out[43]:
array([ 0.03515625,  0.15334375, -0.12402344,  0.13379986, -0.11320125,
-0.01335667, -0.16113281,  0.14548438, -0.06835938,  0.146825,
```

Figure 5.35: Vektor Bag Aip

- Vektor car nilainya 0.48 tidak ada yang mendekati vektor manapun sehingga tidak dapat dikategorikan dalam satu kategori.

```
In [44]: gmodel['car']
Out[44]:
array([ 0.13085938,  0.08842285,  0.03344737, -0.05833789,  0.04003986,
-0.14257812,  0.04931641, -0.16894531,  0.20898458,  0.11962891,
```

Figure 5.36: Vektor Car Aip

- Vektor wash memiliki nilai 0.102 jauh dari vektor vektor lainnya.

```
In [45]: gmodel['wash']
Out[45]:
array([-9.40684932e-03, -2.38281256e-01, 1.41691562e-01, -5.46875000e-02, 1.34765625e-01,
       2.34213795e-01, -8.4472652e-02, -1.29892812e-01,
      1.87931936e-01, 2.35986250e-01, 1.1352331e-02, -1.66992188e-01,
      2.23212500e-01, 2.10948625e-01, 1.1352331e-02, -1.66992188e-01,
     -7.42187500e-02, 3.04687500e-01, 2.11944862e-01, -8.88671875e-02,
      2.6575125e-01, 2.12890625e-01, 1.74548947e-02, 2.02941895e-03,
      6.39804688e-02, 1.74548947e-02, 1.74548947e-02, 1.74548947e-02])
```

Figure 5.37: Vektor Wash Aip

- Vektor cycle memiliki nilai identitas 0.179 yang bisa mendekati vektor wash. Dapat dikatakan bahwa motor dapat dicuci jika diarti dalam satu kategori yang sama.

```
In [47]: gmodel['cycle']
Out[47]:
array([ 0.04541016,  0.21579688, -0.82799961,  0.12353516, -0.38703125,
       0.18261719, -0.15808312, -0.46179758,  0.21972656, -0.15808312,
      0.02563477, -0.07568359, -0.8625,  0.04614258, -0.18954688,
     -0.1330375,  0.04614258, 0.04614258, 0.04614258, 0.04614258,
      0.07226562, -0.06445312, 0.05517578, 0.14891486, 0.13971875,
      0.10392734, 0.02172852, -0.18093379, 0.02490234, -0.1864531,
     -0.05541991, -0.29492186, -0.48039861, 0.05347655, -0.88447286,
```

Figure 5.38: Vektor cycle Aip

5.8.1.2 Similariti

1. Lihat gambar berikut yang merupakan hasil prediksi similariti

```
In [52]: gmodel.similarity('motor', 'love')
Out[52]: 0.08877447064184623
In [53]: gmodel.similarity('motor', 'faith')
Out[53]: 0.049242012249214546
In [54]: gmodel.similarity('motor', 'fall')
Out[54]: 0.046621998318158265
In [55]: gmodel.similarity('motor', 'sick')
Out[55]: 0.08594041275958331
In [56]: gmodel.similarity('motor', 'clean')
Out[56]: -0.025459404931213092
In [57]: gmodel.similarity('motor', 'shine')
Out[57]: 0.035080895156368616
In [58]: gmodel.similarity('motor', 'bag')
Out[58]: 0.02609472187403132
In [59]: gmodel.similarity('motor', 'car')
Out[59]: 0.17900000000000002
```

Figure 5.39: Similariti Aip

Dapat disimpulkan bahwa

- Untuk Motor dan faith hasilnya adalah 0.049
- Untuk Motor dan fall hasilnya adalah 0.046
- Untuk Motor dan clear hasilnya adalah -0.025
- Untuk Motor dan wash hasilnya adalah 0.102
- Untuk Motor dan cycle hasilnya adalah 0.179
- Artinya Motor dan cycle memang dalam kategori ang sama misalnya dalam kategori kendaraan. Mesin sudah mengetahui bahwa keduanya dapat dikategorikan sebagai kendaraan.

5.8.2 Extract Words dan PermuteSentences

5.8.2.1 Extract Words

ExtractWords merupakan function untuk menambahkan, menghilangkan atau menghapuskan, hal hal yang tidak penting atau tidak perlu di dalam teks. Dalam contoh dibawah ini. menggunakan function extract words untuk menghapus komen dengan python style , mencari data yang diinginkan, dan memberikan spasi pada teks.

```
In [50]: import re
...: def extract_words(sku):
...:     sku = sku.lower()
...:     sku = re.sub(r'<.*?>', '', sku) #menghapus tag html
...:     sku = re.sub(r'\n+', '\n', sku) #menghapus garis tanda satu
...:     sku = re.sub(r'\u202c', ' ', sku) #menghapus tanda buka
...:     sku = re.sub(r'\u202d', ' ', sku) #menghapus tanda tutup
...:     return sku.split()
```

Figure 5.40: Extract Words Aip

5.8.2.2 PermuteSentences

PermuteSentences merupakan class yang digunakan untuk melakukan pengocokan secara acak pada data yang ada. Digunakan cara ini agar tidak terjadi kelebihan memori pada saat dijalankan. Contoh dibawah yaitu fungsi akan memanggil lenght. Yang kemudian mendefinisikan variabel req untuk lenght dan melakukan random choice yaitu pengocokan acak untuk kata motor.

```
import random
class PermuteSentences(object):
    def __init__(self, sku):
        self._sku = sku
    def __iter__(self):
        req = list(self._sku)
        random.choice('motor')
```

Figure 5.41: PermuteSentences Aip

5.8.2.3 Library Gensim TaggedDocument Dan Doc2Vec

Import library gensim dan meng-import modul Tagged Document dan Doc2Vec. Tagged Document adalah dokumen yang 'memisahkan informasi dan struktur dari presentasi' dengan menggunakan tag. Fungsi Doc2Vec berisi alpha dan min alpha parameter,berarti bahwa tingkat pembelajaran meluruh selama satu periode dari alpha untuk min alpha.

```
In [31]: from gensim.models.doc2vec import TaggedDocument
...: from gensim.models import Doc2Vec
```

Figure 5.42: Library Gensim TaggedDocument Dan Doc2Vec Aip

```

...: 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(dirname+"/"+fname,encoding="UTF-8") as f:
...:                 sent = f.read()
...:                 words = extract_words(sent)
...:                 unsup_sentences.append(TaggedDocument(words,[dirname+"/"+fname]))

```

Figure 5.43: Data 1 - Aip

```

In [15]: for dirname in ["review_polarity/txt_sentoken/pos","review_polarity/txt_sentoken/neg"]:
...:     for fname in sorted(os.listdir(dirname)):
...:         if fname[-4:] == '.txt':
...:             with open(dirname+"/"+fname,encoding="UTF-8") as f:
...:                 for i, skey in enumerate(f):
...:                     words = extract_words(skey)
...:                     unsup_sentences.append(TaggedDocument(words,[ "%s-%s-%d" % (dirname,fname,i)]))

```

Figure 5.44: Data 2 - Aip

```

In [18]: with open("stanfordSentimentTreebank/original_rt_snippets.txt",encoding="UTF-8") as f:
...:     for i, sent in enumerate(f):
...:         words = extract_words(sent)
...:         unsup_sentences.append(TaggedDocument(words,[ "rt-%d" % i]))

```

Figure 5.45: Data 2 - Aip

5.8.2.4 Menambahkan Data Training

Membaca direktori name dari data yang ada di dalam kurung, terdapat ada 3 data.

5.8.2.5 Pengocokan Dan Pembersihan Data

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

```

In [59]: import re
...: def extract_words(sent):
...:     sent = sent.lower()
...:     sent = re.sub('<[^>]+>', ' ', sent) #menghapus tag html
...:     sent = re.sub('([.,!?!,:;])', ' ', sent) #menghapus petik satu
...:     sent = re.sub('(\w)', ' ', sent) #menghapus petik dua
...:     sent = re.sub('(\n)', ' ', sent) #menghapus tanda baca
...:     sent = re.sub('(\s)', ' ', sent) #menghapus spasi yang berurutan
...:     return sent.split()
...:

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

```

Figure 5.46: Pengocokan dan Pembersihan Data - Aip

```

In [74]: len(unsup_sentences)
Out[74]: 3182
In [75]: unsup_sentences[0][1]
Out[75]: TaggedDocument(words=['after', 'watching', 'a_night_at_the_cobbury',
'you', 'll', 'be', 'left', 'with', 'exactly', 'the', 'same'], tags=[['rt-0']])
In [76]: muto=PermuteSentences(unsup_sentences)
In [77]: model = Doc2Vec(muto, dm=0, hs=1, vector_size=52)

```

Figure 5.47: Pengocokan dan Pembersihan Data - Aip

5.8.2.6 Mengapa Model Harus Di Save Dan Temporari Training Harus Dihapus

Untuk mencegah ram agar lambat. Sedangkan kenapa temporari training harus dihapus mengosongkan memori agar sedikit lega atau tidak lemot.

```
In [56]: model.delete_temporary_training_data(keep_inference=True)
```

Figure 5.48: Model Disave dan Temporari Train Hapus - Aip

```
In [56]: model.save("haci.d2v")
```

Figure 5.49: Model Disave dan Temporari Train Hapus - Aip

5.8.2.7 Infercode

Untuk menyimpulkan vektor yang berhubungan dengan vektor dokumen baru dan output tersebut merupakan sebuah array.

```
In [59]: model.infer_vector(extract_words("cape uyy"))
Out[59]:
array([-0.0013956,  0.0023099,  0.0025457,  0.0021709,  0.0015054,  0.0014287,
       0.0005958,  0.001727 , -0.00041525,  0.0054407, -0.00227619,
      -0.0007309,  0.0011829,  0.0011255,  0.0011255,  0.0011255,  0.0011255,
      -0.00063066,  0.00073265, -0.00062555,  0.00101114, -0.0022863,
      -0.0055729,  0.00168609, -0.00118292, -0.00352464,  0.00602394,
      -0.00084245,  0.00084245,  0.00084245,  0.00084245,  0.00084245,
      -0.0057237,  0.00084245, -0.00063093, -0.00362349, -0.0005215 ,
      -0.0076601, -0.00738083,  0.00465577, -0.0009361, -0.0062011,
      -0.00093609,  0.00093609,  0.00093609,  0.00093609,  0.00093609,
      -0.00092625, -0.0057103], dtype=float32)
```

Figure 5.50: Infercode - Aip

5.8.2.8 Cosinesimilarity

Cosine Similarity dapat diimplementasikan untuk menghitung nilai kemiripan antar kalimat dan menjadi salah satu teknik untuk mengukur kemiripan teks yang popular.

```
In [60]: from sklearn.metrics.pairwise import cosine_similarity
...: cosine_similarity(
...:     [model.infer_vector(extract_words("leur euy"))],
...:     [model.infer_vector(extract_words("aowwkuuwkwukkk"))])
Out[60]: array([[0.2488943]], dtype=float32)
```

Figure 5.51: Cosinesimilarity - Aip

```
In [61]: from sklearn.metrics.pairwise import cosine_similarity
...: cosine_similarity(
...:     [model.infer_vector(extract_words("capedeh"))],
...:     [model.infer_vector(extract_words("capedeh"))])
Out[61]: array([[0.9999999]], dtype=float32)
```

Figure 5.52: Cosinesimilarity - Aip

```
In [21]: sentvecs = []
...: sentences = []
...: sentiments = []
...: for file in ["yelp","amazon_cells","imdb"]:
...:     with open("dataset/{}-labelled.txt".format(file), encoding='UTF-8') as f:
...:         for i, line in enumerate(f):
...:             line_split = line.rstrip().split("\t")
...:             sentence = line_split[0]
...:             words = extract_words(line_split[1])
...:             sentvecs.append(model.infer_vector(words, steps=10)) # buat vektor dari dokumen (ini
...:             sentiments.append(int(line_split[1]))
```

Figure 5.53: Cosinesimilarity - Aip

```
In [22]: combined = list(zip(sentences, sentvecs, sentiments))
...: random.shuffle(combined)
...: sentences, sentvecs, sentiments = zip(*combined)
```

Figure 5.54: Cosinesimilarity - Aip

5.8.2.9 Praktek Score Dari Cross Validation

Hasil dari praktek tersebut menunjukan untuk menghitung memprediksi dan mengetahui keakuratan dari suatu nilai.

```
In [66]: from sklearn.neighbors import KNeighborsClassifier
...: from sklearn.ensemble import RandomForestClassifier
...: from sklearn.model_selection import cross_val_score
...: import numpy as np
...: ...
...: clf = KNeighborsClassifier(n_neighbors=9)
...: clfrf = RandomForestClassifier()
```

Figure 5.55: Score Cross Validation - Aip

```
In [67]: scores = cross_val_score(clf, sentvecs, sentiments, cv=5)
...: np.mean(scores), np.std(scores)
Out[67]: (0.4933333333333333, 0.03880161029656395)
```

Figure 5.56: Score Cross Validation - Aip

```
In [68]: scores = cross_val_score(clfrf, sentvecs, sentiments, cv=5)
C:\ProgramData\Anaconda3\lib\site-packages\sklearn\ensemble\forest.py:246:
FutureWarning: The default value of n_estimators will change from 10 in version 0.20
to 100 in 0.22.
"10 in version 0.20 to 100 in 0.22.", FutureWarning)
C:\ProgramData\Anaconda3\lib\site-packages\sklearn\ensemble\forest.py:246:
FutureWarning: The default value of n_estimators will change from 10 in version 0.20
to 100 in 0.22.
"10 in version 0.20 to 100 in 0.22.", FutureWarning)
C:\ProgramData\Anaconda3\lib\site-packages\sklearn\ensemble\forest.py:246:
FutureWarning: The default value of n_estimators will change from 10 in version 0.20
to 100 in 0.22.
"10 in version 0.20 to 100 in 0.22.", FutureWarning)
C:\ProgramData\Anaconda3\lib\site-packages\sklearn\ensemble\forest.py:246:
FutureWarning: The default value of n_estimators will change from 10 in version 0.20
to 100 in 0.22.
"10 in version 0.20 to 100 in 0.22.", FutureWarning)
Out[68]: (0.5186666666666667, 0.02128118626608165)
```

Figure 5.57: Score Cross Validation - Aip

5.9 Penanganan Error

AIP SUPRAPTO MUNARI 1164063

5.9.1 Error

- Berikut ini merupakan eror yang didapatkan saat menjalankan program diatas

```

In [69]: 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)
C:\ProgramData\Anaconda3\lib\site-packages\sklearn\ensemble\forest.py:246:
FutureWarning: The default value of n_estimators will change from 10 in version 0.20
to 100 in 0.22.
"10 in version 0.20 to 100 in 0.22.", FutureWarning)
C:\ProgramData\Anaconda3\lib\site-packages\sklearn\ensemble\forest.py:246:
FutureWarning: The default value of n_estimators will change from 10 in version 0.20
to 100 in 0.22.
"10 in version 0.20 to 100 in 0.22.", FutureWarning)
C:\ProgramData\Anaconda3\lib\site-packages\sklearn\ensemble\forest.py:246:
FutureWarning: The default value of n_estimators will change from 10 in version 0.20
to 100 in 0.22.
"10 in version 0.20 to 100 in 0.22.", FutureWarning)
C:\ProgramData\Anaconda3\lib\site-packages\sklearn\ensemble\forest.py:246:
FutureWarning: The default value of n_estimators will change from 10 in version 0.20
to 100 in 0.22.
"10 in version 0.20 to 100 in 0.22.", FutureWarning)
C:\ProgramData\Anaconda3\lib\site-packages\sklearn\ensemble\forest.py:246:
FutureWarning: The default value of n_estimators will change from 10 in version 0.20
to 100 in 0.22.
"10 in version 0.20 to 100 in 0.22.", FutureWarning)
Out[69]: (0.787, 0.053592281409377231)

```

Figure 5.58: Score Cross Validation - Aip

```
| KeyError: 'generic-food'
```

Figure 5.59: Error Key Aip

- Pastikan dataset berada dalam satu folder.

Chapter 6

Discussion

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

6.1 Aip Suprapto Munari/1164063

6.1.1 Teori

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

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

- Ilustrasi Gambar:

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

Neural Network merupakan replika dari sistem syaraf yang terdapat pada sistem otak manusia. Dalam proses kerjanya, otak manusia disusun atas miliaran neuron dimana setiap neuron akan terhubung pada puluhan ribu neuron lain.

- Ilustrasi Gambar:

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

Bobot merupakan suatu nilai yang mendefinisikan tingkat atau kepentingan hubungan antara suatu node dengan node yang lain. Semakin besar bobot

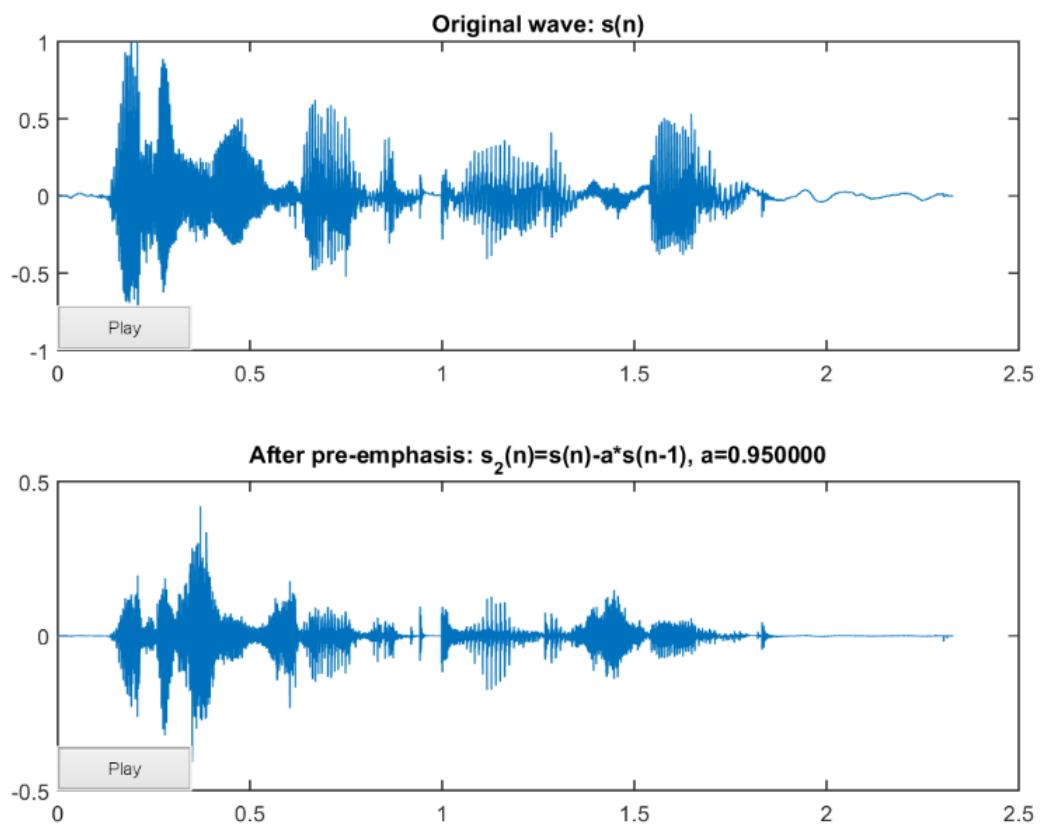


Figure 6.1: MFCC - Aip

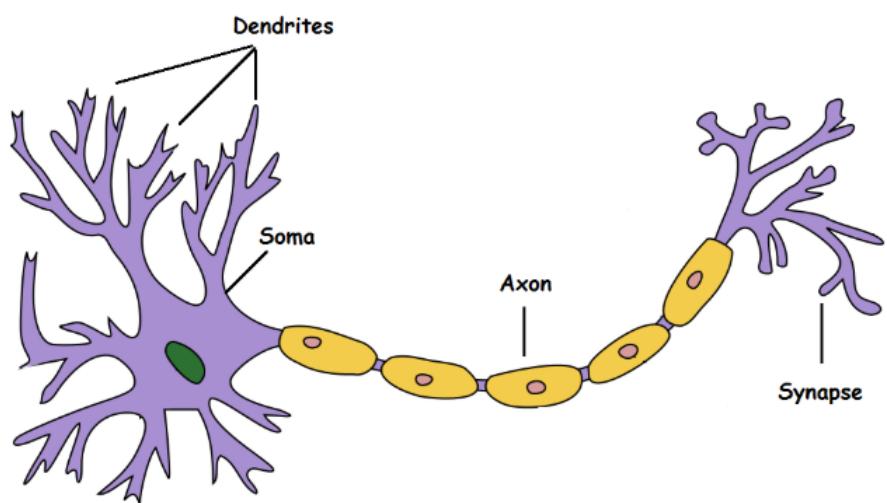


Figure 6.2: Konsep Dasar Neural Network - Aip

suatu hubungan menandakan semakin pentingnya hubungan kedua node tersebut. Bobot merupakan suatu hubungan berupa bilangan real maupun integer, tergantung dari jenis permasalahan dan model yang digunakan. Bobot-bobot tersebut bisa ditentukan untuk berada didalam interval tertentu. selama proses pelatihan, bobot tersebut dapat menyesuaikan dengan pola-pola input.

- Ilustrasi Gambar :

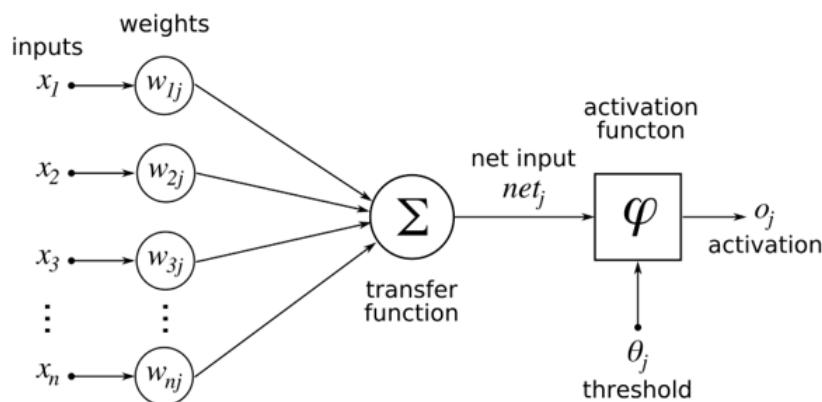


Figure 6.3: Konsep Pembobotan Neural Network - Aip

4. Konsep fungsi aktifasi dalam Neural Network, dilengkapi dengan ilustrasi atau gambar. Operasi matematik yang dikenakan pada sinyal output y. Sehingga fungsi ini akan digunakan untuk pengaktifan dan juga penonaktifan neuron.

- Dalam konsep fungsi aktivasi Neuron Network terdapat beberapa jenis:
 - Fungsi Undak Biner Hard Limit (Menkonversi nilai masukan dari suatu variabel)
 - Fungsi Undak Biner Threshold (Menggunakan nilai ambang 0 sebagai batas eksekusil)
 - Fungsi Bipolar Symetric Hard Limit (Mempunyai keluaran bernilai 1 dan 0)
 - Fungsi Bipolar Threshold (Mempunyai keluaran bernilai 1, 0 atau -1)
 - Ilustrasi Gambar:

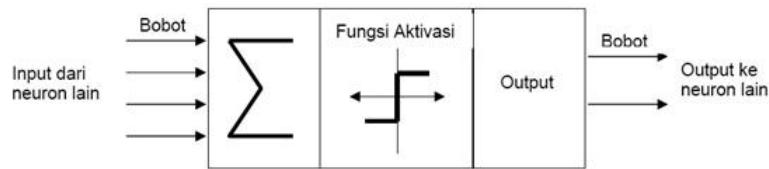


Figure 6.4: Konsep Fungsi Aktifasi - Aip

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

Penjelasan Cara Membaca Hasil Plot Dari MFCC :

- Ilustrasi Gambar :

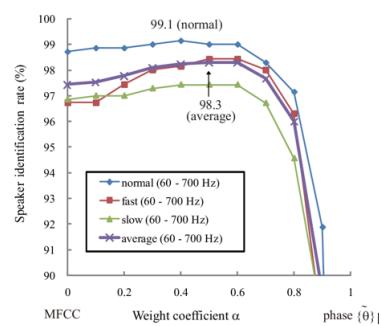


Figure 6.5: Plot MFCC - Aip

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

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

- Ilustrasi Gambar:

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 6.6: One-Hot Encoding - Aip

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

(a) np.unique:

Berfungsi untuk menemukan elemen unik array. Ada tiga output opsional selain elemen unik:

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

(b) Ilustrasi Gambar :

```
1 | >>> import numpy as np
2 | >>> x = np.array([[1, 1], [2,3], [3,4]])
3 | >>> np.unique(x)
4 | array([1, 2, 3, 4])
```

Figure 6.7: np.unique - Aip

(c) to.categorical:

Berfungsi untuk mengubah vektor kelas yang berupa integer menjadi matriks kelas biner.

- Ilustrasi Gambar :

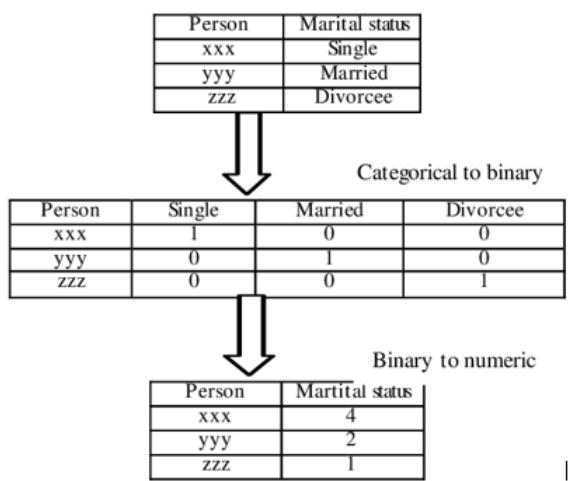


Figure 6.8: to.categorical - Aip

8. Fungsi dari Sequential, dilengkapi dengan ilustrasi atau gambar. Sebuah jenis model yang digunakan dalam perhitungan ataupun code program yang direalisasikan.

- Ilustrasi Gambar:

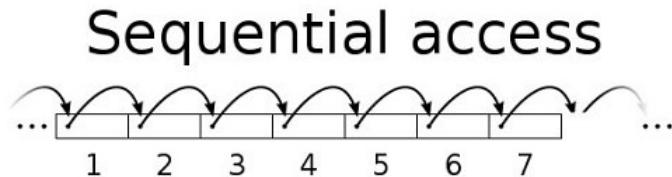


Figure 6.9: Sequential - Aip

6.1.2 Praktek

1. Menjelaskan Isi Dari Data GTZAN Genre Colection dan Data Freesound

Isi dari data GTZAN adalah data musik berdasarkan genre atau jenis dari lagu. Yang sudah di folderkan berdasarkan jenis lagu nya masing-masing.

```
1 # -*- coding: utf-8 -*-
2 """
3 Created on Wed Apr  4 01:03:26 2019
4
5 @author: kiting13
6 """
7
8 filename_metal = 'genres/metal/metal.00000.au'
9 x_metal, sr_metal = librosa.load(filename_metal, duration=10)
```

Hasil 6.10 :

```
In [12]: filename_metal = 'genres/metal/metal.00000.au'
...: x_metal, sr_metal = librosa.load(filename_metal, duration=10)
```

Figure 6.10: Hasil No 1 Aip

Baris 1: filename metal merupakan variabel yang berisikan direktori dari file yang dituju, disini digunakan file audio dari genre metal

Baris 2: x metal dan sr metal variabel yang digunakan untuk meload file dari variabel filename metal menggunakan librari Librosa. Yang nantinya akan digunakan pada MFCC

2. Menjelaskan Perbaris Kode Fungsi Dari display mfcc()

- Kode Program

```
1 # -*- coding: utf-8 -*-
2 """
3 Created on Wed Apr  4 01:04:16 2019
4
5 @author: kiting13
6 """
7
8 def display_mfcc(song):
9     y, _ = librosa.load(song)
10    mfcc = librosa.feature.mfcc(y)
11
12    plt.figure(figsize=(10, 4))
13    librosa.display.specshow(mfcc, x_axis='time', y_axis='mel')
14    plt.colorbar()
15    plt.title(song)
16    plt.tight_layout()
17    plt.show()
18 display_mfcc('genres/hiphop/hiphop.00000.au')
```

Kode Program 26.11 :

```
In [13]: def display_mfcc(song):
...:     y, _ = librosa.load(song)
...:     mfcc = librosa.feature.mfcc(y)
...:
...:     plt.figure(figsize=(10, 4))
...:     librosa.display.specshow(mfcc, x_axis='time', y_axis='mel')
...:     plt.colorbar()
...:     plt.title(song)
...:     plt.tight_layout()
...:     plt.show()
```

Figure 6.11: Kode Program No 2 Aip

- Hasil Plot

Apabila Sudah Di Plot 6.12 :

Baris 1: Membuat fungsi display mfcc untuk menampilkan vektorisasi sebuah suara

Baris 2: Membuat variabel y untuk membaca variabel song dari perintah library load song

Baris 3: Membuat variabel mfcc untuk variabel y dan mengubah suara menjadi vektor

Baris 4: Memplot gambar dengan ukuran 10X4

Baris 5: Menampilkan spektrogram atau chromagram agar hasil dari kodinan ini berwarna atau pada grafiknya

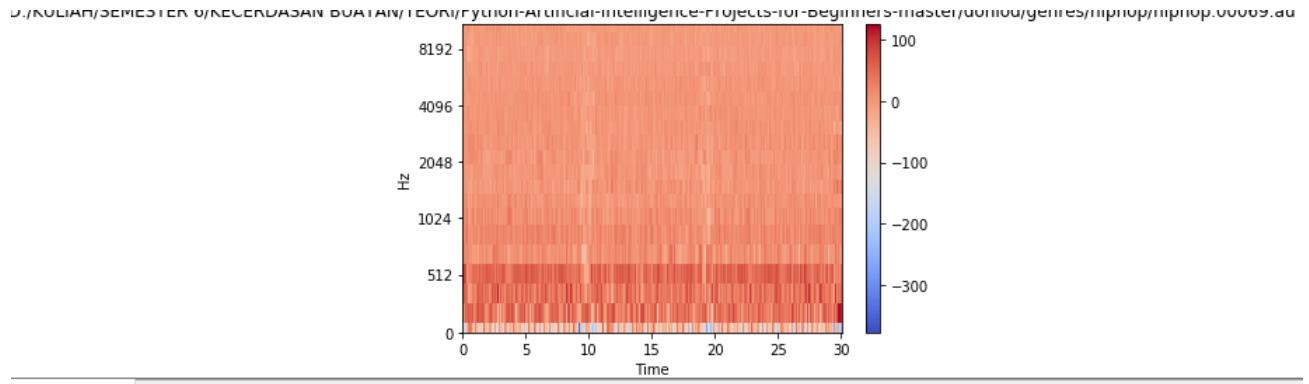


Figure 6.12: Hasil No 2 Aip

Baris 6: Menambahkan colorbar pada plot

Baris 7: Menetapkan judul suara atau lagu

Baris 8: Memberikan label pada sumbu di grafik

Baris 9: Menampilkan hasil plot

3. Menjelaskan Kode Program extract features song() dan Mengapa Yang Diambil 25000 Baris Pertama

- Penjelasan Kode program

```

1 # -*- coding: utf-8 -*-
2 """
3 Created on Wed Apr  4 01:20:25 2019
4
5 @author: kiting13
6 """
7
8 def extract_features_song(f):
9     y, _ = librosa.load(f)
10
11     # get Mel-frequency cepstral coefficients
12     mfcc = librosa.feature.mfcc(y)
13     # normalize values between -1,1 (divide by max)
14     mfcc /= np.amax(np.absolute(mfcc))
15
16     return np.ndarray.flatten(mfcc)[:25000]

```

Hasil 36.13 :

Baris 1: Membuat fungsi extract features song dengan inputan f

Baris 2: Membuat variabel y untuk meload inputan f dari perintah librosa load song

Baris 3: Membuat variabel mfcc untuk membuat featuredari variabel y

```

In [15]: def extract_features_song(f):
...:     y, _ = librosa.load(f)
...:
...:     # get Mel-frequency cepstral coefficients
...:     mfcc = librosa.feature.mfcc(y)
...:     # normalize values between -1,1 (divide by max)
...:     mfcc /= np.amax(np.absolute(mfcc))
...:
...:     return np.ndarray.flatten(mfcc)[:25000]

```

Figure 6.13: Hasil No 3 Aip

Baris 4: Membuat normalisasi nilai antara -1 sampai 1

Baris 5: Mengambil 25000 data pertama berdasarkan durasi suara lalu dikembalikan salinan arraynya dan dikecilkan menjadi satu.

- Mengapa Yang Diambil 25000 Baris Pertama

Diambil 25000 baris pertama karena agar eksekusi data atau saat running tidak memakan waktu yang cukup lama.

4. Menjelaskan Kode Program generate features dan labels

- Penjelasan Kode program

```

1  # -*- coding: utf-8 -*-
2  """
3  Created on Wed Apr  4 01:46:37 2019
4
5  @author: kiting13
6  """
7
8  def generate_features_and_labels():
9      all_features = []
10     all_labels = []
11
12     genres = [ 'blues' , 'classical' , 'country' , 'disco' , 'hiphop'
13               , 'jazz' , 'metal' , 'pop' , 'reggae' , 'rock' ]
14     for genre in genres:
15         sound_files = glob.glob('genres/' + genre + '/*.au')
16         print('Processing %d songs in %s genre...' % (len(
17             sound_files), genre))
18         for f in sound_files:
19             features = extract_features_song(f)
20             all_features.append(features)
21             all_labels.append(genre)
22
23     # convert labels to one-hot encoding cth blues : 1000000000
24     # classic 0100000000
25     label_uniq_ids, label_row_ids = np.unique(all_labels,
26                                              return_inverse=True) #ke integer
27     label_row_ids = label_row_ids.astype(np.int32, copy=False)

```

```

24     onehot_labels = to_categorical(label_row_ids, len(
25         label_uniq_ids))#ke one hot
        return np.stack(all_features), onehot_labels

```

Hasil 46.14 :

```

In [16]: 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 classic 0100000000
...:     label_uniq_ids, label_row_ids = np.unique(all_labels, return_inverse=True)#ke integer
...:     label_row_ids = label_row_ids.astype(np.int32, copy=False)
...:     onehot_labels = to_categorical(label_row_ids, len(label_uniq_ids))#ke one hot
...:     return np.stack(all_features), onehot_labels

```

Figure 6.14: Hasil No 4 Aip

Baris 1: Pembuatan perintah untuk fungsi generate geatures dan labels

Baris 2: Membuat variabel all features dengan array / parameter kosong

Baris 3: Membuat variabel label dengan array / parameter kosong

Baris 4: Mendefinisikan variable genres yang didalamnya berisi nama folder-folder pada variabel genres tersebut

Baris 5: Membuat perintah looping (header)

Baris 6: Membuat atribut sound files yang berisi perintah looping perfolder dari folder genres dan mengambil semua file berekstensi au (semuanya dieksekusi berdasarkan module glob).

Baris 7: Menampilkan jumlah song yang dieksekusi.

Baris 8: Membuat perintah fungsi dari sound files

Baris 9: Membuat variabel features untuk memanggil fungsi extract features song (f) sebagai inputan. Setiap satu file array sound files dilakukan ekstrak fitur.

Baris 10: Memasukkan semua features menggunakan perintah append kedalam all features

Baris 11: Memasukkan semua genres menggunakan perintah append ke dalam all labels

Baris 12: Mendefinisikan label uniq ids dan label row ids sebagai variabel dimana mengeksekusi perintah np.unique dengan parameter variabelnya all labels dan return inverse=True.

Baris 13: Membuat variabel label row ids untuk menentukan type dari variabel tersebut dengan type bit yang sesuai dengan yang digunakan.

Baris 14: Membuat variabel onehot labels dimana mengeksekusi to categorical dengan variabel parameter low row ids dan len(label uniq ids)

Baris 15: Mengembalikan dan menampilkan hasil eksekusi dari variabel parameter all features dan onehot labels perintah dari np.stack.

5. Menjelaskan Mengapa Fungsi generate features and labels sangat lama ketika meload dataset genre

- Kenapa saat load dataset genre lama? karena ada 1000 data yang di proses.

```
1 # -*- coding: utf-8 -*-
2 """
3 Created on Wed Apr  4 01:50:04 2019
4
5 @author: kiting13
6 """
7
8 features , labels = generate_features_and_labels()
```

Hasil 6.15 :

```
In [17]: features, labels = generate_features_and_labels()
Processing 0 songs in blues genre...
Processing 0 songs in classical genre...
Processing 0 songs in country genre...
Processing 0 songs in disco genre...
Processing 0 songs in hiphop genre...
Processing 0 songs in jazz genre...
Processing 0 songs in metal genre...
Processing 0 songs in pop genre...
```

Figure 6.15: Hasil No 5 Aip

- Penjelasan Hasil:

Baris 1: Variabel features and label akan mengeksekusi isi dari features and label

Baris 2: Memproses 100 lagu di genre blues

Baris 3: Memproses 100 lagu di genre classical

Baris 4: Memproses 100 lagu di genre country

Baris 5: Memproses 100 lagu di genre disco

Baris 6: Memproses 100 lagu di genre hip hop

Baris 7: Memproses 100 lagu di genre jazz

Baris 8: Memproses 100 lagu di genre metal

Baris 9: Memproses 100 lagu di genre pop

Baris 10: Memproses 100 lagu di genre reggae

Baris 11: Memproses 100 lagu di genre rock

6. Mengapa Harus Dilakukan Pemisahan Data Training Dan Data Set Sebesar 80%?

- Penjelasan :

Diperlukan Pemisahan sebesar 80% data dikarenakan untuk kemudahan dalam melakukan pengacakan yang dimana untuk komposisinya sendiri sebesar 80% untuk data training dan 20% untuk data test (dari dataset). Data trainingnya memang diharuskan lebih banyak sehingga pada saat pengacakan yang dilakukan datanya akan berada pada urutan yang berbeda.

- Code Yang Digunakan :

```
1 # -*- coding: utf-8 -*-
2 """
3 Created on Thu Apr  4 02:04:15 2019
4
5 @author: kiting13
6 """
7
8 # In [3]: fitur ekstraksi
9 training_split = 0.8
10 # In [3]: fitur ekstraksi
11 # last column has genre, turn it into unique ids
12 alldata = np.column_stack((features, labels))
13 # In [3]: fitur ekstraksi
14 np.random.shuffle(alldata)
15 splitidx = int(len(alldata) * training_split)
16 train, test = alldata[:splitidx ,:], alldata[splitidx :,:]
17 # In [3]: fitur ekstraksi
18 print(np.shape(train))
19 print(np.shape(test))
20 # In [3]: fitur ekstraksi
21 train_input = train[:, :-10]
22 train_labels = train[:, -10:]
23 # In [3]: fitur ekstraksi
24 test_input = test[:, :-10]
25 test_labels = test[:, -10:]
26 # In [3]: fitur ekstraksi
27 print(np.shape(train_input))
28 print(np.shape(train_labels))
```

- Penjelasan Perbaris (Tapi dalam soal tidak diperintahkan namun hanya saya jelaskan) :
 - (a) Baris Code 1 : Melakukan Proses Training split dimana akan memisahkan training set sebanyak 80%
 - (b) Baris Code 2 : Melakukan penumpukan features dan labels yang didefinisikan dalam variabel all_data
 - (c) Baris Code 3 : Melakukan Pengocokan (shuffle) untuk variabel all_data
 - (d) Baris Code 4 : Membuat variabel splitidx untuk mengkalikan isi dari variabel all_data dengan training_split yang ada.
 - (e) Baris Code 5 : Merealisasikan variabel train dan test dengan all_data yang telah diproses dengan variabel splitidx
 - (f) Baris Code 6 - 7 : Memisahkan mana yang termasuk data train dan mana yang termasuk data test dengan perintah / pada np.shape kemudian di cetak
 - (g) Baris Code 8 - 9 : Menampilkan isi train dari 2 variabel yaitu train_input dan train_labels
 - (h) Baris Code 10-11 : Menampilkan isi test dari 2 variabel yaitu test_input dan test_labels
 - (i) Baris Code 12-13 : Mencetak / menampilkan data training dimana terdapat 800 baris dan data test sebesar 200 baris dengan jumlah kolom yang sama yaitu 25010

- Ilustrasi Gambar : 6.16 dan 6.17

- Penjelasan :

Berdasarkan gambar ataupun hasil dari codingan tersebut, memperlihatkan bahwa data dipisah dan dipecah berpatokan dengan ketentuan 80%. Untuk hasil pertama data trainingnya ada 800 baris dengan 25000 kolom dan data set sebanyak 200 baris dengan 10 kolom, sedangkan untuk hasil kedua yang telah digabungkan dengan one-hot encoding maka data training terdapat 800 baris dan data set dengan 200 baris namun keduanya memiliki jumlah kolom yang sama yaitu 25010.

7. Penjelasan Parameter Dari Fungsi Sequential().

- Code Yang Digunakan :

```

In [7]: training_split = 0.8
In [8]: all_data = np.column_stack((features, labels))
In [9]: np.random.shuffle(all_data)
In [10]: train, test = all_data[:int(len(all_data) * training_split)], all_data[int(len(all_data) * training_split):]
In [10]: print(np.shape(train))
In [10]: (380, 100)
In [10]: print(np.shape(test))
In [10]: (95, 100)

```

Figure 6.16: Pemisahan Data Training Dan Dataset - Aip

```

In [11]: train_input = train[:, :-1]
In [11]: train_labels = train[:, -1]
In [12]: test_input = test[:, :-1]
In [12]: test_labels = test[:, -1]
In [13]: print(np.shape(train_input))
In [13]: print(np.shape(train_labels))
In [13]: (380, 99)
In [13]: (380, 1)

```

Figure 6.17: Pemisahan Data Training Dan Dataset 2 - Aip

```

1 # -*- coding: utf-8 -*-
2 """
3 Created on Wed Apr  4 01:59:47 2019
4
5 @author: kiting13
6 """
7
8 model = Sequential([
9     Dense(100, input_dim=np.shape(train_input)[1]),
10    Activation('relu'),
11    Dense(10),
12    Activation('softmax'),
13])

```

Penjelasan :

- Ilustrasi Gambar : 6.18

Pada gambar tersebut dapat dilihat bahwa untuk layer pertama densenya dari 100 neuron kemudian untuk inputan activationnya menggunakan fungsi relu (nilai maksimum yang akan dipilih). Dense 10 mengkategorikan 10 neuron untuk jenis genrenya untuk keluarannya menggunakan aktivasi yaitu fungsi Softmax.

```

In [38]: model = Sequential([
...     Dense(100, input_dim=np.shape(train_input)[1]),
...     Activation('relu'),
...     Dense(10),
...     Activation('softmax'),
... ])

```

Figure 6.18: Fungsi Sequential - Aip

8. Penjelasan Masing-Masing Parameter Dari Fungsi Compile().

- Code Yang Digunakan :

```

1 # -*- coding: utf-8 -*-
2 """
3 Created on Wed Apr  4 01:59:34 2019
4
5 @author: kiting13
6 """

```

```

7
8 model.compile(optimizer='adam',
9                 loss='categorical_crossentropy',
10                metrics=['accuracy'])
11 print(model.summary())

```

- Ilustrasi Gambar : 6.19

Penjelasan :

Pada gambar dapat dilihat bahwa untuk model.compilenya dilakukan pemrosesan menggunakan algortima adam sebagai optimizer yang sudah didefinisikan. Kemudian adam tersebut merupakan algoritme pengoptimalan dan untuk memperbarui bobot jaringan yang berulang berdasarkan data training sebelumnya. Untuk loss sendiri menggunakan categorical_crossentropy yang difungsikan sebagai optimasi skor / accuracy. Dan model tersebut digabungkan serta disimpulkan kemudian dicetak.

```

In [8]: model.compile(optimizer='adam',
...                 loss='categorical_crossentropy',
...                 metrics=['accuracy'])
... print(model.summary())

```

Layer (Type)	Output Shape	Param #
dense_1 (Dense)	(None, 100)	250000
activation_1 (Activation)	(None, 100)	0
dense_2 (Dense)	(None, 10)	100
activation_2 (Activation)	(None, 10)	0

Figure 6.19: Fungsi Compile - Aip

9. Penjelasan Masing-Masing Parameter Dari Fungsi Fit().

- Code Yang Digunakan :

```

1 # -*- coding: utf-8 -*-
2 """
3 Created on Wed Apr  4 02:01:15 2019
4
5 @author: kiting13
6 """
7
8 model.fit(train_input, train_labels, epochs=10, batch_size=32,
9           validation_split=0.2)

```

- Penjelasan :

Berdasarkan gambar tersebut dapat dilihat bahwa pada model fit dilakukan pelatihan dengan epoch(iterasi berapa kali nilai digunakan) dengan rambaran balik sebanyak 10, kemudian dalam sekali epochs dilakukan 32 sampel yang diproses sebelum model diperbarui. Dilakukan validation_split sebesar 20% untuk melakukan pengecekan pada cross score validation yang telah dilakukan.

- Ilustrasi Gambar : 6.20

```
In [4]: model.FitAip(test_input, test_labels, epochs=10, batch_size=32)
WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/tensorflow/python/math/math_ops.py:1360: 
    _tg_1023 (from tensorflow.python.ops.math_ops) is deprecated and will be removed in a future
    version.
    Instructions for updating:
    Use tf.math.log10 instead.
Epoch 1/10
1/10 [00:00 < 00:00] - loss: 1.8792 - acc: 0.2000 - val_loss:
1.8792 - val_acc: 0.3784
Epoch 2/10
1/10 [00:00 < 00:00] - loss: 1.8792 - acc: 0.2000 - val_loss:
1.8792 - val_acc: 0.3784
Epoch 3/10
1/10 [00:00 < 00:00] - loss: 1.8792 - acc: 0.2000 - val_loss:
1.8792 - val_acc: 0.3784
Epoch 4/10
1/10 [00:00 < 00:00] - loss: 1.8792 - acc: 0.2000 - val_loss:
1.8792 - val_acc: 0.3784
Epoch 5/10
1/10 [00:00 < 00:00] - loss: 1.8792 - acc: 0.2000 - val_loss:
1.8792 - val_acc: 0.3784
Epoch 6/10
1/10 [00:00 < 00:00] - loss: 1.8792 - acc: 0.2000 - val_loss:
1.8792 - val_acc: 0.3784
Epoch 7/10
1/10 [00:00 < 00:00] - loss: 1.8792 - acc: 0.2000 - val_loss:
1.8792 - val_acc: 0.3784
Epoch 8/10
1/10 [00:00 < 00:00] - loss: 1.8792 - acc: 0.2000 - val_loss:
1.8792 - val_acc: 0.3784
Epoch 9/10
1/10 [00:00 < 00:00] - loss: 1.8792 - acc: 0.2000 - val_loss:
1.8792 - val_acc: 0.3784
Epoch 10/10
1/10 [00:00 < 00:00] - loss: 1.8792 - acc: 0.2000 - val_loss:
1.8792 - val_acc: 0.3784
```

Figure 6.20: Fungsi Fit-Aip

10. Penjelasan Masing-Masing Parameter Dari Fungsi Evaluate().

- Code Yang Digunakan :

```
1 # -*- coding: utf-8 -*-
2 """
3 Created on Wed Apr  4 02:09:28 2019
4
5 @author: kiting13
6 """
7
8
9 loss , acc = model.evaluate(test_input , test_labels , batch_size
10 =32)
11 #%%%
12 print("Done!")
13 print("Loss: %.4f , accuracy: %.4f" % (loss , acc))
```

- Penjelasan :

Berdasarkan code 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.

Kemudian pada hasilnya sendiri dapat dilihat bahwa Loss merupakan hasil prediksi yang salah sebanyak 1,7985 dan keakurasiannya prediksinya sebesar 0,4200.

- Ilustrasi Gambar : 6.21

```
In [5]: loss, acc = model.evaluate(test_input, test_labels, batch_size=32)
200/200 [=====] - 0s 2ms/step
In [5]: print("Done!")
Done!
In [5]: print("Loss: %.4f , accuracy: %.4f" % (loss, acc))
Loss: 1.7985, accuracy: 0.4200
```

Figure 6.21: Fungsi Evaluate - Aip

11. Penjelasan Masing-Masing Parameter Dari Fungsi Predict().

- Code Yang Digunakan :

```
1 # -*- coding: utf-8 -*-
2 """
3 Created on Wed Apr  4 02:13:55 2019
```

```

4
5 @author: kiting13
6 """
7
8 model.predict(test_input [:1])

```

- Ilustrasi Gambar : 6.22

Penjelasan :

Code tersebut dipergunakan untuk melakukan prediksi diambil dari satu baris berdasarkan test_input . Nilai yang tertinggi terdapat pada label kedua yang dipilih prediksi yang tepat kemudian akan dikelompokkan.

```
In [22]: model.predict(test_input[:1])
Out[22]:
array([0.18210195, 0.88640732, 0.17503217, 0.13698447, 0.0812095,
       0.01279172, 0.00026373, 0.21091531, 0.04761446, 0.21170557],)
```

Figure 6.22: Fungsi Predict - Aip

6.1.3 Penanganan Error

```

1 # -*- coding: utf-8 -*-
2 """
3 Created on Wed Apr  3 18:50:04 2019
4
5 @author: kiting13
6 """
7
8 NameError: name 'librosa' is not defined

```

1. Skrinsut Error

```

File "<ipython-input-1-59953e4547ad>", line 2, in display_mfcc
    y, _ = librosa.load(song)

NameError: name 'librosa' is not defined

```

Figure 6.23: Error Aip

2. Kode Error dan Jenis Errornya

Kode Error: "Name Error" name 'librosa' is not defined

3. Penanganan

Mendefinisikan library librosa

6.2 Andi Muh Aslam

6.2.1 Teori

1. Kenapa File Suara Harus Dilakukan MFCC

- Penjelasan: Agar bisa mengubah suara menjadi vektor. Sehingga data suara bisa diolah menjadi outputan. Jadi semua parameter inputan baik itu suara, dokumen harus dipersiapkan datanya terlebih dahulu, kalau untuk dokumen untuk teks menggunakan word2vec, sedangkan untuk suara menggunakan MFCC (Mel Frequency Cepstral Coeficients)
- Ilustrasi Gambar

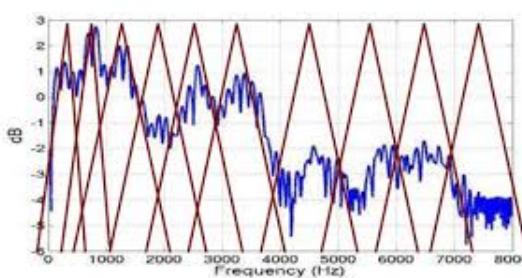


Figure 6.24: MFCC

2. Konsep Dasar Neural Network

- Penjelasan:

Neural Network merupakan sistem komputasi yang efisien dimana tema utamanya dipinjam dari analogi jaringan saraf biologis. Neural Network biasa disebut dengan jaringan saraf tiruan dimana Neural Network sebenarnya mengadopsi dari kemampuan otak manusia yang mampu memberikan stimulasi/rangsangan, melakukan proses, dan memberikan output.

- Ilustrasi Gambar

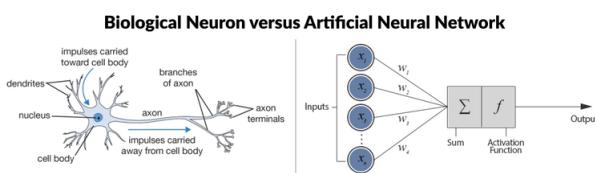


Figure 6.25: Konsep Dasar Neural Network

3. Konsep Pembobotan Dalam Neural Network

- Penjelasan:

Cara kerja Neural Network dapat dianalogikan sebagaimana halnya manusia belajar dengan menggunakan contoh atau yang disebut sebagai supervised learning. Sebuah Neural Network dikonfigurasi untuk aplikasi tertentu, seperti pengenalan pola atau klasifikasi data, dan kemudian disempurnakan melalui proses pembelajaran. Proses belajar yang terjadi dalam sistem biologis melibatkan penyesuaian koneksi sinaptik yang ada antara neuron, dalam halnya pada Neural Network penyesuaian koneksi sinaptik antar neuron dilakukan dengan menyesuaikan nilai bobot yang ada pada tiap koneksi baik dari input, neuron maupun output.

- Ilustrasi Gambar

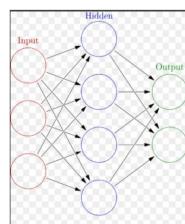


Figure 6.26: Konsep Pembobotan

4. Konsep Fungsi Aktivasi Dalam Neural Network

- Penjelasan:

Setiap neuron 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.

- Ilustrasi Gambar

5. Cara Membaca Hasil Plot Dari MFCC

- Penjelasan:

Cara membaca hasil plot 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

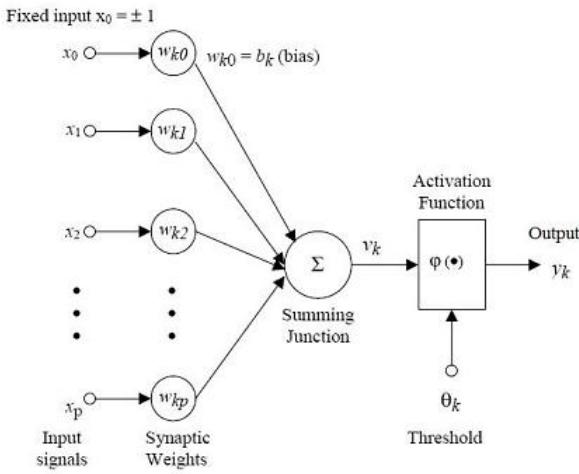


Figure 6.27: Konsep Aktivasi

simbol sumbu x. Sedangkan pada bagian dalam atau bisa disimbolkan dengan sumbu z merupakan power atau kekuatan dari lagu atau suara atau desibel 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.

- Ilustrasi Gambar

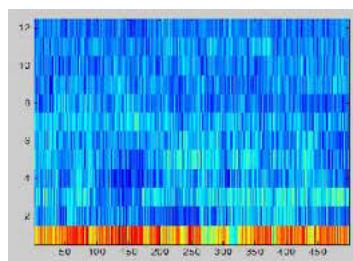


Figure 6.28: Cara Membaca Hasil Plot MLCC

6. Apa Itu One-Hot Encoding?

- Penjelasan:

One-hot encoding adalah representasi dari variabel kategori sebagai vektor biner. Yaitu nilai kategorika harus dipetakan ke nilai integer. Kemudian,

setiap nilai integer direpresentasikan sebagai vektor biner yang semuanya bernilai nol kecuali indeks integer, yang ditandai dengan 1.

- Ilustrasi Gambar

color	color_red	color_blue	color_green
red	1	0	0
green	0	0	1
blue	0	1	0
red	1	0	0

Figure 6.29: One Hot Encoding

7. Fungsi Dari np.unique dan to_categorical dalam kode program

- Np.unique
- Penjelasan: np.unique untuk mengekstaksi elemen-elemen unik tertentu dalam array.

```
>>> 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 6.30: NP Unique

- To categorical
- Penjelasan: Berfungsi untuk mengubah vektor kelas yang berupa integer (number) menjadi matriks kelas biner.

```
> 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)
```

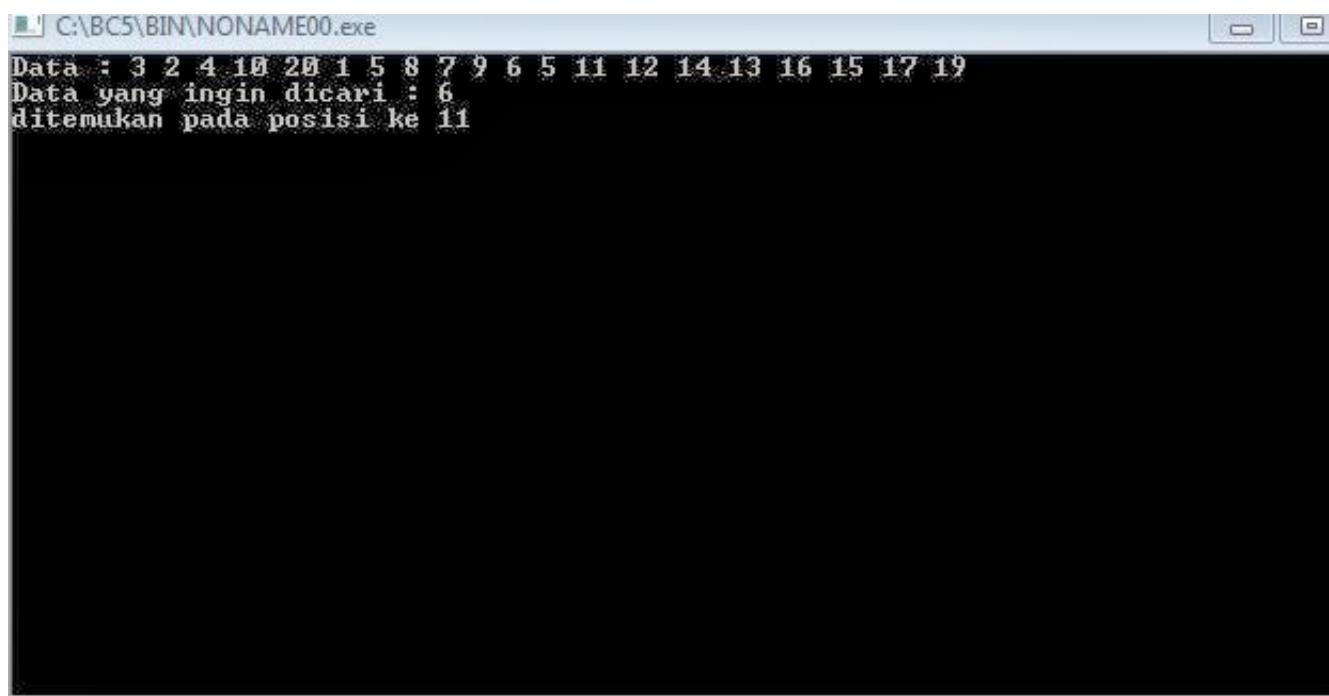
Figure 6.31: To Categorical

8. Fungsi Dari Sequential Pada Kode Program

- Penjelasan:

Fungsi dari Sequential dalam kode program yaitu merupakan sebuah jenis model yang digunakan dalam perhitungan ataupun code program yang direalisasikan. Neural Networks Sequential 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.

- Ilustrasi Gambar



The screenshot shows a terminal window with the title bar 'C:\BC5\BIN\NONAME00.exe'. The window contains the following text:
Data : 3 2 4 10 20 1 5 8 7 9 6 5 11 12 14 13 16 15 17 19
Data yang ingin dicari : 6
ditemukan pada posisi ke 11

Figure 6.32: Sequential

6.2.2 Praktek

1. Menjelaskan Isi Dari Data GTZAN Genre Colection dan Data Freesound

Dataset GTZAN memiliki 1000 potongan file musik yang berdurasi kurang lebih sekitar 30 detik untuk meload data kita akan menggunakan library python, librosa untuk mengekstrak fitur dari lagu yang terdiri dari 10 kelas genre yaitu blues, classical, country, disco, hip hop, jazz, metal, popular, reggae, dan rock.

Hasil :

```
In [17]: filename_jazz = 'genres/jazz/jazz.00000.au'  
...: x_jazz, sr_jazz = librosa.load(filename_jazz, duration=10)
```

Figure 6.33: NO 1

Baris 1: filename jazz merupakan code program untuk memuat inputan data jazz

Baris 2: x jazz dan y jazz merupakan variable yang datanya akan di load menggunakan library librosa, yang nantinya akan di gunakan pada MFCC

2. Menjelaskan Perbaris Kode Program Fungsi Dari display mfcc()

- Kode Program

Kode Program 2?? :

```
In [2]: def display_mfcc(song):  
...:     y, _ = librosa.load(song)  
...:     mfcc = librosa.feature.mfcc(y)  
...:  
...:     plt.figure(figsize=(10, 4))  
...:     librosa.display.specshow(mfcc, x_axis='time', y_axis='mel')  
...:     plt.colorbar()  
...:     plt.title(song)  
...:     plt.tight_layout()  
...:     plt.show()
```

Figure 6.34: Kode Program No 2

- Hasil

```
In [7]: display_mfcc('genres/classical/classical.00067.au')
```

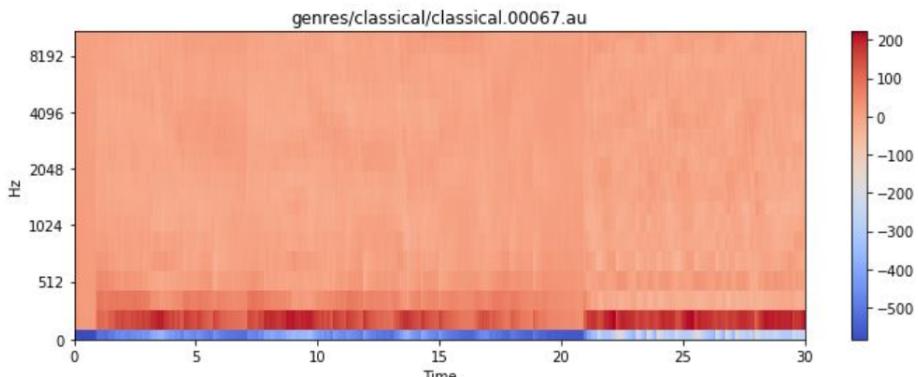


Figure 6.35: Hasil No 2

Baris 1: Untuk menampilkan vektorisasi sebuah suara

Baris 2: membuat variable y agar dapat membaca variable song pada library load song

Baris 3: Untuk mengubah variable y menjadi suara vektor

Baris 4: Mengubah plot gambar dengan ukuran 10x4

Baris 5: Menampilkan Spektogram atau chromagram agar code program dapat berwarna

Baris 6: menambahkan colorbar pada plot

Baris 7: menetapkan judul lagu

Baris 8: memberikan label pada grafik

Baris 9: Menampilkan hasil plot

3. Jelaskan Kode Program extract features song(). Mengapa Yang Diambil 25000 Baris Pertama

- Penjelasan Kode program

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

Figure 6.36: Hasil No 3

Baris 1: Membuat fungsi extract features song dengan inputan f

Baris 2: Membuat variabel y untuk meload inputan f dari perintah librosa load song

Baris 3: Membuat variabel mfcc untuk membuat feature dari variabel y

Baris 4: Membuat normalisasi nilai antara -1 sampai 1

Baris 5: Mengambil 25000 data pertama berdasarkan durasi suara lalu dikembalikan salinan arraynya dan dikecilkan menjadi satu.

- Mengapa Yang Diambil 25000 Baris Pertama

Diambil 25000 baris pertama karena agar eksekusi data atau saat running tidak memakan waktu yang cukup lama.

6.2.3 Penanganan Eror

- (a) ScreenShoot Error
- (b) Tuliskan kode eror dan jenis errornya.
- (c) Solusi pemecahan masalah error tersebut.

Chapter 7

Discussion

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

7.1 Andi Muhammad Aslam/1164064

7.1.1 Teori

1. Jelaskan kenapa file teks harus di lakukan tokenizer.

Tahap Tokenizing adalah tahap pemotongan string input berdasarkan tiap kata yang menyusunnya. Tokenisasi secara garis besar memecah sekumpulan karakter dalam suatu teks ke dalam satuan kata, bagaimana membedakan karakter-karakter tertentu yang dapat diperlakukan sebagai pemisah kata atau bukan. Sebagai contoh karakter whitespace, seperti enter, tabulasi, spasi dianggap sebagai pemisah kata. Namun untuk karakter petik tunggal (), titik (.), semikolon (;), titik dua (:) atau lainnya, dapat memiliki peran yang cukup banyak sebagai pemisah kata.

2. Jelaskan konsep dasar K Fold Cross Validation pada dataset komentar Youtube pada kode listing 7.1.dilengkapi dengan ilustrasi atau gambar.

Starfiedkfold yaitu mengembalikan lipatan data bertingkat. Lipatan dibuat dengan mempertahankan persentase sampel setiap kelas dari dataset youtube. dari contoh sampel dalam ilustrasi mempresentasikan sampel dataset menjadi 5 bagian di setiap kelasnya.

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

Listing 7.1: K Fold Cross Validation

3. Berikut ilustrasi pada K Fold Cross Validation :



Figure 7.1: Tokenizer

```
>>> from sklearn.model_selection import StratifiedKFold
>>> X = np.ones(10)
>>> y = [0, 0, 0, 0, 1, 1, 1, 1, 1]
>>> skf = StratifiedKFold(n_splits=3)
>>> for train, test in skf.split(X, y):
...     print("%s %s" % (train, test))
[2 3 6 7 8 9] [0 1 4 5]
[0 1 3 4 5 8 9] [2 6 7]
[0 1 2 4 5 6 7] [3 8 9]
```

Figure 7.2: KFold Cross validation

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

Maksudnya For Train, test in split untuk menguji dataset yang sudah di split sebelumnya agar tidak terjadi penumpukan, yaitu dengan cepat dapat mengelompokkan validasi input data dan dapat di ShuffleSplit dan aplikasi untuk memasukkan data ke dalam satu panggilan tunggal agar dapat memisahkan data secara opsional dan subsampling.

- Berikut ilustrasi pada for train, test in splits :

```
>>> import numpy as np
>>> from sklearn.cross_validation import train_test_split
>>> a, b = np.arange(10).reshape((5, 2)), range(5)
>>> a
array([[0, 1],
       [2, 3],
       [4, 5],
       [6, 7],
       [8, 9]])
>>> list(b)
[0, 1, 2, 3, 4]
```

Figure 7.3: Ilustrasi for train, test in splits

- Jelaskan apa maksudnya kode program train content = d[CONTENT].iloc[train idx] dan test content = d[CONTENT].iloc[test idx]. dilengkapi dengan ilustrasi

atau gambar.

Maksudnya untuk mengambil data pada Index CONTENT yang merupakan bagian dari train_idx dan test_idx. Ilustrasi, apabila data diubah menjadi train content dan test content maka kita dapat memilih content apa yang ingin ditampilkan pada kolom.

7. Jelaskan apa maksud dari fungsi tokenizer = Tokenizer(num words=2000) dan tokenizer.fit on texts(train content), dilengkapi dengan ilustrasi atau gambar.

Dimana Tokenizer yang akan melakukan vektorisasi pada kata yang dimana jumlah kata yang ingin diubah kedalam bentuk token adalah 2000 kata. Untuk tokenizer.fit on texts(train content) maksudnya yaitu untuk melakukan fit tokenizer hanya untuk data train saja, tidak berlaku pada data test nya untuk kolom CONTENT. Ilustrasi, " bayi itu sedang tertidur lelap" maksudnya ini akan membuat kamus s.t word_index ["tidur"] = 0; word_index ["bayi"] = 1; itu merupakan kata kamus indeks sehingga setiap kata mendapatkan nilai integer yang unik.

8. Jelaskan apa maksud dari fungsi d train inputs = tokenizer.texts to matrix(train content, mode=tfidf) dan d test inputs tokenizer.texts to matrix(test content, mode=tfidf), dilengkapi dengan ilustrasi kode dan atau gambar.

fungsi tersebut yang akan membagi matrix TF IDF dengan amax yaitu mengembalikan nilai max array atau nilai max sepanjang sumbu. yang hasilnya akan dimasukkan kedalam variable (train input) untuk daata train dan (test input) untuk data test dengan nominal tanpa adanya bilangan negatif dan koma.

9. Jelaskan apa maksud dari fungsi d train inputs = d train inputs/np.amax(np.absolute(d train dan d test inputs = d test inputs/np.amax(np.absolute(d test inputs))), dilengkapi dengan ilustrasi atau gambar.

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.

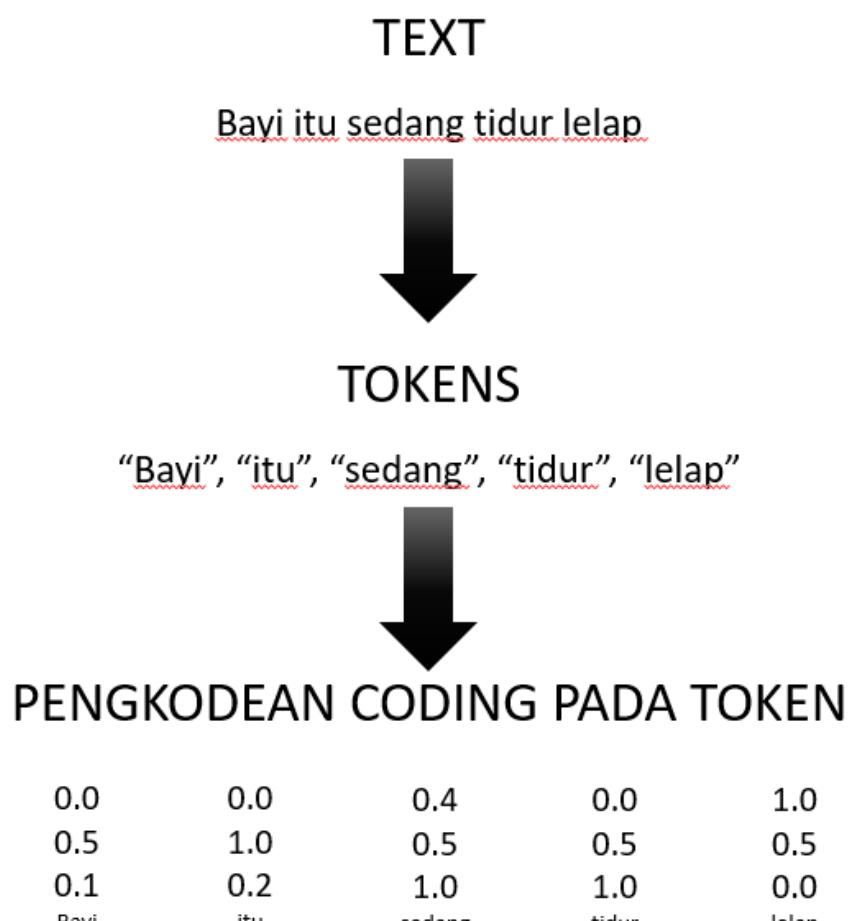


Figure 7.4: Fungsi Tokenizer

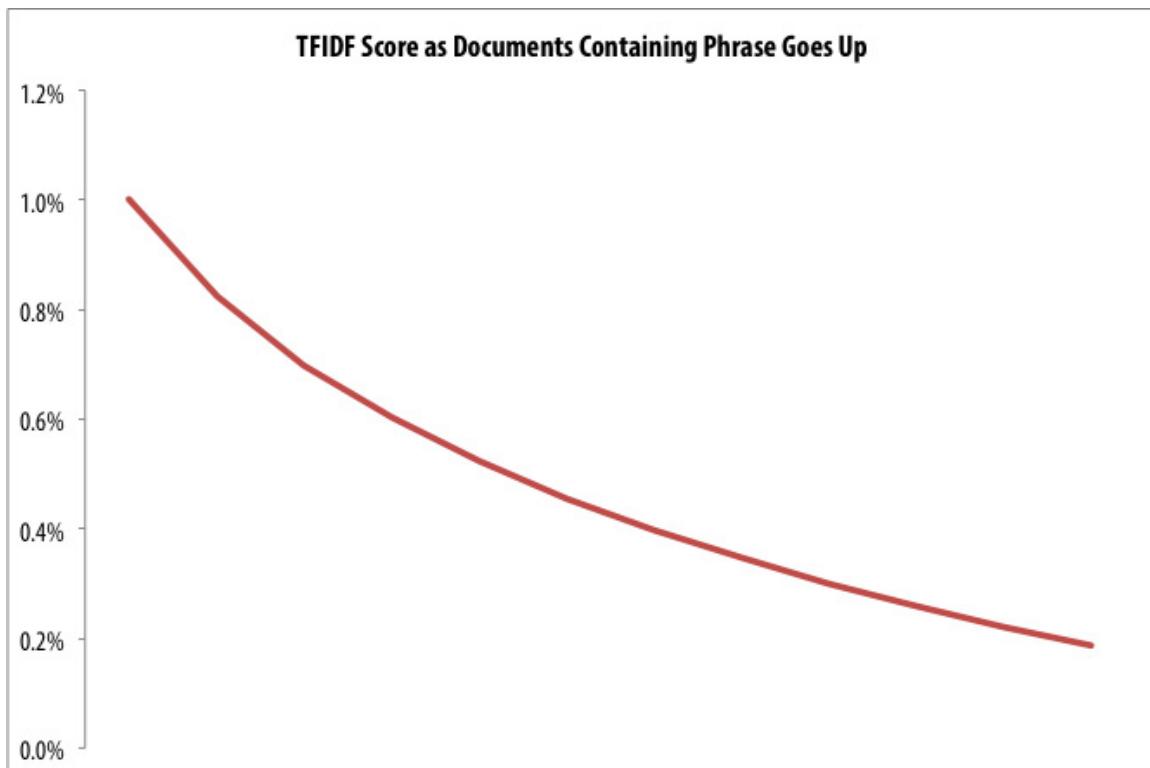


Figure 7.5: Matrix TF IDF

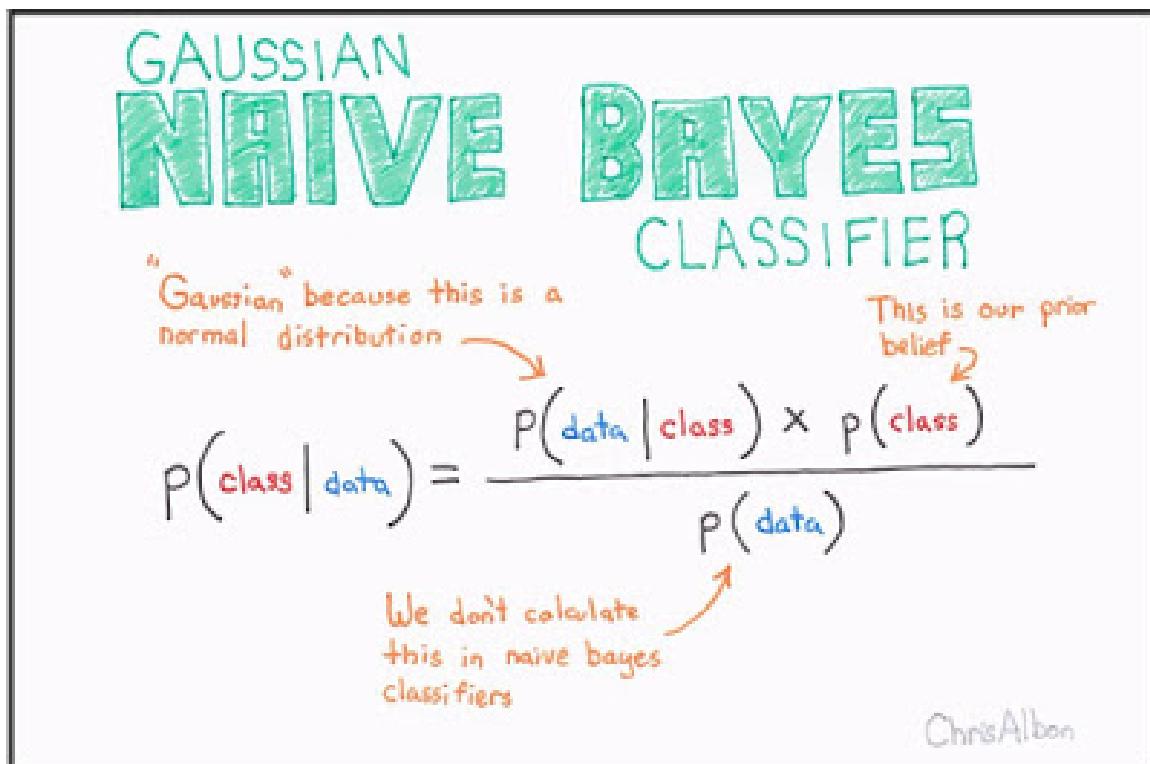


Figure 7.6: Matrix TFID

10. Jelaskan apa maksud fungsi dari `d train outputs = np utils.to_categorical(d['CLASS'].iloc[train_idx])` 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.

```
# 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 7.7: Matrix Training dan Testing

11. Jelaskan apa maksud dari fungsi di listing.

```
1 # -*- coding: utf-8 -*-
2 """
3 Created on Fri Apr  3 19:59:26 2019
4
5 @author: kiting13
6 """
7
8 model = Sequential()
9     model.add(Dense(512, input_shape=(2000,)))
10    model.add(Activation('relu'))
11    model.add(Dropout(0.5))
12    model.add(Dense(2))
13    model.add(Activation('softmax'))
```

12. Dari fungsi pada kode program tersebut ditujukan untuk melakukan pemodelan dengan sequential, membandingkan setiap satu larik elemen dengan cara satu persatu secara beruntun. Dimana terdapat 512 neuron inputan dengan input shape 2000 vektor. Lalu model dilakukan aktivasi dengan fungsi 'relu'. Kemudian dilakukan pemotongan bobot supaya tidak overfitting sebesar 50 persen dari neuron inputan 512. Lalu pada layer output terdapat 2 neuron outputan yaitu nol(1,0) atau nol satu(0,1). Kemudian outputan tersebut diaktivasi menggunakan fungsi softmax.

13. Jelaskan apa maksud dari fungsi di listing.

```
1 # -*- coding: utf-8 -*-
2 """
3 Created on Fri Apr  3 19:59:26 2019
4
5 @author: kiting13
6 """
7
8 model.compile(loss='categorical_crossentropy', optimizer='adamax',
    metrics=['accuracy'])
```

- Dari fungsi pada kode program tersebut model yang telah dibuat selanjutnya dicompile dengan menggunakan algoritma optimisasi dengan fungsi-fungsi loss, fungsi optimaizer dan fungsi metrik. Dimana nama masing-masing fungsi tersebut adalah categorical_crossentropy, adamax dan accuracy.

14. Jelaskan apa itu Deep Learning.

Deep Learning adalah salah satu jenis algoritma jaringan saraf tiruan yang menggunakan metadata sebagai input dan mengolahnya menggunakan sejumlah lapisan tersembunyi (hidden layer) transformasi non linier dari data masukan untuk menghitung nilai output. Algoritma pada Deep Learning memiliki fitur yang unik yaitu sebuah fitur yang mampu mengekstraksi secara otomatis.

15. Jelaskan apa itu Deep Neural dan bedanya dengan Deep Learning.

Deep Neural Network atau DNN merupakan algoritma yang berbasis neural network yang digunakan untuk mengambil keputusan.

16. Yang membedakan Deep Learning dengan Deep Neural Network (DNN)

DNN merupakan algoritma yang digunakan pada Deep Learning, sedangkan Deep Learning merupakan model yang menggunakan algoritma DNN.

17. Jelaskan 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.

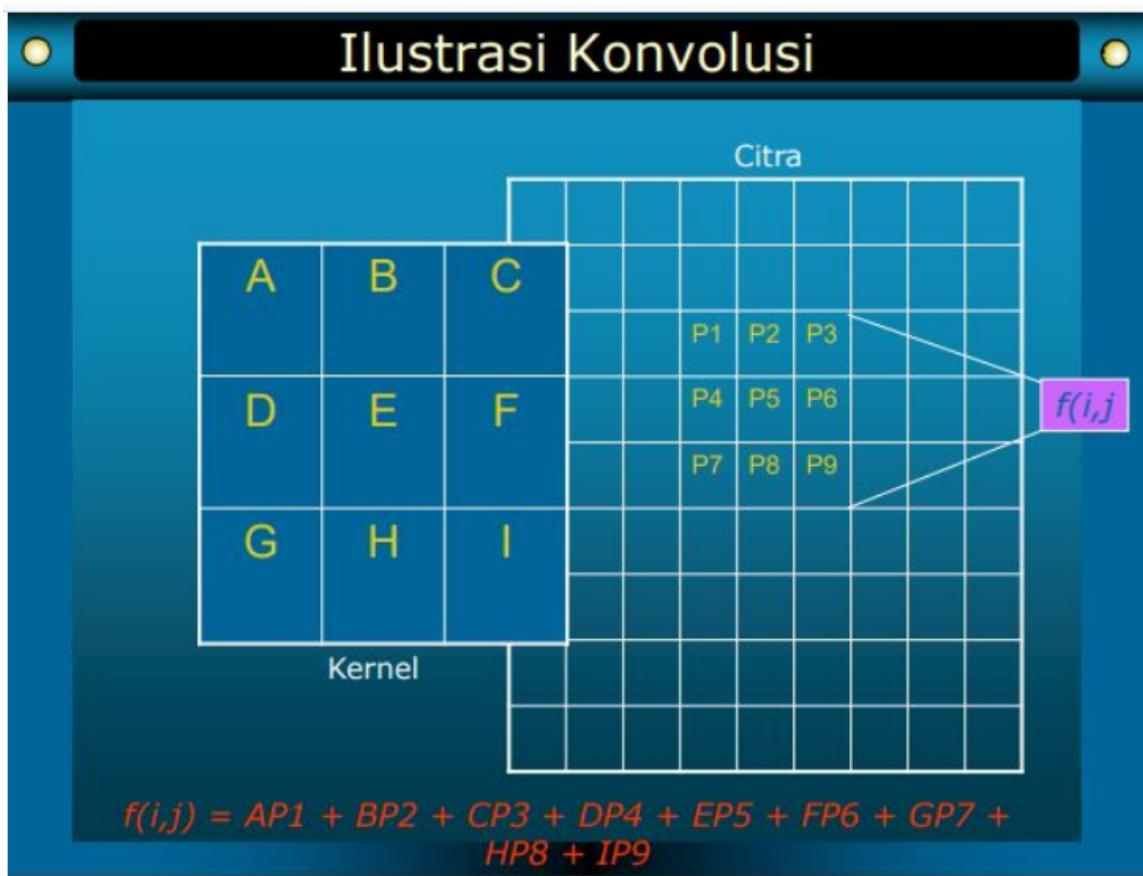


Figure 7.8: Konvolusi

7.1.2 Praktek Program

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

Berikut adalah kode program yang digunakan :

```

1 import csv
2 from PIL import Image as pilimage
3 import keras.preprocessing.image

```

Listing 7.2: Kode Program 1

Npm saya adalah 1164064 hasil dari ($NPM = 1164064 \bmod 3+1 = 2$, maka saya menggunakan matrix kernel berukuran 3×3 .

Contoh : $f(x,y)$ yang digunakan berukuran 4×4 dan kernel berukuran 3×3 dengan data masing-masing sebagai berikut :

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

Penyelesaian pada algoritma perhitungan konvolusi antara $f(x,y)$ dan $g(x,y)$ yaitu $f(x,y) * g(x,y)$. 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 matriks kernel dengan 2×2 seperti berikut ini:

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

Figure 7.9: Algoritma Perhitungan Konvolusi

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

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

Figure 7.10: In[1]

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

Berikut adalah kode program yang digunakan :

```
1 imgs = []
2 classes = []
```

```

3 with open('Chapter04/hasy-data-labels.csv') as csvfile:
4     csvreader = csv.reader(csvfile)
5     i = 0
6     for row in csvreader:
7         if i > 0:
8             img = keras.preprocessing.image.img_to_array(pil_image.
9                 open("Chapter04/" + row[0]))
10            # neuron activation functions behave best when input
11            # values are between 0.0 and 1.0 (or -1.0 and 1.0),
12            # so we rescale each pixel value to be in the range 0.0
13            # instead of 0-255
14            img /= 255.0
15            imgs.append((row[0], row[2], img))
16            classes.append(row[2])
17
18    i += 1

```

Listing 7.3: Kode Program 2

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

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 7.11: In[2]

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

Berikut adalah kode program yang digunakan :

```

1 import random
2 random.shuffle(imgs)
3 split_idx = int(0.8*len(imgs))
4 train = imgs[:split_idx]
5 test = imgs[split_idx:]

```

Listing 7.4: Kode Program 3

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

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

Figure 7.12: In[3]

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

Berikut adalah kode program yang digunakan :

```

1 import numpy as np
2
3 train_input = np.asarray(list(map(lambda row: row[2], train)))
4 test_input = np.asarray(list(map(lambda row: row[2], test)))
5
6 train_output = np.asarray(list(map(lambda row: row[1], train)))
7 test_output = np.asarray(list(map(lambda row: row[1], test)))

```

Listing 7.5: Kode Program 4

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 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 [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 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 [2] difungsikan untuk membuat objek menjadi lebih kecil sehingga mudah dieksekusi dari variabel test.

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

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

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

Figure 7.13: In[4]

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

Berikut adalah kode program yang digunakan :

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

Listing 7.6: Kode Program 5

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

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

Figure 7.14: In[5]

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

Berikut adalah kode program yang digunakan :

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

Listing 7.7: Kode Program 6

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

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

Figure 7.15: In[6]

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

Berikut adalah kode program yang digunakan :

```

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

```

Listing 7.8: Kode Program 7

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

```

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 7.16: In[7]

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

Berikut adalah kode program yang digunakan :

```

1 train_output_int = label_encoder.transform(train_output)
2 train_output = onehot_encoder.transform(train_output_int.reshape(
    len(train_output_int), 1))
3 test_output_int = label_encoder.transform(test_output)
4 test_output = onehot_encoder.transform(test_output_int.reshape(len(
    test_output_int), 1))
5
6 num_classes = len(label_encoder.classes_)
7 print("Number of classes: %d" % num_classes)

```

Listing 7.9: Kode Program 8

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

Number of classes: 369

Figure 7.17: In[8]

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

Berikut adalah kode program yang digunakan :

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

Listing 7.10: Kode Program 9

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

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

Figure 7.18: In[9]

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

Berikut adalah kode program yang digunakan :

```
1 model = Sequential()
2 model.add(Conv2D(32, kernel_size=(3, 3), activation='relu',
3                  input_shape=np.shape(train_input[0])))
4 model.add(MaxPooling2D(pool_size=(2, 2)))
5 model.add(Conv2D(32, (3, 3), activation='relu'))
6 model.add(MaxPooling2D(pool_size=(2, 2)))
7 model.add(Flatten())
8 model.add(Dense(1024, activation='tanh'))
9 model.add(Dropout(0.5))
10 model.add(Dense(num_classes, activation='softmax'))
11
12 model.compile(loss='categorical_crossentropy', optimizer='adam',
13                 metrics=['accuracy'])
14
15 print(model.summary())
```

Listing 7.11: Kode Program 10

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

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

Berikut adalah kode program yang digunakan :

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

Listing 7.12: Kode Program 11

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

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

Berikut adalah kode program yang digunakan :

Layer (type)	Output Shape	Param #
conv2d_1 (Conv2D)	(None, 30, 30, 32)	896
max_pooling2d_1 (MaxPooling2D)	(None, 15, 15, 32)	0
conv2d_2 (Conv2D)	(None, 13, 13, 32)	9248
max_pooling2d_2 (MaxPooling2D)	(None, 6, 6, 32)	0
flatten_1 (Flatten)	(None, 1152)	0
dense_1 (Dense)	(None, 1024)	1180672
dropout_1 (Dropout)	(None, 1024)	0
dense_2 (Dense)	(None, 369)	378225
Total params:	1,569,041	
Trainable params:	1,569,041	
Non-trainable params:	0	
None		

Figure 7.19: In[10]

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

Figure 7.20: In[11]

```

1 model.fit(train_input, train_output,
2           batch_size=32,
3           epochs=10,
4           verbose=2,
5           validation_split=0.2,
6           callbacks=[tensorboard])
7
8 score = model.evaluate(test_input, test_output, verbose=2)
9 print('Test loss:', score[0])
10 print('Test accuracy:', score[1])

```

Listing 7.13: Kode Program 12

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
- Vebrose=2 maksudnya menampilkan nomor dari epoch yang sedang berjalan atau yang sudah dijalankan.
- Validasi plit sebanayk 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 lost 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

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

Berikut adalah kode program yang digunakan :

```

1 import time
2
3 results = []
4 for conv2d_count in [1, 2]:
5     for dense_size in [128, 256, 512, 1024, 2048]:
6         for dropout in [0.0, 0.25, 0.50, 0.75]:
7             model = Sequential()
8             for i in range(conv2d_count):
9                 if i == 0:

```

```

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

```

Figure 7.21: In[12]

```

10     model.add(Conv2D(32, kernel_size=(3, 3),
11                     activation='relu', input_shape=np.shape(train_input[0])))
12                     else:
13                         model.add(Conv2D(32, kernel_size=(3, 3),
14                     activation='relu'))
15                         model.add(MaxPooling2D(pool_size=(2, 2)))
16                         model.add(Flatten())
17                         model.add(Dense(dense_size, activation='tanh'))
18                         if dropout > 0.0:
19                             model.add(Dropout(dropout))
20                         model.add(Dense(num_classes, activation='softmax'))
21
22                         model.compile(loss='categorical_crossentropy',
23                         optimizer='adam', metrics=['accuracy'])
24
25                         log_dir = './logs/conv2d_%d-dense_%d-dropout_.%2f' % (
26 conv2d_count, dense_size, dropout)
27                         tensorboard = keras.callbacks.TensorBoard(log_dir=
log_dir)
28
29                         start = time.time()
30                         model.fit(train_input, train_output, batch_size=32,
31 epochs=10,
32                         verbose=0, validation_split=0.2, callbacks=[tensorboard])
33                         score = model.evaluate(test_input, test_output, verbose
=2)
34                         end = time.time()
35                         elapsed = end - start
36                         print("Conv2D count: %d, Dense size: %d, Dropout: %.2f
- Loss: %.2f, Accuracy: %.2f, Time: %d sec" % (conv2d_count,
dense_size, dropout, score[0], score[1], elapsed))

```

```
32     results.append((conv2d_count, dense_size, dropout,
    score[0], score[1], elapsed))
```

Listing 7.14: Kode Program 13

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

```
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 7.22: In[13]

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

Berikut adalah kode program yang digunakan :

```
1 model = Sequential()
2 model.add(Conv2D(32, kernel_size=(3, 3), activation='relu',
      input_shape=np.shape(train_input[0])))
3 model.add(MaxPooling2D(pool_size=(2, 2)))
4 model.add(Conv2D(32, (3, 3), activation='relu'))
5 model.add(MaxPooling2D(pool_size=(2, 2)))
6 model.add(Flatten())
7 model.add(Dense(128, activation='tanh'))
8 model.add(Dropout(0.5))
9 model.add(Dense(num_classes, activation='softmax'))
10 model.compile(loss='categorical_crossentropy', optimizer='adam',
     metrics=['accuracy'])
11 print(model.summary())
```

Listing 7.15: Kode Program 14

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 digubakan 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

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

Berikut adalah kode program yang digunakan :

```

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

```

Listing 7.16: Kode Program 15

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

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

Berikut adalah kode program yang digunakan :

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 7.23: 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 7.24: In[15]

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

Listing 7.17: Kode Program 16

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

```
In [16]: model.save("mathsymbols.model")
└── mathsymbols.model 13/04/2019 03.45 MODEL File 2.443 KB
```

Figure 7.25: In[16]

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

Berikut adalah kode program yang digunakan :

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

Listing 7.18: Kode Program 17

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

```
In [17]: np.save('classes.npy', label_encoder.classes_)
└── classes.npy 13/04/2019 03.45 NPY File 28 KB
```

Figure 7.26: In[17]

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

Berikut adalah kode program yang digunakan :

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

Listing 7.19: Kode Program 18

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

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

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

Figure 7.27: In[18]

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

Berikut adalah kode program yang digunakan :

```
1 label_encoder2 = LabelEncoder()
2 label_encoder2.classes_ = np.load('classes.npy')
3
4 def predict(img_path):
5     newimg = keras.preprocessing.image.img_to_array(pil_image.open(
6         img_path))
7     newimg /= 255.0
8
9     # do the prediction
10    prediction = model2.predict(newimg.reshape(1, 32, 32, 3))
```

```

10
11     # figure out which output neuron had the highest score , and
12     # reverse the one-hot encoding
13     inverted = label_encoder2.inverse_transform([np.argmax(
14         prediction)]) # argmax finds highest-scoring output
15     print("Prediction: %s, confidence: %.2f" % (inverted[0], np.max(
16         prediction)))

```

Listing 7.20: Kode Program 19

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

```

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

```

Figure 7.28: In[19]

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

Berikut adalah kode program yang digunakan :

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

Listing 7.21: Kode Program 20

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

- 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

```
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 7.29: In[20]

7.1.3 Penanganan Eror

1. skrinsut error
2. Tuliskan kode eror dan jenis errornya

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

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.

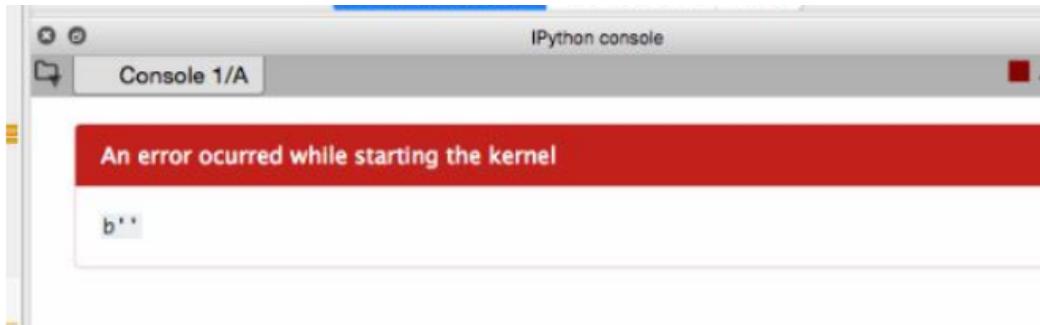


Figure 7.30: Error

7.2 Aip Suprapto Munari-1164063

7.2.1 Teori

1. Kenapa File Suara Harus Dilakukan Tokenizer

- Penjelasan: Untuk membedakan karakter-karakter tertentu dalam suatu teks dan juga sebagai pemisah kata atau bukan. Tokenizer dilakukan dengan cara melakukan pemotongan string input berdasarkan tiap kata yang menyusunnya.
- Ilustrasi Gambar
- Tokenizer 7.31

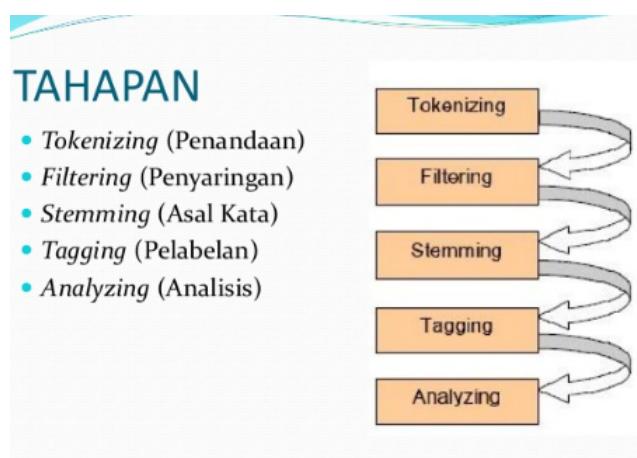


Figure 7.31: Tokenizer Aip

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

```

1 # -*- coding: utf-8 -*-
2 """
3 Created on Wed Apr  3 19:59:26 2019
4
5 @author: kiting13
6 """
7
8 kfolds = StratifiedKFold(n_splits=5)
9 splits = kfolds.split(d, d['CLASS'])

```

- Penjelasan: Startified KFold 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.
- Ilustrasi Gambar
- K-Fold Cross Validation 7.32

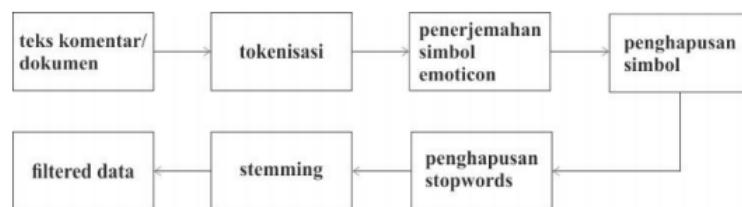


Figure 7.32: K-Fold Cross Validation Aip

3. Jelaskan apa maksudnya kode program for train, test in splits.dilengkapi dengan ilustrasi atau gambar.
 - Penjelasan: Maksudnya yaitu untuk menguji apakah setiap data pada dataset sudah di split dan tidak terjadi penumpukan. Yang dimana maksudnya di setiap class tidak akan muncul id yang sama. Ilustrasinya misalkan kita memiliki 4 botol minuman dengan model yang berbeda. Kemudian kita bagikan kedua anak, tentunya setiap anak yang menerima botol tidak memiliki botol minuman yang sama modelnya.
 - Ilustrasi Gambar
 - No 3 7.33
4. Jelaskan apa maksudnya kode program train_content = d['CONTENT'].iloc[train_idx] dan test_content = d['CONTENT'].iloc[test_idx].



Figure 7.33: No 3 Aip

- Penjelasan: 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.

- Ilustrasi Kalimat :

Jika dicontohkan dengan sebuah penjelasan maka bisa dikatakan apabila index CONTENT tersebut direalisasikan maka untuk data yang telah diubah menjadi training data maupun test data bisa dipilih kolom ataupun nilai apa yang akan ditampilkan dan dieksekusi sesuai dengan kebutuhan ataupun keinginan.

5. Jelaskan apa maksud dari fungsi tokenizer = Tokenizer(num_words=2000) dan tokenizer.tokenizer.fit_on_texts(train_content), dilengkapi dengan ilustrasi atau gambar.

- Penjelasan: 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 data trainnya saja tidak dengan data test nya untuk kolom CONTENT.

- Ilustrasi Gambar

- No 5 7.34

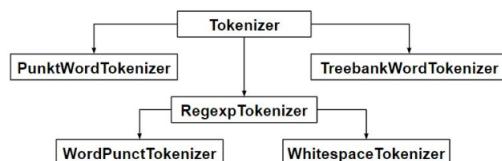


Figure 7.34: No 5 Aip

6. Jelaskan apa maksud dari fungsi *d_train_inputs = tokenizer.texts_to_matrix(train_content, mode='tfidf')* dan *d_test_inputs = tokenizer.texts_to_matrix(test_content, mode='tfidf')*, dilengkapi dengan ilustrasi kode dan atau gambar.

- Penjelasan: Maksudnya yaitu untuk variabel d_train_inputs akan melakukan tokenizer dari bentuk teks ke matrix dari data train_content dengan mode matriksnya yaitu tf-idf begitu juga dengan variabel d_test_inputs untuk data test. Berikut gambar ilustrasinya
- Ilustrasi Gambar
- No 6 7.35

Matriks

kolom 1	kolom 2	kolom 3	kolom 4
---------	---------	---------	---------

$$A = \begin{bmatrix} 1 & 4 & 1 & 1 \\ 3 & 5 & 4 & 3 \\ 6 & 2 & 1 & 7 \end{bmatrix}$$

Baris 1	Baris 2	Baris 3
---------	---------	---------

Figure 7.35: No 6 Aip

7. Jelaskan apa maksud dari fungsi $d_train_inputs = d_train_inputs / np.amax(np.absolute(d_train_inputs))$ dan $d_test_inputs = d_test_inputs / np.amax(np.absolute(d_test_inputs))$, dilengkapi dengan ilustrasi atau gambar.
 - Penjelasan: Fungsi tersebut akan membagi matrix tf-idf 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.
 - Ilustrasi Gambar
 - No 7 7.36
8. Jelaskan apa maksud fungsi dari $d_train_outputs = np_utils.to_categorical(d['CLASS'].iloc[train_idx])$ dan $d_test_outputs = np_utils.to_categorical(d['CLASS'].iloc[test_idx])$ dalam kode program
 - Penjelasan: Dari fungsi pada kode program tersebut dijelaskan fungsi tersebut ditujukan untuk melakukan one-hot encoding agar dapat masuk dan digunakan di neural network. One-hot encoding diambil dari 'CLASS' yang berarti hanya terdapat 2 neuron, yaitu satu nol(1,0) atau nol satu(0,1) karena pilihannya hanya ada dua (spam atau bukan).
 - Ilustrasi Gambar

TF-IDF

TF-IDF is a measure of originality of a word by comparing the number of times a word appears in a doc with the number of docs the word appears in.

$$TF-IDF = TF(t, d) \times IDF(t)$$

$$TF(t, d) = \frac{\text{Number of times term } t \text{ appears in a doc, } d}{\log \frac{1 + n}{1 + df(d, t)} + 1}$$

Term frequency
 ↑
 Number of times term t appears in a doc, d

Inverse document frequency
 ↑
 # of documents

Document frequency of the term t

Figure 7.36: No 7 Aip

ONE-HOT ENCODING

Feature

Apple
Pear
Apple
Pear
Apple

ONE HOT ENCODING

	Apple	Pear
Apple	1	0
Pear	0	1
Apple	1	0
Pear	0	1
Apple	1	0

Figure 7.37: No 8 Aip

- No 8 7.37

9. Jelaskan apa maksud dari fungsi di listing!

```

1 # -*- coding: utf-8 -*-
2 """
3 Created on Fri Apr  3 19:59:26 2019
4
5 @author: kiting13
6 """
7
8 model = Sequential()
9     model.add(Dense(512, input_shape=(2000,)))
10    model.add(Activation('relu'))
11    model.add(Dropout(0.5))
12    model.add(Dense(2))
13    model.add(Activation('softmax'))

```

- Penjelasan: Dari fungsi pada kode program tersebut ditujukan untuk melakukan pemodelan dengan sequential, membandingkan setiap satu larik elemen dengan cara satu persatu secara beruntun. Dimana terdapat 512 neuron inputan dengan input shape 2000 vektor. Lalu model dilakukan aktivasi dengan fungsi 'relu'. Kemudian dilakukan pemotongan bobot supaya tidak overfitting sebesar 50 persen dari neuron inputan 512. Lalu pada layer output terdapat 2 neuron outputan yaitu nol(1,0) atau nol satu(0,1). Kemudian outputan tersebut diaktivasi menggunakan fungsi softmax.

10. Jelaskan apa maksud dari fungsi di listing!

```

1 # -*- coding: utf-8 -*-
2 """
3 Created on Fri Apr  3 19:59:26 2019
4
5 @author: kiting13
6 """
7
8 model.compile(loss='categorical_crossentropy', optimizer='adamax',
9                 metrics=['accuracy'])

```

- Penjelasan: Dari fungsi pada kode program tersebut model yang telah dibuat selanjutnya dicompile dengan menggunakan algoritma optimisasi dengan fungsi-fungsi loss, fungsi optimaizer dan fungsi metrik. Dimana nama masing-masing fungsi tersebut adalah categorical_crossentropy, adamax dan accuracy.

11. Apa Itu Deep Learning

- Penjelasan:

Deep learning merupakan sub bidang pembelajaran mesin yang berkaitan dengan algoritma.

12. Apa itu Deep Neural Network Dan Apa Bedanya Dengan Deep Learning :

- Penjelasan Deep Neural Network :

Deep neural network adalah jaringan syaraf dengan tingkat kompleksitas tertentu, jaringan syaraf dengan lebih dari dua lapisan.

- Perbedaan Deep Neural Network Dan Deep Learning :

Perbedaan antara deep neural network dan deep learning terletak pada kedalaman model. deep learning adalah frasa yang digunakan untuk jaringan saraf yang kompleks. Kompleksitas ini disebabkan oleh pola yang rumit tentang bagaimana informasi dapat mengalir di seluruh model. Arsitekturnya menjadi lebih kompleks tetapi konsep deep learning masih sama. Meskipun sekarang ada peningkatan jumlah layer dan node tersembunyi yang terintegrasi untuk memperkirakan output.

13. Bagaimana Perhitungan Algoritma Dengan Ukuran Stride ($(NPM \bmod 3 + 1) \times (NPM \bmod 3 + 1)$) Yang Terdapat Pada Max Pooling :

- Penjelasan :

Konvolusi pada sebuah gambar dilakukan dalam image processing untuk menerapkan operator yang mempunyai nilai output dari piksel gambar yang berasal dari kombinasi linear nilai input piksel tertentu pada gambar.

- Langkah-langkah Algoritma Konvolusi Sesuai NPM : 7.38

- Plagiarisme Aip: ??

7.2.2 Praktek

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

- Kode Program:

```

1 # In [1]: import lib
2 import csv
3 from PIL import Image as pil_image
4 import keras.preprocessing.image

```

Listing 7.22: Praktek1.py

NPM saya 1164063 apabila diperhitung dengan operasi $(NPM \text{ mod } 3)+1$ hasil nya = 1 maka saya menggunakan matrik kernel berukuran 1×1 . sehingga perhitungan yang digunakan seperti berikut dengan $f(x,y)$ yang digunakan berukuran :

2×2 dan kernel atau mask berukuran 1×1 , masing-masing sebagai berikut:

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

keterangan = tanda * menyatakan posisi $(0,0)$ dari kernel

penyelesaian dari operasi konvolusi antara $f(x,y)$ dengan kernel $g(x,y)$ pada ketara -ngan diatas adalah $f(x,y) * g(x,y)$. menghitung posisi matriks dari ujung kiri atas pada $f(x,y)$ ke $g(x,y)$ seperti berikut :

1. $(2 \times 1), (1 \times 1), (3 \times 1) = 2, 1, 3$
2. $(3 \times 1), (4 \times 1), (3 \times 1) = 3, 4, 3$
3. $(1 \times 1), (5 \times 1), (6 \times 1) = 1, 5, 6$

langsung bisa menjadi matrix yang dijadikan sebagai hasil. seperti hasil berikut :

$$\begin{matrix} 2 & 1 & 3 \\ 3 & 4 & 3 \\ 1 & 5 & 6 \end{matrix}$$

matrix tersebut diubah menjadi angka genap menjadi -

$$\begin{matrix} -2 & 1 & 3 \\ 3 & -4 & 3 \\ 1 & 5 & -6 \end{matrix}$$

Figure 7.38: Langkah Algoritma Konvolusi Berdasarkan NPM- Aip

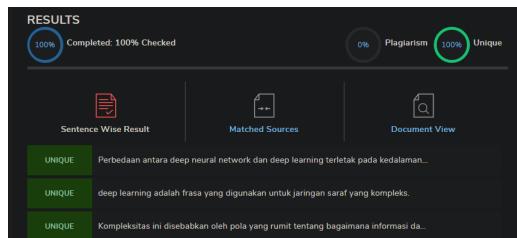


Figure 7.39: Plagiarisme- Aip

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

Figure 7.40: In 1 Aip

Hasil 7.40 :

Baris 1: Mengimpoer file csv.

Baris 2: Dari library PIL impor gambar sebagai pil_image.

Baris 3: Impor fungsi fungsi keras.preprocessing.image

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

- Kode Program:

```
1 # In [2]: load all images (as numpy arrays) and save their
2             classes
3
4 imgs = []
5 classes = []
6 with open('HASYv2/hasy-data-labels.csv') as csvfile:
7     csvreader = csv.reader(csvfile)
8     i = 0
9     for row in csvreader:
10         if i > 0:
11             img = keras.preprocessing.image.img_to_array(
12                 pil_image.open("HASYv2/" + row[0]))
13                 # neuron activation functions behave best when
14                 input values are between 0.0 and 1.0 (or -1.0 and 1.0),
15                 # so we rescale each pixel value to be in the range
16                 0.0 to 1.0 instead of 0-255
17                 img /= 255.0
18                 imgs.append((row[0], row[2], img))
19                 classes.append(row[2])
20         i += 1
```

Listing 7.23: Praktek2.py

Hasil 7.41 :

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

Figure 7.41: In 2 Aip

Baris 1: Membuat variabel images tanpa parameter, karena di dalam kurung kosong tanpa ada parameter yang diisi.

Baris 2: Membuat variabel class tanpa parameter juga karna tiada ada parameter di dalam kurung.

Baris 3: Membuka file HASYv2/hasy-data-labels.csv sebagai csvfile.

Baris 4: Membuat variabel csvreader yang memfungsikan pembacaan dari file csv yang dimasukkan

Baris 5: Membuat variabel i dengan parameter 0

Baris 6: Mengeksekusi data dari baris di pembacaan file csv.

Baris 7: Menggunakan perintah "if" dengan ketentuan variabel i lebih besar dari 0.

Baris 8: Membuat variabel img dengan nama keras.processing yang mengubah image menjadi bentuk array (bilangan) dari file HASYv2 yang dibuka dengan row berparameter 0.

Baris 9: Membuat variabel img tidak sama dengan 255.0

Baris 10: Mendefinisikan fungsi imgs.append dimana merupakan proses menggabungkan data dengan file lain yang ditentukan dengan 3 parameter yaitu row[0], row[2] dan variabel img.

Baris 11: Mendefinisikan fungsi append kembali dari variabel classes dengan parameternya row[2].

Baris 12: Mendefinisikan fungsi dimana i variabel i akan ditambah nilainya sehingga akan bernilai 1 (Contoh nilai i=0 dengan adanya penambahan maka hasilnya akan menjadi 1)

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

- Kode Program:

```
1 # In [3]: shuffle the data, split into 80% train , 20% test
2
3 import random
4 random.shuffle(imgs)
5 split_idx = int(0.8*len(imgs))
6 train = imgs[:split_idx]
7 test = imgs[split_idx:]
```

Listing 7.24: Praktek3.py

Hasil 7.42 :

Baris 1: Import modul random

Baris 2: Melakukan pengocokan terhadap modul dengan parameter dimana variabelnya imgs

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

Figure 7.42: In 3 Aip

Baris 3: Membagi dan memecah index dalam bentuk integer dengan mengkalikan nilai 0.8 dengan fungsi len yang mengembalikan jumlah item dari variabel imgs.

Baris 4: Membuat variabel train yang mengeksekusi imgs dengan pemecahan index pada data awal

Baris 5: Membuat variabel test yang mengeksekusi imgs dengan pemecahan index pada data akhir.

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

- Kode Program:

```
1 # In [4]:
2
3 import numpy as np
4
5 train_input = np.asarray(list(map(lambda row: row[2], train)))
6 test_input = np.asarray(list(map(lambda row: row[2], test)))
7
8 train_output = np.asarray(list(map(lambda row: row[1], train)))
9 test_output = np.asarray(list(map(lambda row: row[1], test)))
```

Listing 7.25: Praktek4.py

Hasil 7.43 :

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

Figure 7.43: In 4 Aip

Baris 1: Import library numpy sebagai np.

Baris 2: Membuat variabel train_input dimana mengubah input menjadi sebuah array dari np dengan menggunakan fungsi list untuk mengoleksikan data yang dipilih dan dapat diubah. Didalamnya diterapkan fungsi

map untuk mengembalikan iterator dari datanya dengan memfungsikan lamda pada row dengan parameter [2] untuk membuat objek fungsi menjadi lebih kecil dan mudah dieksekusi dari variabel train.

Baris 3: Membuat variabel test_input dengan fungsi yang sama seperti train_input yang membedakan hanya datanya / inputan yang diproses berasal dari variabel test

Baris 4: Membuat variabel train_output dimana mengubah keluaran menjadi sebuah array dari np dengan menggunakan fungsi list untuk mengoleksikan data yang dipilih dan dapat diubah. Didalamnya diterapkan fungsi map untuk mengembalikan iterator dari datanya dengan memfungsikan lamda pada row dengan parameter[1] untuk membuat objek fungsi menjadi lebih kecil dan mudah dieksekusi dari variabel train.

Baris 5: Membuat variabel test_output dengan fungsi yang sama seperti train_output yang membedakan hanya datanya / inputan yang diproses berasal dari variabel test

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

- Kode Program:

```
1 # In [5]: import encoder and one hot
2 from sklearn.preprocessing import LabelEncoder
3 from sklearn.preprocessing import OneHotEncoder
```

Listing 7.26: Praktek5.py

Hasil 7.44 :

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

Figure 7.44: In 5 Aip

Baris 1: Import labelEncoder dari sklearn.processing digunakan untuk menormalkan label dimana label encoder hanya didefinisikan dengan nilai antara 0 dan n_classes-1.

Baris 2: Memasukkan modul / fungsi OneHotEncoder dari sklearn.processing yang digunakan untuk mendefinisikan fitur input dimana mengambil nilai dalam kisaran [0, maks (nilai)).

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

- Kode Program:

```

1 # In [6]: convert class names into one-hot encoding
2
3 # first, convert class names into integers
4 label_encoder = LabelEncoder()
5 integer_encoded = label_encoder.fit_transform(classes)

```

Listing 7.27: Praktek5.py

Hasil 7.45 :

```

In [5]: label_encoder = LabelEncoder()
...: integer_encoded = label_encoder.fit_transform(classes)

```

Figure 7.45: In 6 Aip

Baris 1: Membuat variabel label_encoder dengan penerapan modul / fungsi dari LabelEncoder tanpa parameter

Baris 2: Membuat variabel integer_encoded dengan penerapan fungsi label_encoder.fit_transform (ekstrasi fitur object) dari variabel classes dimana akan mengembalikan beberapa data yang diubah kembali dari variabel label_encoder.

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

- Kode Program:

```

1 # In [7]: then convert integers into one-hot encoding
2 onehot_encoder = OneHotEncoder(sparse=False)
3 integer_encoded = integer_encoded.reshape(len(integer_encoded), 1)
4 onehot_encoder.fit(integer_encoded)

```

Listing 7.28: Praktek7.py

Hasil 7.46 :

```

In [9]: onehot_encoder = OneHotEncoder(sparse=False)
...: integer_encoded = integer_encoded.reshape(len(integer_encoded), 1)
...: onehot_encoder.fit(integer_encoded)
E:\Anaconda3\lib\site-packages\sklearn\preprocessing\_encoders.py:371: FutureWarning: The handling
of integer data will change in version 0.22. Currently, the categories are determined based on the
range [0, max(values)], while in the future they will be determined based on the unique values.
If you want the future behaviour and silence this warning, you can specify "categories='auto'".
In case you used a LabelEncoder before this OneHotEncoder to convert the categories to integers,
then you can now use the OneHotEncoder directly.
warnings.warn(msg, FutureWarning)
Out[9]:
OneHotEncoder(categorical_features=None, categories=None,
               dtype=<class 'numpy.float64'>, handle_unknown='error',
               n_values=None, sparse=False)

```

Figure 7.46: In 7 Aip

Baris 1: Membuat variabel onehot_encoder yang memanggil fungsi OneHotEncoder tanpa mengembalikan matriks karena sparse=False.

Baris 2: Membuat variabel integer_encoded memanggil variabel integer_encoded pada kode program 6 untuk dieksekusi memberikan bentuk baru ke array tanpa mengubah datanya dari mengembalikan panjang nilai dari integer_encoded.

Baris 3: Onehotencoding melakukan fitting pada integer_encoded.

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

- Kode Program:

```
1 # In[8]: convert train and test output to one-hot
2 train_output_int = label_encoder.transform(train_output)
3 train_output = onehot_encoder.transform(train_output_int.
4     reshape(len(train_output_int), 1))
5 test_output_int = label_encoder.transform(test_output)
6 test_output = onehot_encoder.transform(test_output_int.reshape(
7     len(test_output_int), 1))
8 num_classes = len(label_encoder.classes_)
9 print("Number of classes: %d" % num_classes)
```

Listing 7.29: Praktek8.py

Hasil 7.47 :

```
In [10]: train_output_int = label_encoder.transform(train_output)
...: train_output = onehot_encoder.transform(train_output_int.reshape(len(train_output_int),
1))
...: test_output_int = label_encoder.transform(test_output)
...: test_output = onehot_encoder.transform(test_output_int.reshape(len(test_output_int), 1))
...
...: num_classes = len(label_encoder.classes_)
...: print("Number of classes: %d" % num_classes)
Number of classes: 369
```

Figure 7.47: In 8 Aip

Baris 1: Membuat variabel train_output_int yang mengeksekusi label_encoder dengan mengubah nilai dari parameter variabel train_output.

Baris 2: Membuat variabel train_output yang mengeksekusi variabel onehot_encoder dari kode program 7 dengan mengubah nilai dari variabel parameter train_output_int yang datanya sudah diubah kedalam bentuk array dan panjang nilai dari train_output_int telah dikembalikan.

Baris 3: Membuat variabel test_output_int yang mengeksekusi label_encoder dengan mengubah nilai dari parameter variabel test_output.

Baris 4: Membuat variabel test_output yang mengeksekusi variabel one-hot_encoder dari kode program 7 dengan mengubah nilai dari variabel parameter test_output_int yang datanya sudah diubah kedalam bentuk array dan panjang nilai dari test_output_int telah dikembalikan.

Baris 5: Membuat variabel num_classes untuk mengetahui jumlah class dari label_encoder

Baris 6: Perintah print digunakan untuk memunculkan hasil dari variabel num_classes

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

- Kode Program:

```
1 # In [9]: import sequential  
2  
3 from keras.models import Sequential  
4 from keras.layers import Dense, Dropout, Flatten  
5 from keras.layers import Conv2D, MaxPooling2D
```

Listing 7.30: Praktek9.py

Hasil 7.48 :

```
In [11]: from keras.models import Sequential  
...: from keras.layers import Dense, Dropout, Flatten  
...: from keras.layers import Conv2D, MaxPooling2D
```

Figure 7.48: In 9 Aip

Baris 1: Memanggil atau melakukan importing fungsi model sequential dari library keras.

Baris 2: Memanggil atau melakukan importing fungsi layer dense, dropout, dan flatten dari library keras.

Baris 3: Memanggil atau melakukan importing fungsi layer Conv2D dan MaxPooling2D dari library keras.

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

- Kode Program:

```
1 # In [10]: desain jaringan  
2 model = Sequential()  
3 model.add(Conv2D(32, kernel_size=(3, 3), activation='relu',  
4 input_shape=np.shape(train_input[0])))  
5 model.add(MaxPooling2D(pool_size=(2, 2)))
```

```

6 model.add(Conv2D(32, (3, 3), activation='relu'))
7 model.add(MaxPooling2D(pool_size=(2, 2)))
8 model.add(Flatten())
9 model.add(Dense(1024, activation='tanh'))
10 model.add(Dropout(0.5))
11 model.add(Dense(num_classes, activation='softmax'))
12
13 model.compile(loss='categorical_crossentropy', optimizer='adam',
14                 metrics=['accuracy'])
15
16 print(model.summary())

```

Listing 7.31: Praktek10.py

Hasil 7.49 :

```

Please use `rate` instead of `keep_prob`. Rate should be set to `rate = 1 - keep_prob'
-----  

Layer (type)          Output Shape         Param #  

=====  

conv2d_1 (Conv2D)     (None, 30, 30, 32)      896  

max_pooling2d_1 (MaxPooling2D) (None, 15, 15, 32)    0  

conv2d_2 (Conv2D)     (None, 13, 13, 32)      9248  

max_pooling2d_2 (MaxPooling2D) (None, 6, 6, 32)    0  

flatten_1 (Flatten)   (None, 1152)           0  

dense_1 (Dense)       (None, 1024)           1180672  

dropout_1 (Dropout)   (None, 1024)           0  

dense_2 (Dense)       (None, 369)            378225  

-----  

Total params: 1,569,041  

Trainable params: 1,569,041  

Non-trainable params: 0  

-----  

None

```

Figure 7.49: In 10 Aip

Baris 1: Melakukan pemodelan Sequential.

Baris 2: Menambahkan Konvolusi 2D dengan 32 filter konvolusi masing-masing berukuran 3x3 dengan algoritam activation relu dengan data dari train input mulai dari baris nol.

Baris 3: Menambahkan Max Pooling dengan matriks 2x2.

Baris 4: Penambahan Konvolusi 2D dengan 32 filter, konvolusi masing-masing berukuran 3x3 dengan algoritam activation relu.

Baris 5 Menambahkan Max Pooling dengan matriks 2x2.

Baris 6: Mendefinisikan inputan dengan 1024 neuron dan menggunakan algoritma tanh untuk activationnya.

Baris 7: Dropout terdiri dari pengaturan secara acak tingkat pecahan unit input ke 0 pada setiap pembaruan selama waktu pelatihan, yang membantu mencegah overfitting sebesar 50% .

Baris 8: Untuk output layer menggunakan data dari variabel num classes dengan fungsi activationnya softmax.

Baris 9: Konfigurasi proses pembelajaran, melalui metode compile, sebelum melatih suatu model.

Baris 10: Menampilkan model yang telah dibuat.

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

- Kode Program:

```
1 # In [11]: import sequential
2
3 import keras.callbacks
4 tensorboard = keras.callbacks.TensorBoard(log_dir='./logs/mnist-
style')
```

Listing 7.32: Praktek11.py

Hasil 7.50 :

```
In [13]: import keras.callbacks
...: tensorboard = keras.callbacks.TensorBoard(log_dir='./logs/mnist-style')
...:
```

Figure 7.50: In 11 Aip

Baris 1: Memasukkan / Mengimport library keras.callbacks dimana digunakan dalam penulisan log untuk TensorBoard, yang memungkinkan untuk memvisualisasikan grafik dinamis dari pelatihan dan metrik pengujian.

Baris 2: Membuat variabel tensorboard yang mendefinisikan fungsi TensorBoard pada keras.callbacks yang digunakan sebagai alat visualisasi yang disediakan dengan TensorFlow. Kemudian untuk fungsi log_dir (jalur direktori tempat menyimpan file log yang akan diuraikan oleh TensorBoard) memanggil data yaitu './logs/mnist-style'

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

- Kode Program:

```
1 # In [12]:
2 model.fit(train_input, train_output,
3            batch_size=32,
```

```

4     epochs=10,
5     verbose=2,
6     validation_split=0.2,
7     callbacks=[tensorboard])
8
9 score = model.evaluate(test_input, test_output, verbose=2)
10 print('Test loss:', score[0])
11 print('Test accuracy:', score[1])

```

Listing 7.33: Praktek12.py

Hasil 7.51 :

```

Train on 107668 samples, validate on 26918 samples
Epoch 1/10
- 1381s - loss: 1.5627 - acc: 0.6223 - val_loss: 1.0765 - val_acc: 0.7042
Epoch 2/10
- 708s - loss: 0.9834 - acc: 0.7260 - val_loss: 0.9293 - val_acc: 0.7370
Epoch 3/10
- 669s - loss: 0.8682 - acc: 0.7514 - val_loss: 0.8491 - val_acc: 0.7632
Epoch 4/10
- 773s - loss: 0.7955 - acc: 0.7670 - val_loss: 0.8378 - val_acc: 0.7642
Epoch 5/10
- 752s - loss: 0.7398 - acc: 0.7792 - val_loss: 0.8303 - val_acc: 0.7679
Epoch 6/10
- 770s - loss: 0.6993 - acc: 0.7880 - val_loss: 0.8385 - val_acc: 0.7659
Epoch 7/10
- 788s - loss: 0.6579 - acc: 0.7972 - val_loss: 0.8819 - val_acc: 0.7650
Epoch 8/10
- 764s - loss: 0.6317 - acc: 0.8020 - val_loss: 0.8602 - val_acc: 0.7664
Epoch 9/10
- 749s - loss: 0.5999 - acc: 0.8105 - val_loss: 0.8785 - val_acc: 0.7578
Epoch 10/10
- 742s - loss: 0.5863 - acc: 0.8136 - val_loss: 0.9032 - val_acc: 0.7573
Test loss: 0.9253497116118656
Test accuracy: 0.7576306951638724

```

Figure 7.51: In 12 Aip

Baris 1: Menerapkan fungsi `model.fit` yang didalamnya memproses `train_input`, `train_output`

Baris 2: Selanjutnya pada penerapan fungsi yang sama difungsikan `batch_size` (jumlah sampel per pembaharuan sampel dari data yang diolah) apabila `batch_size`nya tidak ditemukan maka otomatis akan dijadikan nilai 32

Baris 3: Pada penerapan fungsi yang sama, difungsikan `epochs` dimana perulangan dari berapa kali nilai yang digunakan untuk data, dan jumlahnya ialah 10

Baris 4: Mendefinisikan fungsi `verbose` dimana digunakan sebagai opsi untuk menghasilkan informasi logging dari data yang ditentukan dengan nilai 2

Baris 5: Mendefinisikan fungsi `validation_split` untuk memecah nilai dari perhitungan validasinya sebesar 0,2. (Fraksi data pelatihan untuk digunakan sebagai data validasi)

Baris 6: Mendefinisikan fungsi callbacks dengan parameternya yang mengeksekusi tensorboard dimana digunakan untuk visualisasikan parameter training, metrik, hiperparameter pada nilai/data yang diproses

Baris 7: Mendefinisikan variabel score dengan fungsi evaluate dari model yang ada dengan parameter test_input, tst_output dan verbose=2 dimana memprediksi output untuk input yang diberikan dan kemudian menghitung fungsi metrik yang ditentukan dalam modelnya

- Baris 8: Mencetak score optimasi dari test dengan ketentuan nilai parameter 0
- Baris 9: Mencetak score akurasi dari test dengan ketentuan nilai parameter 1

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

- Kode Program:

```
1 import time
2
3 results = []
4 for conv2d_count in [1, 2]:
5     for dense_size in [128, 256, 512, 1024, 2048]:
6         for dropout in [0.0, 0.25, 0.50, 0.75]:
7             model = Sequential()
8             for i in range(conv2d_count):
9                 if i == 0:
10                     model.add(Conv2D(32, kernel_size=(3, 3),
11 activation='relu', input_shape=np.shape(train_input[0])))
12                 else:
13                     model.add(Conv2D(32, kernel_size=(3, 3),
14 activation='relu'))
15                     model.add(MaxPooling2D(pool_size=(2, 2)))
16                     model.add(Flatten())
17                     model.add(Dense(dense_size, activation='tanh'))
18                     if dropout > 0.0:
19                         model.add(Dropout(dropout))
20                     model.add(Dense(num_classes, activation='softmax'))
21
22                     model.compile(loss='categorical_crossentropy',
23 optimizer='adam', metrics=['accuracy'])
24
25                     log_dir = './logs/conv2d_%d-dense_%d-dropout_%.2f'
26 % (conv2d_count, dense_size, dropout)
27                     tensorboard = keras.callbacks.TensorBoard(log_dir=
log_dir)
28
29                     start = time.time()
```

```

27         model.fit(train_input, train_output, batch_size=32,
28                     epochs=10,
29                     verbose=0, validation_split=0.2,
30                     callbacks=[tensorboard])
31                     score = model.evaluate(test_input, test_output,
32                     verbose=2)
33                     end = time.time()
34                     elapsed = end - start
35                     print("Conv2D count: %d, Dense size: %d, Dropout: %.2f - Loss: %.2f, Accuracy: %.2f, Time: %d sec" % (
36                     conv2d_count, dense_size, dropout, score[0], score[1],
37                     elapsed))
38                     results.append((conv2d_count, dense_size, dropout,
39                     score[0], score[1], elapsed))

```

Listing 7.34: Praktek13.py

Hasil 7.52 :

```

...:
...:     model.compile(loss='categorical_crossentropy', optimizer='adam',
...: metrics=['accuracy'])
...:
...:     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=[tensorboard])
...:     score = model.evaluate(test_input, test_output, verbose=2)
...:     end = time.time()
...:     elapsed = end - start
...:     print("Conv2D count: %d, Dense size: %d, Dropout: %.2f - Loss: %.2f, Accuracy: %.2f, Time: %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: 1540 sec
Conv2D count: 1, Dense size: 128, Dropout: 0.25 - Loss: 0.93, Accuracy: 0.76, Time: 1579 sec
Conv2D count: 1, Dense size: 128, Dropout: 0.50 - Loss: 0.81, Accuracy: 0.77, Time: 1599 sec
Conv2D count: 1, Dense size: 128, Dropout: 0.75 - Loss: 0.82, Accuracy: 0.77, Time: 1627

```

Figure 7.52: In 13 Aip

Baris 1: impor modul time dari python anaconda

Baris 2: Variabel result berisikan array kosong.

Baris 3: Menggunakan convolution 2D yang dimana akan memiliki 1 atau 2 layer.

Baris 4: Mendefinisikan dense_size dengan ukuran 128, 256, 512, 1024, 2048

Baris 5: Mendefinsikan drop_out dengan 0, 25%, 50%, dan 75%

Baris 6: Melakukan pemodelan Sequential

Baris 7: Jika ini adalah layer pertama, kita perlu memasukkan bentuk input.

Baris 8: Kalau tidak kita hanya akan menambahkan layer.

Baris 9: Kemudian, setelah menambahkan layer konvolusi, kita akan melakukan hal yang sama dengan max pooling.

Baris 10: Lalu, kita akan meratakan atau flatten dan menambahkan dense size ukuran apa pun yang berasal dari dense_size. Dimana akan selalu menggunakan algoritma tanh

Baris 11: 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

Baris 12: 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.

Baris 13: Atur direktori log yang berbeda untuk TensorBoard sehingga dapat membedakan konfigurasi yang berbeda.

Baris 14: Variabel start akan memanggil modul time atau waktu

Baris 15: Melakukan fit atau compile

Baris 16: Melakukan scoring dengan .evaluate yang akan menampilkan data loss dan accuracy dari model

Baris 17: end merupakan variabel untuk melihat waktu akhir pada saat pemodelan berhasil dilakukan.

Baris 18: Menampilkan hasil dari run skrip diatas

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

- Kode Program:

```
1
2 model = Sequential()
3 model.add(Conv2D(32, kernel_size=(3, 3), activation='relu',
4                 input_shape=np.shape(train_input[0])))
5 model.add(MaxPooling2D(pool_size=(2, 2)))
6 model.add(Conv2D(32, (3, 3), activation='relu'))
7 model.add(MaxPooling2D(pool_size=(2, 2)))
8 model.add(Flatten())
9 model.add(Dense(128, activation='tanh'))
```

```

9 model.add(Dropout(0.5))
10 model.add(Dense(num_classes, activation='softmax'))
11 model.compile(loss='categorical_crossentropy', optimizer='adam',
12 , metrics=['accuracy'])
12 print(model.summary())

```

Listing 7.35: Praktek14.py

Hasil 7.53 :

Layer (type)	Output Shape	Param #
conv2d_8 (Conv2D)	(None, 30, 30, 32)	896
max_pooling2d_8 (MaxPooling2D)	(None, 15, 15, 32)	0
conv2d_9 (Conv2D)	(None, 13, 13, 32)	9248
max_pooling2d_9 (MaxPooling2D)	(None, 6, 6, 32)	0
flatten_7 (Flatten)	(None, 1152)	0
dense_13 (Dense)	(None, 128)	147584
dropout_5 (Dropout)	(None, 128)	0
dense_14 (Dense)	(None, 369)	47601
<hr/>		
Total params: 205,329		
Trainable params: 205,329		
Non-trainable params: 0		
<hr/>		
None		

Figure 7.53: In 14 Aip

Baris 1: Melakukan pemodelan Sequential

Baris 2: Untuk layer pertama, Menambahkan Convolutio 2D dengan dmensi 32, dan ukuran matriks 3x3 dengan function aktivasi yang digunakan yaitu relu dan menampilkan input_shape

Baris 3: Dilakukan Max Pooling 2D dengan ukuran matriks 2x2

Baris 4:Untuk layer kedua, melakukan Convolusi lagi dengan kriteria yang sama tanpa menambahkan input, ini dilakukan untuk mendapatkan data yang terbaik

Baris 5: Flatten digunakan ntuk meratakan inputan

Baris 6: Menambahkan dense input sebanyak 128 neuron dengan menggunakan function aktivasi tanh.

Baris 7: Dropout sebanyak 50% untuk menghindari overfitting

Baris 8: Menambahkan dense pada model untuk output dimana layer ini akan menjadi jumlah dari class yang ada.

Baris 9: Mengcompile model yang didefinisikan diatas

Baris 10: Menampilkan ringkasan dari pemodelan yang dilakukan

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

- Kode Program:

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

Listing 7.36: Praktek15.py

Hasil 7.54 :

```
In [26]: model.fit(np.concatenate((train_input, test_input)),  
...:             np.concatenate((train_output, test_output)),  
...:             batch_size=32, epochs=10, verbose=2)  
Epoch 1/10  
- 247s - loss: 1.7949 - acc: 0.5843  
Epoch 2/10  
- 236s - loss: 1.0797 - acc: 0.7064  
Epoch 3/10  
- 235s - loss: 0.9648 - acc: 0.7308  
Epoch 4/10  
- 237s - loss: 0.9060 - acc: 0.7434  
Epoch 5/10  
- 237s - loss: 0.8671 - acc: 0.7519  
Epoch 6/10  
- 236s - loss: 0.8362 - acc: 0.7573  
Epoch 7/10  
- 238s - loss: 0.8137 - acc: 0.7631  
Epoch 8/10  
- 239s - loss: 0.7980 - acc: 0.7652  
Epoch 9/10  
- 239s - loss: 0.7831 - acc: 0.7692  
Epoch 10/10  
- 239s - loss: 0.7695 - acc: 0.7724  
Out[26]: <keras.callbacks.History at 0x14c1941f5f8>
```

Figure 7.54: In 15 Aip

Baris 1: Melakukan fit dengan join data train dan test agar dapat dilakukan pelatihan untuk jaringan pada semua data yang dimiliki.

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

- Kode Program:

```
1  
2 model.save("mathsymbols.model")
```

Listing 7.37: Praktek1.py

Hasil 7.55 :

```
In [18]: model.save("mathsymbols.model")
```

Figure 7.55: In 16 Aip

Baris 1: Menyimpan atau save model yang telah di latih dengan nama mathsymbols.model

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

- Kode Program:

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

Listing 7.38: Praktek1.py

Hasil 7.56 :

```
In [19]: np.save('classes.npy', label_encoder.classes_)
```

Figure 7.56: In 17 Aip

Baris 1: Simpan label enkoder (untuk membalikkan one-hot encoder) dengan nama classes.npy

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

- Kode Program:

```
1 # In[18]: load the pre-trained model and predict the math symbol
  for an arbitrary image;
2 # the code below could be placed in a separate file
3
4 import keras.models
5 model2 = keras.models.load_model("mathsymbols.model")
6 print(model2.summary())
```

Listing 7.39: Praktek18.py

Hasil 7.57 :

Baris 1: Impor models dari librari Keras

Baris 2: Variabel model2 akan memanggil model yang telah disave tadi

Baris 3: Menampilkan ringkasan dari hasil pemodelan

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

- Kode Program:

```
1 # In[19]: restore the class name to integer encoder
2 label_encoder2 = LabelEncoder()
3 label_encoder2.classes_ = np.load('classes.npy')
4
5 def predict(img_path):
6     newimg = keras.preprocessing.image.img_to_array(pil_image.
7         open(img_path))
8     newimg /= 255.0
```

```

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

```

Layer (type)	Output Shape	Param #
conv2d_8 (Conv2D)	(None, 30, 30, 32)	896
max_pooling2d_8 (MaxPooling2D)	(None, 15, 15, 32)	0
conv2d_9 (Conv2D)	(None, 13, 13, 32)	9248
max_pooling2d_9 (MaxPooling2D)	(None, 6, 6, 32)	0
flatten_7 (Flatten)	(None, 1152)	0
dense_13 (Dense)	(None, 128)	147584
dropout_5 (Dropout)	(None, 128)	0
dense_14 (Dense)	(None, 369)	47601

```

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

```

```

None

```

Figure 7.57: In 18 Aip

```

9   # do the prediction
10  prediction = model2.predict(newimg.reshape(1, 32, 32, 3))
11
12  # figure out which output neuron had the highest score , and
13  # reverse the one-hot encoding
14  inverted = label_encoder2.inverse_transform([np.argmax(
    prediction)]) # argmax finds highest-scoring output
    print("Prediction: %s, confidence: %.2f" % (inverted[0], np
        .max(prediction)))

```

Listing 7.40: Praktek19.py

Hasil 7.58 :

```

In [21]: 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.pred

```

Figure 7.58: In 19 Aip

- Menampilkan hasil dari variabel prediction dan inverted

Baris 1: Memanggil fungsi LabelEncoder

Baris 2: Variabel label_encoder akan memanggil class yang disave sebelumnya.

Baris 3: Function Predict akan mengubah gambar kedalam bentuk array

Baris 4: Variabel prediction akan melakukan prediksi untuk model2 dengan reshape variabel newimg dengan bentukarray 4D.

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

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

- Kode Program:

```
1 # In [20]: grab an image (we'll just use a random training image
2     for demonstration purposes)
3 predict("HASYv2/hasy-data/v2-00010.png")
4
5 predict("HASYv2/hasy-data/v2-00500.png")
6 predict("HASYv2/hasy-data/v2-00700.png")
```

Listing 7.41: Praktek20.py

Hasil 7.59 :

```
In [31]: predict("HASYv2/hasy-data/v2-00010.png")
...:
...: predict("HASYv2/hasy-data/v2-00500.png")
...:
...: predict("HASYv2/hasy-data/v2-00700.png")
Prediction: A, confidence: 0.86
Prediction: \prod, confidence: 0.53
Prediction: \alpha, confidence: 0.94
```

Figure 7.59: In 20 Aip

Baris 1: Melakukan prediksi dari pelatihan dari gambar v2-00010.png

Baris 2: Melakukan prediksi dari pelatihan dari gambar v2-00500.png

Baris 3: Melakukan prediksi dari pelatihan dari gambar v2-00700.png

7.2.3 Penanganan Error

```
1
2 File "<ipython-input-2-dd892d606c29>", line 2, in <module>
3     random.shuffle(imgs)
4
5 NameError: name 'imgs' is not defined
```

Listing 7.42: ErorChapter7aip.py

(a) Skrinsut Error

(b) Kode Error dan Jenis Errornya

Kode Error: "Name Error" name 'imgs' is not defined

Jenis Error: Images tidak terdefinisi

```
In [2]: import random
....: random.shuffle(imgs)
....: split_idx = int(0.8*len(imgs))
....: train = imgs[:split_idx]
....: test = imgs[split_idx:]
Traceback (most recent call last):

File "<ipython-input-2-dd892d606c29>", line 2, in <module>
    random.shuffle(imgs)

NameError: name 'imgs' is not defined
```

Figure 7.60: Error Aip

(c) Penanganan

Menentukan atau memebuka file explorer dari file yang ditempatkan atau disesuaikan dengan letak filenya.

Chapter 8

Discussion

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

Chapter 9

Discussion

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

Chapter 10

Discussion

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

Chapter 11

Discussion

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

Chapter 12

Discussion

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

Chapter 13

Discussion

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

Chapter 14

Discussion

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

Appendix A

Form Penilaian Jurnal

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

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

Figure A.1: Form nilai bagian 1.

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

Figure A.2: form nilai bagian 2.

Appendix B

FAQ

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

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

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

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

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

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

M : Pa saya bingung D : Coba lihat standar alasan

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

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

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

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

1. Tidak Mengerti : anda boleh menggunakan alasan ini jika anda sudah melakukan tahapan membaca dan meresumekan 15 jurnal. Sudah mencoba dan mempraktekkan teorinya dengan mencari di youtube dan google minimal 6 jam sehari selama 3 hari berturut-turut.
2. Bingung : anda boleh mengatakan alasan bingung setelah maksimal dalam berusaha menyelesaikan tugas bimbingan dari dosen(sudah dilakukan semua). Anda belum bisa mengatakan alasan bingung jika anda masih belum menyelesaikan tugas bimbingan dan poin nomor 1 diatas. Setelah anda menyelesaikan tugas bimbingan secara maksimal dan tahap 1 poin diatas, tapi anda masih tetap bingung maka anda boleh memakai alasan ini.

Bibliography

- [1] Cahyo Darujati and Agustinus Bimo Gumelar. Pemanfaatan teknik supervised untuk klasifikasi teks bahasa indonesia. *Jurnal Bandung Text Mining*, 16(1):5–1, 2012.
- [2] Joshua Eckroth. *Python Artificial Intelligence Projects for Beginners: Get up and running with Artificial Intelligence using 8 smart and exciting AI applications*. Packt Publishing Ltd, 2018.
- [3] Helfi Nasution. Implementasi logika fuzzy pada sistem kecerdasan buatan. *ELKHA*, 4(2), 2012.
- [4] Stuart J Russell and Peter Norvig. *Artificial intelligence: a modern approach*. Malaysia; Pearson Education Limited,, 2016.