

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 Fadila/1164072	2
1.4.1 Teori	2
1.4.2 Instalasi	6
1.4.3 Penanganan Error	19
1.5 Rahmi Roza/1164085	20
1.5.1 Teori	20
1.5.2 Instalasi	24
1.5.3 Penanganan Error	26
1.6 Lusia Violita Aprilian/1164080	27
1.6.1 Artificial Intelligence	27
1.6.2 Supervised Learning dan Data	30
1.6.3 Learning and Predicting	31
1.6.4 Model Persistence	32
1.6.5 Conventions	32
1.6.6 Penanganan Error	33
2 Membangun Model Prediksi	35
2.1 Fadila/1164072	35
2.1.1 Teori	35
2.1.2 Praktek Scikit-Learn	43
2.1.3 Penanganan Error	49
2.2 Lusia Violita Aprilian	51
2.2.1 Teori	51
2.2.2 scikit-learn	57

2.2.3	Penanganan Error	62
2.3	Rahmi Roza/1164085	62
2.3.1	Teori	62
2.3.2	Scikit-learn	69
2.3.3	Penanganan Error	74
3	Prediksi dengan Random Forest	76
3.1	Lusia Violita Aprilian/1164080	76
3.1.1	Teori	76
3.1.2	Praktek	80
3.1.3	Penanganan Error	88
3.2	Rahmi Roza/1164085	89
3.2.1	Teori	89
3.2.2	Praktek	94
3.3	Fadila/1164072	102
3.3.1	Teori	102
3.3.2	Praktek	108
3.3.3	Penanganan Error	121
4	Klasifikasi Teks	127
4.1	Lusia Violita Aprilian/1164080	127
4.1.1	Teori	127
4.1.2	Praktek	130
4.1.3	Penanganan Error	133
4.2	Rahmi Roza-1164085	134
4.2.1	Teori	134
4.2.2	Praktek	137
4.2.3	Penanganan Error	141
4.3	Fadila-1164072	143
4.3.1	Teori	143
4.3.2	Praktek Fadila	147
4.3.3	Penanganan Error Fadila	153
4.3.4	Praktek	154

5 Vektorisasi Kata dan Dokumen	155
5.1 Lusia Violita Aprilian-1164080	155
5.1.1 Teori	155
5.1.2 Praktek Program	160
5.1.3 Penanganan Error	170
5.2 Rahmi Roza-1164085	171
5.2.1 Teori	171
5.2.2 Praktek	174
5.2.3 Penanganan Error	185
5.3 Fadila-1164072	186
5.3.1 Teori	186
5.3.2 Praktek	190
6 MFCC dan Neutral Network	203
6.1 Rahmi Roza-1164085	203
6.1.1 Teori	203
6.1.2 Praktek	208
6.1.3 Penanganan Error	218
6.2 Fadila-1164072	218
6.2.1 Teori	218
6.2.2 Praktek	225
6.2.3 Fadila-Penanganan Error Chapter 6	235
6.3 Lusia Violita Aprilian-1164080	236
6.3.1 Teori	236
6.3.2 Praktek	240
6.3.3 Penanganan Error	250
6.3.4 Cek plagiarisme	251
7 NCC	252
7.1 Lusia Violita Aprilian-1184080	252
7.1.1 Teori	252
7.1.2 Praktek	260
7.1.3 Penanganan Error	277
7.1.4 Cek Plagiarisme	278
7.2 Rahmi Roza-1164085	278
7.2.1 Teori	278
7.2.2 Praktek	285

7.2.3	Penanganan Error	302
7.3	Fadila-1164072	303
7.3.1	Teori	303
7.3.2	Praktek	311
7.3.3	Penanganan Error	330
8	Discussion	333
9	Discussion	334
10	Discussion	335
11	Discussion	336
12	Discussion	337
13	Discussion	338
14	Discussion	339
A	Form Penilaian Jurnal	340
B	FAQ	343
	Bibliography	345

List of Figures

1.1	error model persistence	3
1.2	capturing	4
1.3	penanganan error model persistence	6
1.4	install anaconda 1	6
1.5	penanganan error model persistence	7
1.6	install anaconda 2	7
1.7	Pengecekan Anaconda	8
1.8	instalasi pip scikit-learn	8
1.9	instalasi conda scikit-learn	9
1.10	uji coba codingan	9
1.11	pengujian loading an example dataset	10
1.12	pengujian loading an example dataset	10
1.13	hasil print uji cobat	10
1.14	pengujian learning dan predicting	11
1.15	pengujian model persistence 1	12
1.16	pengujian model persistence 2	12
1.17	pengujian type casting 1	14
1.18	pengujian type casting 2	14
1.19	pengujian refitting and updating	14
1.20	pengujian multiclass and multiable 1	15
1.21	pengujian multiclass and multiable 2	15
1.22	error model persistence	19
1.23	penanganan error model persistence	20
1.24	penanganan error model persistence	20
1.25	gambar1	24
1.26	gambar2	24
1.27	gambar3	25
1.28	gambar4	25

1.29 gambar5	26
1.30 gambar6	27
1.31 gambar7	27
1.32 Learning and predicting 1	31
1.33 Learning and predicting 2	31
1.34 Learning and predicting 3	32
1.35 Model Persistence	32
1.36 Conventions	33
1.37 skrinsut error	33
1.38 gb 1	34
2.1 binary classification	35
2.2 supervised	36
2.3 unsupervised	36
2.4 clustering	37
2.5 Evaluasi	37
2.6 Akurasi	38
2.7 Contoh Evaluasi Dan Akurasi Secara Bersamaan	38
2.8 confusion matrix	39
2.9 recall	39
2.10 precision	40
2.11 f-measure	40
2.12 k-fold classification 1	41
2.13 k-fold classification 2	41
2.14 decision tree	41
2.15 informaion gain 1	42
2.16 information gain 2	42
2.17 entropi	43
2.18 codingan pertama	44
2.19 codingan kedua	44
2.20 codingan ketiga	45
2.21 codingan keempat	45
2.22 codingan kelima	46
2.23 codingan keenam	46
2.24 codingan ketujuh	46
2.25 codingan kedelapan	47

2.26 codingan kesembilan	47
2.27 codingan ke-10	48
2.28 codingan ke-11	48
2.29 codingan ke-12	49
2.30 error 1	49
2.31 error 2	50
2.32 error 3	50
2.33 error 4	51
2.34 Binary Classification	52
2.35 Supervised Learning	52
2.36 Unsupervised Learning	53
2.37 Cluster	54
2.38 Evaluasi dan Akurasi	54
2.39 K-fold cross validation	55
2.40 Decision Tree	56
2.41 Information gain	56
2.42 Entropi	56
2.43 load dataset	57
2.44 generate binary label	57
2.45 use one-hot encoding on categorical columns	57
2.46 shuffle rows, split training dan testing data	58
2.47 fit a decision tree	58
2.48 visualize tree	59
2.49 visualize tree	59
2.50 save tree	59
2.51 t.score	59
2.52 show average score	60
2.53 for max depth in range	60
2.54 depth acc	61
2.55 matplotlib	61
2.56 error	62
2.57 penyelesaian	62
2.58 binary classification	63
2.59 supervised	63
2.60 unsupervised	64
2.61 clustering	64

2.62 Evaluasi	65
2.63 Akurasi	65
2.64 Contoh Evaluasi Dan Akurasi Secara Bersamaan	66
2.65 confusion matrix	67
2.66 k-fold classification 1	68
2.67 decision tree	69
2.68 informaion gain 1	69
2.69 entropi	70
2.70 Gambar pertama	70
2.71 Gambar kedua	70
2.72 Gambar Ketiga	71
2.73 Gambar Keempat	71
2.74 Gambar Kelima	71
2.75 Gambar Keenam	72
2.76 Gambar Ketujuh	72
2.77 Gambar Kedelapan	72
2.78 Gambar Kesembilan	73
2.79 Gambar Kesepuluh	73
2.80 Gambar Kesebelas	74
2.81 Gambar Keduabelas	74
2.82 Gambar Ketigabelas	75
3.1 Random Forest	76
3.2 Kode Python	77
3.3 Pesan Window Console	77
3.4 Dataset	77
3.5 Hasil Dataset Cell	77
3.6 Perintah	77
3.7 Tabel Confusion Matriks	78
3.8 Rumus Confusion Matriks	79
3.9 Confusion Matriks	79
3.10 Voting Pada Random Forest	79
3.11 Aplikasi Pandas	80
3.12 Hasil Pandas	80
3.13 Aplikasi Numpy	80
3.14 Hasil Numpy	80

3.15 Aplikasi Matplotlib	81
3.16 Hasil Matplotlib	81
3.17 Membaca Data File	82
3.18 Melihat Data Sebagian	82
3.19 Melihat Jumlah Data	82
3.20 Mengubah menjadi kolom	82
3.21 Lihat sebagian data awal	82
3.22 Melihat jumlah data	83
3.23 Mengelompokkan burung	83
3.24 Melakukan pivot	83
3.25 Melihat data awal imgid	83
3.26 Melihat jumlah data imgid	83
3.27 Data ciri label dari join	84
3.28 Mengubah menjadi kolom	84
3.29 Melihat isi data frame	84
3.30 Membagi data	84
3.31 Kelas Random Forest	84
3.32 Membangun Random forest	85
3.33 Melihat hasil	85
3.34 Lihat hasil score	85
3.35 Memetakan ke confusion matrix	85
3.36 Melihat hasil	85
3.37 Melakukan Plot	86
3.38 Plotting nama data	86
3.39 Melakukan perintah plot	86
3.40 Klasifikasi menggunakan decision tree	86
3.41 Klasifikasi menggunakan SVM	87
3.42 Pengecekan cross validation random forest	87
3.43 Pengecekan cross validation decision tree	87
3.44 Pengecekan cross validation SVM	87
3.45 Pengamatan Komponen	88
3.46 Plot informasi	88
3.47 Skrinsut Error	88
3.48 Penyelesaian	89
3.49 Hasil	89
3.50 Random Forest	90

3.51 (b)	90
3.52 (c)	90
3.53 (d)	91
3.54 (e)	91
3.55 (h)	91
3.56 Confussion Matrik	93
3.57 Voting Random forest	94
3.58 Pandas	95
3.59 Numpy	95
3.60 Matplotlib	96
3.61 Gambar1	97
3.62 Gambar2	97
3.63 Gambar3	98
3.64 Gambar 4	98
3.65 Gambar 5	99
3.66 Gambar 6	99
3.67 Gambar 7	100
3.68 Gambar 8	100
3.69 Gambar 9	101
3.70 Gambar 10	101
3.71 Gambar 11	102
3.72 Gambar 12	102
3.73 Gambar 13	103
3.74 Gambar 14	103
3.75 Gambar 15	104
3.76 Gambar 16	104
3.77 Gambar 17	105
3.78 Gambar 18	105
3.79 Gambar 19	106
3.80 Gambar 20	106
3.81 Gambar 21	107
3.82 Gambar 22	108
3.83 Gambar 23	108
3.84 SVM	109
3.85 Decission Tree	109
3.86 Cross Validation 1	110

3.87	Cross Validation 2	110
3.88	Cross Validation 3	111
3.89	Program Pengamatan Komponen Informasi 1	112
3.90	Program Pengamatan Komponen Informasi 2	113
3.91	Error	114
3.92	random forest	114
3.93	random forest 1	115
3.94	random forest 2	115
3.95	random forest 3	116
3.96	random forest 4	116
3.97	cross validation	117
3.98	confusion matrix	117
3.99	voting random forest	118
3.100	pandas	118
3.101	numpy	119
3.102	matplotlib	119
3.103	random forest	120
3.104	random forest 2	120
3.105	random forest 3	121
3.106	random forest 4	121
3.107	random forest 5	122
3.108	random forest 6	122
3.109	random forest 7	123
3.110	random forest 8	123
3.111	random forest 9	124
3.112	random forest 10	124
3.113	random forest 11	124
3.114	random forest 12	124
3.115	random forest 13	124
3.116	random forest 14	124
3.117	random forest 15	124
3.118	random forest 16	124
3.119	random forest 17	124
3.120	random forest 18	124
3.121	confusion matrix 1	124
3.122	confusion matrix 2	124

3.123confusion matrix 3	125
3.124confusion matrix 4	125
3.125confusion matrix 5	125
3.126svm	125
3.127decision tree	125
3.128cross validation 1	125
3.129cross validation 2	125
3.130cross validation 3	125
3.131pengamatan komponen informasi 1	125
3.132pengamatan komponen informasi 2	126
3.133error 1	126
3.134error 2	126
3.135error 3	126
4.1 Lusia-Klasifikasi teks	127
4.2 Lusia-Klasifikasi bunga	128
4.3 Lusia-Teknik YouTube	128
4.4 Lusia-Bag of Word	129
4.5 Lusia-TF IDF	129
4.6 Lusia-Pandas	130
4.7 Lusia-Hasil Pandas	130
4.8 Lusia-Pecah data	130
4.9 Lusia-Hasil Pecah data	131
4.10 Lusia-Vektorisasi dan klasifikasi	131
4.11 Lusia-Decission Tree Katty Perry	131
4.12 Lusia-Hasil klasifikasi SVM	132
4.13 Lusia-Decission Tree	132
4.14 Lusia-ploting confusion matrix	132
4.15 Lusia-Program cross validation	133
4.16 Lusia-Program pengamatan komponen informasi	133
4.17 Lusia-skrinsut error	134
4.18 Klasifikasi Teks Roza	134
4.19 Klasifikasi Bunga Roza	135
4.20 Youtube Roza	136
4.21 Bag of Words Roza	137
4.22 TF-ID Rroza	137

4.23 Nomor 1 Roza	138
4.24 Hasil 1 Roza	138
4.25 Hasil 1 Roza	138
4.26 Nomor 2 Roza	138
4.27 Hasil 2 Roza	139
4.28 Nomor 3 Roza	139
4.29 Hasil 3 Roza	139
4.30 Nomor 4 Roza	139
4.31 Nomor 5 Roza	140
4.32 Nomor 6 Roza	140
4.33 Nomor 7 Roza	141
4.34 Hasil 7 Roza	141
4.35 Hasil 7 Roza	141
4.36 Nomor 8 Roza	142
4.37 Nomor 8 Roza	142
4.38 Error Roza	143
4.39 text-fadila	143
4.40 bunga-fadila	144
4.41 youtube-fadila	145
4.42 bag-fadila	146
4.43 tf2-fadila	147
4.44 1-fadila	148
4.45 lanjutan1-fadila	148
4.46 lanjutan1-1-fadila	148
4.47 lanjutan1-2-fadila	148
4.48 2-fadila	149
4.49 lanjutan2-fadila	149
4.50 3-fadila	150
4.51 4-fadila	150
4.52 5-fadila	151
4.53 cod6-fadila	151
4.54 6-fadila	152
4.55 lanjutan6-fadila	152
4.56 7-fadila	152
4.57 lanjutan7-fadila	152
4.58 8-fadila	153

4.59	lanjutan8-fadila	153
4.60	error1-fadila	154
5.1	Lusia-Vektorisasi	156
5.2	Lusia-Kesamaan Kucing Anjing	156
5.3	Lusia-Kesamaan Kucing Spatula	156
5.4	Lusia-Konsep Vektorisasi Kata	157
5.5	Lusia-Vektorisasi Dokumen	158
5.6	Lusia-Mean	159
5.7	Lusia-Standar Deviasi	159
5.8	Lusia-Skip gram	160
5.9	Lusia-Mencoba dataset	160
5.10	Lusia-Menjelaskan Vektor Love	161
5.11	Lusia-Menjelaskan Vektor Faith	161
5.12	Lusia-Menjelaskan Vektor Fall	161
5.13	Lusia-Menjelaskan Vektor Sick	161
5.14	Lusia-Menjelaskan Vektor Clear	162
5.15	Lusia-Menjelaskan Vektor Shine	162
5.16	Lusia-Menjelaskan Vektor Bag	162
5.17	Lusia-Menjelaskan Vektor Car	162
5.18	Lusia-Menjelaskan Vektor Wash	163
5.19	Lusia-Menjelaskan Vektor Motor	163
5.20	Lusia-Menjelaskan Vektor Cycle	163
5.21	Lusia-Perbandingan	164
5.22	Lusia-extract words	165
5.23	Lusia-Permute Sentences	165
5.24	Lusia-Hasil Permute Sentences	166
5.25	Lusia-TaggedDocument dan Doc2Vec	166
5.26	Lusia-menambahkan data training	167
5.27	Lusia-Pengocokan dan Pembersihan Data	168
5.28	Lusia-Save	168
5.29	Lusia-Delete Temporary	169
5.30	Lusia-Infer code	169
5.31	Lusia-cosine similarity	169
5.32	Lusia-score	170
5.33	Lusia-Error	170

5.34 Lusia-Solusi	171
5.35 Vektorisasi Kata Roza	171
5.36 Google Dataset Roza	172
5.37 Vektorisasi Kata Roza	173
5.38 Vektorisasi Dokumen Roza	173
5.39 Mean Roza	174
5.40 Standar Deviasi Roza	174
5.41 Skip-Gram Roza	174
5.42 Plagiarisme Roza	174
5.43 Love Roza	175
5.44 Faith Roza	175
5.45 Fall Roza	176
5.46 Sick Roza	176
5.47 Clear Roza	177
5.48 Shine Roza	177
5.49 Bag Roza	178
5.50 Car Roza	178
5.51 Wash Roza	179
5.52 Motor Roza	179
5.53 Cycle Roza	180
5.54 Perbandingan Similariti Roza	180
5.55 Extract Word Roza	181
5.56 PermuseSentences Roza	181
5.57 No 3 Roza	182
5.58 No 4 Roza	182
5.59 No 5 Roza	183
5.60 No 5 Roza	183
5.61 No 6 Roza	184
5.62 No 6 Roza	184
5.63 No 7 Roza	184
5.64 No 8 Roza	185
5.65 No 9 Roza	185
5.66 Error Roza	185
5.67 Vektorisasi Kata-fadila	186
5.68 Dimensi Vektor Dataset Wikipedia-fadila	187
5.69 Dimensi Vektor Dataset -fadila	187

5.70	Vektorisasi Untuk Kata-fadila	188
5.71	Vektorisasi Untuk Dokumen-fadila	189
5.72	Mean-fadila	189
5.73	Mean Lanjutan-fadila	189
5.74	Standar Devisiasi-fadila	189
5.75	Skip Gram - fadila	190
5.76	Plagiarisme - fadila	190
5.77	Google Dataset - Love - fadila	191
5.78	Google Dataset - Faith- fadila	191
5.79	Google Dataset - Fall - fadila	191
5.80	Google Dataset - Sick - fadila	192
5.81	Google Dataset - Clear - fadila	192
5.82	Google Dataset - Shine - fadila	193
5.83	Google Dataset - Bag - fadila	193
5.84	Google Dataset - Car - fadila	193
5.85	Google Dataset - Wash - fadila	194
5.86	Google Dataset - Motor - fadila	194
5.87	Google Dataset - Cycle - fadila	194
5.88	Google Dataset - Similarity - fadila	195
5.89	1- Extract-Word - fadila	195
5.90	2- Extract-Word - fadila	196
5.91	PermuteSentences - fadila	196
5.92	3-tagged - fadila	197
5.93	4-model-1- fadila	197
5.94	4-model-2- fadila	198
5.95	4-model-3- fadila	198
5.96	5- Pengocokan Dan Pembersihan Data- fadila	198
5.97	6- Temporari Train Hapus-fadila	199
5.98	6- Model Disave-fadila	199
5.99	7-infercode-fadila	200
5.1008	8-cosine-fadila	200
5.1019	9-score-fadila	200
5.102	10error-error-fadila	201
5.103	10error-error-2-fadila	202
6.1	MFCC Roza	203

6.2	Konsep Dasar Neural Network Roza	204
6.3	Konsep Pembobotan Roza	205
6.4	Konsep Aktivasi Roza	205
6.5	Cara Membaca Hasil Plot MLCC Roza	206
6.6	One Hot Encoding Roza	206
6.7	NP Unique Roza	207
6.8	To Categorical roza	207
6.9	Sequential Roza Roza	208
6.10	Plagiarisme Roza	208
6.11	Hasil No 1 Roza	208
6.12	Kode Program No 2 Roza	209
6.13	Hasil No 2 Roza	209
6.14	Hasil No 3 Roza	210
6.15	Hasil No 4 Roza	211
6.16	Hasil No 5 Roza	213
6.17	Hasil No 6 Roza	214
6.18	Hasil No 7 Roza	215
6.19	Hasil No 8 Roza	215
6.20	Hasil No 9 Roza	216
6.21	Hasil No 10 Roza	217
6.22	Hasil No 11 Roza	217
6.23	Error Roza	218
6.24	Suara Harus Dilakukan MFCC - fadila	219
6.25	Konsep Dasar Neural Network- fadila	219
6.26	Pembobotan Pada Neural Network- fadila	220
6.27	Fungsi Aktifasi - fadila	221
6.28	Pembacaan Hasil Plot MFCC- fadila	222
6.29	Pembacaan Hasil Plot MFCC- fadila	222
6.30	One-Hot Encoding 1- fadila	222
6.31	One-Hot Encoding 1- fadila	222
6.32	One-Hot Encoding 2 - fadila	223
6.33	Np.Unique - fadila	224
6.34	To Categorical - fadila	224
6.35	Sequential - fadila	225
6.36	Plagiarisme- fadila	225
6.37	Contoh GTZAN Genre Collection - fadila	226

6.38	Display Mfcc 1 - fadila	227
6.39	Display Mfcc 2 - fadila	227
6.40	Extract Feature Song - fadila	228
6.41	Generate_Features_And_Labels - fadila	230
6.42	Generate_Features_Labels-2-fadila	231
6.43	Pemisahan Data Training Dan Dataset - fadila	232
6.44	Pemisahan Data Training Dan Dataset 2 - fadila	232
6.45	Fungsi Sequential - fadila	233
6.46	Fungsi Compile - fadila	234
6.47	Fungsi Fit - fadila	234
6.48	Fungsi Evaluate - fadila	235
6.49	Fungsi Predict - fadila	235
6.50	Error - fadila	235
6.51	Lusia-MFCC	236
6.52	Lusia-Neural Network	237
6.53	Lusia-Konsep Pembobotan	238
6.54	Lusia-Konsep Fungsi Aktivasi	238
6.55	Lusia-Membaca hasil plot	239
6.56	Lusia-one hot encoding	239
6.57	Lusia-np.unique	240
6.58	Lusia-to categorical	240
6.59	Lusia-sequential	240
6.60	Lusia-Hasil Load	241
6.61	Lusia-Hasil display MFCC	241
6.62	Lusia-Hasil display	242
6.63	Lusia-Hasil extract feature	243
6.64	Lusia-Hasil generate fefure and label	244
6.65	Lusia-Hasil generate fefure and label	245
6.66	Lusia-Hasil Pemisahan	246
6.67	Lusia-Hasil Sequential	246
6.68	Lusia-Hasil Compile	247
6.69	Lusia-Hasil Fungsi Fit	248
6.70	Lusia-Hasil Fungsi Evaluate	249
6.71	Lusia-Hasil Fungsi Predic	249
6.72	Lusia-skrinsut error 1	250
6.73	Lusia-skrinsut error 2	250

6.74 Lusia-Cek Plagiarisme	251
7.1 Lusia-Tokenizer	252
7.2 Lusia-StratifiedKFold	253
7.3 Lusia-for train, test in splits	253
7.4 Lusia-Hasil Bagian 1	253
7.5 Lusia-Ilustrasi Kode Program No.4	254
7.6 Lusia-Ilustrasi Maksud fungsi No.5	254
7.7 Lusia-Ilustrasi Maksud fungsi No.6	255
7.8 Lusia-Ilustrasi Maksud fungsi No.7	255
7.9 Lusia-Ilustrasi Maksud fungsi No.8	256
7.10 Lusia-Ilustrasi Maksud fungsi No.9	256
7.11 Lusia-Ilustrasi Maksud fungsi No.10	257
7.12 Lusia-Ilustrasi Gambar	257
7.13 Lusia-Ilustrasi konvolusi 1	258
7.14 Lusia-Ilustrasi konvolusi 2	258
7.15 Lusia-Ilustrasi konvolusi 3	258
7.16 Lusia-Ilustrasi konvolusi 4	259
7.17 Lusia-Ilustrasi konvolusi 5	259
7.18 Lusia-Ilustrasi konvolusi 6	259
7.19 Lusia-Ilustrasi konvolusi baris ketiga	259
7.20 Lusia-Hasil Kode Program 1	260
7.21 Lusia-Hasil Kode Program 2	262
7.22 Lusia-Hasil Kode Program 3	262
7.23 Lusia-Hasil Kode Program 4	264
7.24 Lusia-Hasil Kode Program 5	264
7.25 Lusia-Hasil Kode Program 6	265
7.26 Lusia-Hasil Kode Program 7	266
7.27 Lusia-Hasil Kode Program 8	267
7.28 Lusia-Hasil Kode Program 9	267
7.29 Lusia-Hasil Kode Program 10	269
7.30 Lusia-Hasil Kode Program 11	269
7.31 Lusia-Hasil Kode Program 12	271
7.32 Lusia-Hasil Kode Program 13	272
7.33 Lusia-Hasil Kode Program 14	273
7.34 Lusia-Hasil Kode Program 15	273

7.35 Lusia-Hasil Kode Program 16	274
7.36 Lusia-Hasil Kode Program 17	274
7.37 Lusia-Hasil Kode Program 18	275
7.38 Lusia-Hasil Kode Program 19	276
7.39 Lusia-Hasil Kode Program 20	277
7.40 Lusia-skrinsut error 1	277
7.41 Lusia-skrinsut error 2	277
7.42 Lusia-Plagiarisme	278
7.43 Tokenizer Roza	279
7.44 K-Fold Cross Validation Roza	279
7.45 No 3 Roza	280
7.46 No 5 Roza	281
7.47 No 6 Roza	281
7.48 No 7 Roza	282
7.49 No 8 Roza	282
7.50 No 13 Roza	284
7.51 Plagiarisme Roza	284
7.52 In 1 Roza	285
7.53 In 2 Roza	286
7.54 In 3 Roza	287
7.55 In 4 Roza	288
7.56 In 5 Roza	289
7.57 In 6 Roza	289
7.58 In 7 Roza	290
7.59 In 8 Roza	291
7.60 In 9 Roza	292
7.61 In 10 Roza	293
7.62 In 11 Roza	293
7.63 In 12 Roza	294
7.64 In 13 Roza	296
7.65 In 14 Roza	298
7.66 In 15 Roza	299
7.67 In 16 Roza	299
7.68 In 17 Roza	300
7.69 In 18 Roza	300
7.70 In 19 Roza	301

7.71 In 20 Roza	302
7.72 Error Roza	302
7.73 Tokenizer - fadila	303
7.74 Startified K-Fold Cross - fadila	304
7.75 chapter-7-test-in-splits-fadila	305
7.76 Contoh Penerapan .iloc- fadila	305
7.77 Tokenizer Fit On Text- fadila	306
7.78 Tokenizer Text To Matrix- fadila	307
7.79 Contoh Penerapan Np Amax- fadila	307
7.80 Contoh Penerapan Np Absolute- fadila	307
7.81 Contoh Penerapan Np.utils.to categorical - fadila	308
7.82 Perbedaan Deep NW Dan Deep Learn- fadila	310
7.83 Langkah Algoritma Konvolusi Berdasarkan NPM- fadila	310
7.84 Plagiarisme- fadila	311
7.85 Code Program Pada In [1] - fadila	311
7.86 Code Program Pada In [2] - fadila	313
7.87 Code Program Pada In [3] - fadila	314
7.88 Code Program Pada In [4] - fadila	315
7.89 Code Program Pada In [5] - fadila	315
7.90 Code Program Pada In [6] - fadila	316
7.91 Code Program Pada In [7] - fadila	317
7.92 Code Program Pada In [8] - fadila	318
7.93 Code Program Pada In [9] - fadila	318
7.94 Code Program Pada In [10] - fadila	320
7.95 Code Program Pada In [11] - fadila	321
7.96 Code Program Pada In [12] - fadila	322
7.97 Code Program Pada In [13] - fadila	325
7.98 Code Program Pada In [14] - fadila	326
7.99 Code Program Pada In [15] - fadila	327
7.100Code Program Pada In [16] - fadila	327
7.101Code Program Pada In [17] - fadila	328
7.102Code Program Pada In [18] - fadila	328
7.103Code Program Pada In [19] - fadila	330
7.104Code Program Pada In [20] - fadila	330
7.105Error 1 - fadila	331
7.106Penanganan Error 1 - fadila	331

7.107	Error 2 - fadila	331
7.108	Penanganan Error 2 - fadila	332
A.1	Form nilai bagian 1.	341
A.2	form nilai bagian 2.	342

Chapter 1

Mengenal Kecerdasan Buatan dan Scikit-Learn

Buku umum yang digunakan adalah [2] dan untuk sebelum UTS menggunakan buku *Python Artificial Intelligence Projects for Beginners*[1]. Dengan praktek menggunakan python 3 dan editor anaconda dan library python scikit-learn. Tujuan pembelajaran pada pertemuan pertama antara lain:

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

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

1.1 Teori

Praktek teori penunjang yang dikerjakan :

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

1.2 Instalasi

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

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

1.3 Penanganan Error

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

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

1.4 Fadila/1164072

1.4.1 Teori

Teori mencakup resume dari beberapa pembahasan. yaitu :

1. Tentang Kecerdasan Buatan

- Definisi Kecerdasan Buatan.

Kecerdasan Buatan biasa disebut dengan istilah AI (Artificial Intelligence) . AI sendiri merupakan suatu cabang dalam bidang sains komputer sains dimana mengkaji tentang bagaimana cara untuk melengkapi sebuah komputer dengan kemampuan atau kepintaran layaknya atau mirip dengan yang dimiliki manusia. Sebagai contoh, sebagaimana komputer dapat berkomunikasi dengan pengguna baik menggunakan kata, suara maupun lain sebagainya . Dengan kemampuan ini, diharapkan komputer mampu mengambil keputusan sendiri untuk berbagai kasus yang ditemuinya kemudian itulah yang disebut dengan kecerdasan buatan.

Kecerdasan buatan makin canggih dengan kemampuan komputer dalam memperbarui pengetahuannya dengan banyaknya testing dan perkembangan target analisa. Untuk kecerdasan buatan ada banyak contoh dan jenisnya. Salah satu contoh yang paling terkenal dari Artificial Intelligence ialah Google Assistant. Google Assistant digunakan untuk kemudahan user dalam menemukan berbagai hal maupun penyettingan langsung terhadap smartphone yang digunakan dan masih banyak lagi.

- Sejarah Kecerdasan Buatan

Artificial intelligence merupakan inovasi baru di bidang ilmu pengetahuan. Mulai terbentuk sejak adanya komputer modern dan kira-kira terjadi sekitaran tahun 1940 dan 1950. Ilmu pengetahuan komputer ini khusus ditujukan dalam perancangan otomatisasi tingkah laku cerdas dalam sistem kecerdasan komputer.

Pada awalnya, kecerdasan buatan hanya ada di universitas-universitas dan laboratorium penelitian, serta hanya sedikit produk yang dihasilkan dan dikembangkan. Menjelang akhir 1970-an dan 1980-an, mulai dikembangkan secara penuh dan hasilnya berangsur-angsur dipublikasikan di khalayak umum.

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

Figure 1.1: error model persistence

2. Penanganan Model Persistence

- Pertama-tama silahkan membuka command prompt
- Kemudian masukkan perintah untuk melakukan instalasi module joblib perintahnya ialah : pip install joblib
- hasilnya akan nampak seperti pada gambar yang ditampilkan



Figure 1.2: capturing

Jika kita berbicara tentang AI atau Artificial Intelligence maka kita tidak bisa melupakan seorang sosok yang sangat terkenal pada bidang tersebut yaitu bapak John McCarthy. McCarthy mendapatkan gelar sarjana matematika dari California Institute of Technology (Caltech) pada September 1948. Dari masa kuliahnya itulah ia mulai mengembangkan ketertarikannya pada mesin yang dapat menirukan cara berpikir manusia. McCarthy kemudian melanjutkan pendidikan ke program doktoral di Princeton University.

McCarthy kemudian mendirikan dua lembaga penelitian kecerdasan buatan. Kedua lembaga AI itu adalah Stanford Artificial Intelligence Laboratory dan MIT Artificial Intelligence Laboratory. Di lembaga-lembaga inilah bermunculan inovasi pengembangan AI yang meliputi bidang human skill, vision, listening, reasoning dan movement of limbs. Bahkan salah satu lembaga yang didirikan itu, Stanford Artificial Intelligence pernah mendapat bantuan dana dari Pentagon untuk membuat teknologi-teknologi luar angkasa.

- Perkembangan Kecerdasan Buatan

Teknologi Artificial Intelligence semakin ramai dibahas dalam berbagai diskusi teknologi di seluruh dunia. Menurut kebanyakan orang, pekerjaan seperti kasir, operator telepon, pengendara truk, dan lainnya sangat berpeluang besar untuk tergantikan oleh Artificial Intelligence. Mengapa terjadi

hal demikian? dikarenakan memang bahwa AI lebih unggul dalam hal kinerja, fitur dan lain sebagainya. Namun, dalam beberapa aspek memang pekerja manusia masih unggul dibandingkan AI itu sendiri.

Para generasi muda yang ada di dunia terutama di daerah Asia terlihat sudah memahami fungsi dan efek dari AI dalam kehidupan kita sehari-hari. Berdasarkan survei yang dilakukan oleh Microsoft, terdapat 39 persen responden yang mempertimbangkan untuk menggunakan mobil tanpa pengemudi dan 36 persen lainnya setuju bahwa robot masa depan dengan software untuk beroperasi mampu meningkatkan produktivitas. Dari survey tersebut kita sebagai pengguna AI harus lebih bijaksana dalam pengembangan dan penggunaan dari AI sehingga tanpa memberikan efek samping terhadap etos kerja dan keseharian kita sebagai pengguna dalam kehidupan sehari-hari.

3. Tentang Pengertian Terhadap Ilmu Yang Lain

- Supervised Learning adalah pendekatan dimana sudah terdapat data yang dilatih selain itu juga terdapat variable yang ditargetkan sehingga tujuan dari pendekatan ini yaitu mengelompokan suatu data ke data yang sudah ada.
- Klasifikasi adalah pembagian sesuatu menurut kelas-kelas (class). Menurut Ilmu Pengetahuan, Klasifikasi merupakan proses pengelompokan benda berdasarkan ciri-ciri persamaan dan juga perbedaan.
- Regresi adalah metode analisis statistik yang digunakan untuk melihat pengaruh antara dua ataupun lebih variabel.
- Unsupervised Learning berbeda dengan Supervised Learning. Perbedaannya ialah unsupervised learning tidak memiliki data latih, sehingga dari data yang ada kita mengelompokan data tersebut menjadi 2 ataupun 3 bagian dan seterusnya.
- Dataset adalah objek yang merepresentasikan data dan juga relasi yang ada di memory. Strukturnya mirip dengan data di database, namun bedanya dataset berisi koleksi dari data table dan data relation.
- Training Set adalah set digunakan oleh algoritma klasifikasi . Dapat di contohkan dengan : decision tree, bayesian, neural network dll. Semuanya dapat digunakan untuk membentuk sebuah model classifier.

- Testing Set adalah set yang digunakan untuk mengukur sejauh mana classifier berhasil melakukan klasifikasi dengan benar.

1.4.2 Instalasi

Untuk Instalasinya mencakup i beberapa pembahasan dan tutorial. yaitu :

1. Instalasi Scikit-Learn Dari Anaconda

- Instalasi Anaconda
 - Pertama-tama silahkan pastikan bahwa anda telah melakukan instalasi software Anaconda.
 - Apabila belum, silahkan buka web browser anda untuk melakukan pengunduhan software Anaconda
 - Setelah terunduh, silahkan klik kanan lalu run administrator pada software Anaconda
 - Silahkan lakukan penginstalan dengan menekan tombol install pada tampilan instalasi
 - Kemudian tekan tombol next maka akan sampai pada tampilan diatas

```
C:\WINDOWS\system32>pip install joblib
Collecting joblib
  Downloading https://files.pythonhosted.org/packages/cd/c1/50a758e8247561e58cb87305b1e99b171b8c767b15b12a1734001f41d350
/joblib-0.13.2-py3-none-any.whl (278kB)
  100% |████████████████████████████████| 286kB 2.3MB/s
distributed 1.21.0 requires msgpack, which is not installed.
Installing collected packages: joblib
Successfully installed joblib-0.13.2
You are using pip version 10.0.1, however version 19.0.3 is available.
You should consider upgrading via the 'python -m pip install --upgrade pip' command.
C:\WINDOWS\system32>
```

Figure 1.3: penanganan error model persistence

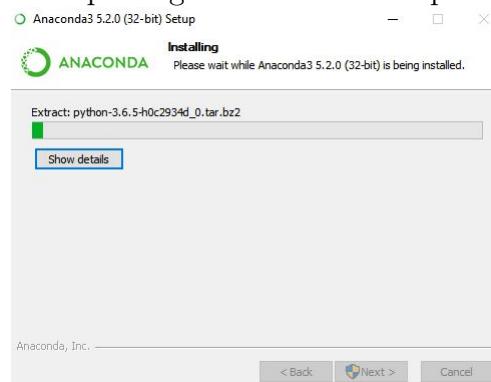


Figure 1.4: install anaconda 1

1. Pengujian Penanganan Model Persistence

- Setelah melakukan penginstalan maka kita harus menguji keberhasilan penginstalan
 - Caranya dengan mengecek lewat command prompt bahwa module joblib-nya telah terdefinisikan di python
 - Silahkan ketikkan perintah python, lalu masukkan perintah sebagai berikut :
- ```
from joblib import dump, load
```
- Maka hasilnya akan nampak seperti pada gambar yang ditampilkan

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

Figure 1.5: penanganan error model persistence

1. Selanjutnya apabila instalasi tersebut telah selesai maka silahkan menekan tombol next
2. Tampilan selanjutnya akan seperti ini

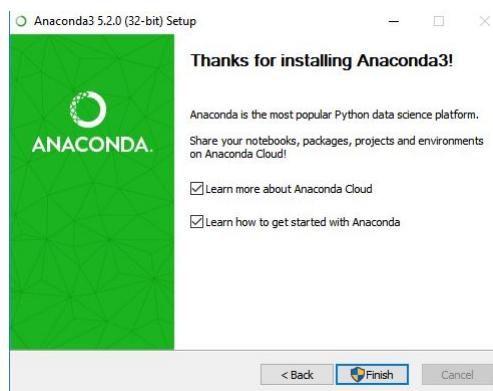


Figure 1.6: install anaconda 2

3. Apabila tampilannya telah sesuai dengan contoh gambar maka instalasi telah selesai
- Instalasi Library Scikit Learn
  1. Silahkan membuka web browser untuk melakukan pengunduhan untuk library scikit dari anaconda.

2. Silahkan mengunjungi halaman ini untuk melakukan pengunduhan library scikit dari anaconda.  
<https://anaconda.org/anaconda/scikit-learn>.
3. Setelah terdownload silahkan melakukan instalasi lanjutan menggunakan Command Prompt
4. Silahkan masukkan perintah berikut untuk melakukan pengecekan bahwa anaconda anda telah terpasang dengan baik.  

```
conda -version
python -version
```
5. Tampilannya akan nampak seperti berikut :



```
Microsoft Windows [Version 10.0.17134.598]
(c) 2018 Microsoft Corporation. All rights reserved.

C:\Users\ASUS>conda -version
conda 4.5.4

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

C:\Users\ASUS>
```

Figure 1.7: Pengecekan Anaconda

6. Selanjutnya silahkan masukkan perintah berikut untuk melakukan instalasi pip scikit-learn  
perintahnya : pip install -U scikit-learn
7. Tampilannya akan nampak seperti berikut :



```
!mercurial@K0101 install -U scikit-learn
Collecting scikit-learn
 Downloading https://files.pythonhosted.org/packages/ea/cf/09e0cb07b8ae0f2233a6257da1298787322d47202a2cscf/scikit_learn-0.20.2-cp36-cp36m-win32.whl (6.3M)
 100% |████████████████████████████████| 6.3M 1.1MB/s
Requirement already up-to-date: numpy<1.18.0, >=1.17.0 in c:\programdata\anaconda\lib\site-packages (from scikit-learn) (1.17.0)
Requirement already up-to-date: scipy<0.19.0, >=0.18.5 in c:\programdata\anaconda\lib\site-packages (from scikit-learn) (0.18.5)
Installing collected packages: scikit-learn
 Preparing metadata for scikit-learn-0.20.2
 Installing scikit-learn-0.20.2
 Successfully installed scikit-learn-0.20.2
Successfully installed scikit-learn-0.20.2
You are using pip version 10.0.3, however version 19.0.3 is available.
To update, run the command "python -m pip install --upgrade pip".
```

Figure 1.8: instalasi pip scikit-learn

8. Selanjutnya silahkan masukkan perintah berikut untuk melakukan instalasi conda scikit-learn  
perintahnya : conda install scikit-learn
9. Tampilannya akan nampak seperti berikut :
10. Apabila telah dipraktekan seperti langkah-langkah dan menghasilkan tampilan seperti contoh diatas, maka instalasi scikit-learn dari anaconda berhasil dilakukan
11. Kemudian untuk pengujian yang lain yaitu pengujian untuk mengecek codingan anaconda

```

c:\Users\ASUS>conda install scikit-learn
Solving environments: done
Package Plan
 environment location: C:\ProgramData\Anaconda
 added / updated specs:
 scikit-learn

The following packages will be downloaded:
 package | build
 conda-4.6.7 | py36_0 1.7 MB

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

Proceed ((y)/(n)) y

Downloading and Extracting Packages
conda: 4.6.7-py36_0 | ## | 100%
Preparing transaction: done
Verifying transaction: done
Executing transaction: done

```

Figure 1.9: instalasi conda scikit-learn

```

In [42]: solok = pd.get_dummies(solok, columns=['sex', 'school', 'address', 'famsize', 'Pstatus', 'Mjob',
...: 'Fjob', ...: 'reason', 'guardian', 'schoolsup', 'famsup', 'paid', 'activities', ...: 'nursery', 'higher', 'internet', 'romantic'])
...: solok.head()
Out[42]:
 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 1.10: uji coba codingan

12. Contoh uji coba codingannya dapat dilihat pada gambar berikut
13. Berdasarkan pengujian tersebut maka dapat dipastikan bahwa anaconda telah ter-include ke dalam python dan dieksekusi dengan script python
14. Setelah pengeksekusianya berdasarkan scripts python, terdapatlah keluaran yang sesuai
15. Keluaran tersebut yang menandakan bahwa anacondanya berfungsi dengan baik.

- Loading An Example Dataset

- Penerapan Loading An Example Dataset Pada Python Di CMD
  1. Pertama-tama silahkan buka command prompt di laptop anda
  2. Selanjutnya masuk ke python
  3. Setelah masuk kedalam python, silahkan masukkan perintah seperti pada gambar berikut :
  4. Secara keseluruhan, hasilnya pada command prompt akan nampak seperti gambar tersebut
  5. Apabila tampilanya telah nampak seperti gambar diatas, maka pengujianya telah selesai dan berhasil.
- Penjelasan Perintah Yang Di Uji

```

In [43]: solok = solok.sample(frac=1)
...: # split training and testing data
...: solok_train = solok[:500]
...: solok_test = solok[500:]
...:
...: solok_train_att = solok_train.drop(['pass'], axis=1)
...: solok_train_pass = solok_train['pass']
...:
...: solok_test_att = solok_test.drop(['pass'], axis=1)
...: solok_test_pass = solok_test['pass']
...:
...: solok_att = solok.drop(['pass'], axis=1)
...: solok_pass = solok['pass']
...:
...: # number of passing students in whole dataset:
...: import numpy as np
...: print("Passing: %d out of %d (%.2f%%)" % (np.sum(solok_pass), len(solok_pass),
100*float(np.sum(solok_pass)) / len(solok_pass)))
Passing: 166 out of 395 (42.03%)

```

Figure 1.11: pengujian loading an example dataset

```

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

```

Figure 1.12: pengujian loading an example dataset

1. Perhatikan perintah yang telah dieksekusi ini :
2. Penjelasan untuk baris pertama ialah :  
Perintahnya yaitu memasukkan dan memanggil dataset dari sklearn
3. Penjelasan untuk baris kedua ialah :  
Terdapat variabel baru yaitu iris. Dimana variabel iris memanggil datasets dan di dalamnya akan ngeload ( menampilkan ) load iris.
4. Penjelasan untuk baris ketiga ialah :  
Kemudian ada juga variabel baru lainnya yaitu digits yang akan memanggil dataset dan di dalamnya akan ngeload ( menampilkan ) load digits
5. Selanjutnya untuk perintah Print( digits.data ) ditujukan untuk menampilkan output dari pengeksekusian variabel digits dan akan berupa data.
6. Hasilnya printnya sebagai berikut :

```

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

```

Figure 1.13: hasil print uji cobat

7. Untuk penjelasan uji cobanya sudah selesai.

- Learning And Predicting

- Penerapan Learning Dan Predicting Pada Python Di CMD

1. Pertama-tama silahkan buka command prompt di laptop anda
2. Selanjutnya masuk ke python
3. Setelah masuk kedalam python, silahkan masukkan perintah ( scriptnya ) sesuai dengan contoh yang akan diberikan

```
In [40]: import pandas as pd
...: solok = pd.read_csv('student-mat.csv', sep=',')
...: len(solok)
Out[40]: 395
```

Figure 1.14: pengujian learning dan predicting

4. Contohnya nampak seperti pada gambar.
  5. Apabila tampilanya telah nampak seperti gambar diatas, maka pen-gujiannya telah selesai dan berhasil.
- Penjelasan Perintah Yang Di Uji, ( sesuai dengan contoh perintah pada gambar).
1. Penjelasan untuk baris 1 ialah :  
Memanggil dan memasukkan datasets dari sklearn
  2. Penjelasan untuk baris 2 ialah :  
membuat variabel iris yang memanggil load data pada datasets tanpa parameter
  3. Penjelasan untuk baris 3 ialah :  
membuat variabel digits yang memanggil load digits dari datasets tanpa parameter
  4. Penjelasan untuk baris 4 ialah :  
melakukan perintah print data yang akan menampilkan data dari eksekusi variabel digits
  5. Penjelasan untuk baris 5 ialah :  
hasil eksekusi
  6. Penjelasan untuk baris 6 ialah :  
membuat clf pada module fit metode dengan menggunakan 2 para-mater yaitu digits data dan digits target

7. Penjelasan untuk baris 7 ialah :

svc ini mengimplementasikan yang namanya data berupa klasifikasi dukungan vektor.

8. Untuk penjelasan uji cobanya sudah selesai.

- Model Persistence

- Penerapan Model Persistence Pada Python Di CMD

1. Pertama-tama silahkan buka command prompt di laptop anda
2. Selanjutnya masuk ke python
3. Setelah masuk kedalam python, silahkan masukkan perintah ( script-snya ) sesuai dengan contoh yang akan diberikan

```
[1]: In [1]: Python 3.6.5 |Anaconda, Inc.| (default, Mar 29 2018, 13:23:52) [MSC v.1900 32 bit (Intel)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> from sklearn import svm
>>> import numpy as np
>>> from sklearn import datasets
>>> clf = svm.SVC(gamma='scale')
>>> iris = datasets.load_iris()
>>> X = iris.data[:, :2]
>>> y = iris.target
>>> clf.fit(X, y)
SVC(C=1.0, cache_size=200, class_weight=None, coef0=0.0,
 decision_function_shape='ovr', degree=3, gamma='scale',
 kernel='rbf', max_iter=-1, probability=False, random_state=None, shrinking=True,
 tol=0.001, verbose=False)
>>>
```

Figure 1.15: pengujian model persistence 1

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

Figure 1.16: pengujian model persistence 2

4. Contohnya nampak seperti pada gambar.

5. Apabila tampilanya telah nampak seperti gambar diatas, maka pengujianya telah selesai dan berhasil.

- Penjelasan Perintah Yang Di Uji, ( sesuai dengan contoh perintah pada gambar).

- Model Persistence 1

1. Penjelasan untuk baris 1 ialah :

Memanggil ataupun memasukkan datasets dari sklear.

2. Penjelasan untuk baris 2 ialah :

Memanggil ataupun memasukkan svm dari sklearn

3. Penjelasan untuk baris 3 ialah :

Membuat variabel baru yaitu clf dimana akan memanggil svm.SVC yang telah mendefinisikan sebuah parameter yaitu gamma.

4. Penjelasan untuk baris 4 ialah :

Membuat variabel baru lainnya yaitu iris dimana akan memanggil datasets yang didalamnya akan ngeload data iris.

5. Penjelasan untuk baris 5 ialah :

Membuat variabel baru lainnya untuk X dan Y dengan mendefinisikan pemanggilan iris data dan iris target.

6. Penjelasan untuk baris 6 ialah :

Variabel clf dipasang pada model fit metode dengan parameter X dan Y

7. Penjelasan untuk baris 7 ialah :

Kemudian svc ini mengimplementasikan yang namanya data berupa klasifikasi dukungan vektor.

- Model Persistence 2

1. Penjelasan untuk baris 1 ialah :

Melakukan pemanggilan terhadap library pickle

2. Penjelasan untuk baris 2 ialah :

Membuat variabel baru yaitu s dengan pemanggilan pickle dumps dengan pendefinisian variabel clf

3. Penjelasan untuk baris 3 ialah :

Membuat variabel clf2 dengan memanggil pickle loads dengan pendefinisian variabel s

4. Penjelasan untuk baris 4 ialah :

Prediksi nilai baru dari variabel clf2 dengan parameternya yaitu X

5. Penjelasan untuk baris 5 ialah :

set array dari prediksi variabel clf2

6. Penjelasan untuk baris 6 ialah :

Parameter X dengan array 0 akan menghasilkan set array 0 juga

7. Untuk penjelasan uji cobanya sudah selesai.

- Conventions

- Penerapan Conventions Pada Python Di CMD

1. Type Casting

\* Pertama-tama silahkan buka command prompt di laptop anda

\* Selanjutnya masuk ke python

- \* Setelah masuk kedalam python, silahkan masukkan perintah ( scriptsnya ) sesuai dengan contoh yang akan diberikan

```

C:\Users\ASUS>python
Microsoft Windows [Version 10.0.17134.586]
(c) 2017 Microsoft Corporation. All Rights Reserved.

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

```

Figure 1.17: pengujian type casting 1

```

C:\Users\ASUS>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.
>>> import numpy as np
>>> from sklearn.svm import SVC
>>> iris = datasets.load_iris()
>>> X = iris.data
>>> y = iris.target
>>> clf = 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, 1, 2]
>>> 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: pengujian type casting 2

## 2. Refitting And Updating Parameters

- \* Pertama-tama silahkan buka command prompt di laptop anda
- \* Selanjutnya masuk ke python
- \* Setelah masuk kedalam python, silahkan masukkan perintah ( scriptsnya ) sesuai dengan contoh yang akan diberikan

```

C:\Users\ASUS>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.
>>> 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: pengujian refitting and updating

## 3. Multiclass And Multilable Fitting

- \* Pertama-tama silahkan buka command prompt di laptop anda
- \* Selanjutnya masuk ke python

```

c:\Users\ASUS>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 SVC
>>> from sklearn.multiclass import OneVsRestClassifier
>>> from sklearn.preprocessing import LabelBinarizer
>>> X = [[1, 2], [2, 4], [4, 5], [3, 2], [3, 1]]
>>> y = [0, 0, 1, 1, 2]
>>> classif = OneVsRestClassifier(estimator=SVC(gamma='scale', random_state=0))
>>> classif.fit(X, y).predict(X)
array([0, 0, 1, 1, 2])
>>> y = LabelBinarizer().fit_transform(y)
>>> classif.fit(X, y).predict(X)
array([[1, 0, 0],
 [0, 1, 0],
 [0, 0, 1],
 [0, 0, 1],
 [0, 0, 0]])

```

Figure 1.20: pengujian multiclass and multiable 1

```

>>> 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, 0, 0, 0],
 [1, 0, 1, 0, 0],
 [0, 1, 0, 1, 0],
 [1, 0, 1, 0, 0],
 [1, 0, 1, 0, 0]])

```

Figure 1.21: pengujian multiclass and multiable 2

- \* Setelah masuk kedalam python, silahkan masukkan perintah ( scriptsnya ) sesuai dengan contoh yang akan diberikan
- 4. Apabila semua proses yang telah dilakukan terlihat seperti contoh-contoh diatas, maka pengujian telah selesai.
- Penjelasan Perintah Yang Di Uji, berdasarkan contoh perintah-perintah diatas :
- 1. Type Casting :
  - \* Type Casting 1
    - Penjelasan untuk baris 1 ialah :
 

Memasukkan dan memanggil module / library numpy sebagai np
    - Penjelasan untuk baris 2 ialah :
 

Memasukkan dan memanggil random projection dari sklearn
    - Penjelasan untuk baris 3 ialah :
 

Membuat variabel rng, dimana memanggil np yang akan mengambil dan mengeksekusi random state dengan parameter 0
    - Penjelasan untuk baris 4 ialah :
 

Membuat variabel baru lainnya yaitu X dengan memanggil variabel rng dengan 2 parameter yaitu 10 dan 2000
    - Penjelasan untuk baris 5 ialah :
 

Membuat variabel X lagi namun dengan pemanggilan yang berbeda yaitu array dari np dengan parameternya x dan dtype='float32'.

- Penjelasan untuk baris 6 ialah :  
Pemanggilan dtype dari variabel X
  - Penjelasan untuk baris 7 ialah :  
Hasil dari pemanggilan dtype dari variabel X
  - Penjelasan untuk baris 8 ialah :  
Membuat variabel transformer dengan pemanggilan gaussian random projection
  - Penjelasan untuk baris 9 ialah :  
Membuat variabel baru yaitu X new dengan memanggil variabel transformer yang berada pada model fit metode dengan parameternya yaitu X
  - Penjelasan untuk baris 10 ialah :  
Pemanggilan dtype dari variabel X new
  - Penjelasan untuk baris 11 ialah :  
Hasil dari pemanggilan dtype variabel X new
  - Untuk penjelasan uji cobanya sudah selesai.
- \* Type Casting 2
- Penjelasan untuk baris 1 ialah :  
Memasukkan dan memanggil dataset dari sklearn
  - Penjelasan untuk baris 2 ialah :  
Memasukkan dan memanggil csv dari sklearn
  - Penjelasan untuk baris 3 ialah :  
Membuat variabel iris dengan memanggil load iris dari datasets
  - Penjelasan untuk baris 4 ialah :  
Membuat variabel clf dengan memanggil svc dengan parameter gamma
  - Penjelasan untuk baris 5 ialah :  
Membuat clf pada module fit metode dengan 2 parameter yaitu iris data dan iris target
  - Penjelasan untuk baris 6 ialah :  
membuat variabel list dengan prediksi clf
  - Penjelasan untuk baris 7 ialah :  
SVC mengimplementasikan yang namanya data berupa klasifikasi dukungan vektor.

- Penjelasan untuk baris 8 ialah :  
membuat variabel list dengan prediksi iris data
  - Penjelasan untuk baris 9 ialah :  
hasil dari prediksi list clf
  - Untuk penjelasan uji cobanya sudah selesai.
- \* Refting And Updating Parameters :
- Penjelasan untuk baris 1 ialah :  
Memasukkan dan memanggil module / library numpy sebagai np
  - Penjelasan untuk baris 2 ialah :  
Memasukkan dan memanggil SVC dari sklearn.svm
  - Penjelasan untuk baris 3 ialah :  
Membuat variabel rng, dimana memanggil np yang akan mengambil dan mengeksekusi random state dengan parameter 0
  - Penjelasan untuk baris 4 ialah :  
Membuat variabel baru lainnya yaitu X dengan memanggil variabel rng dengan 2 parameter yaitu 100 dan 10
  - Penjelasan untuk baris 5 ialah :  
Membuat variabel Y dimana memanggil binomial dari rng dengan 3 parameter yaitu 1, 0.5 dan 100.
  - Penjelasan untuk baris 6 ialah :  
Memuat variabel baru lainnya itu X test dimana memanggil rand dari rng dengan 2 parameter yaitu 5 dan 10.
  - Penjelasan untuk baris 7 ialah :  
Membuat variabel clf dengan mendefinisikan SVC tanpa parameter
  - Penjelasan untuk baris 8 ialah :  
Melakukan parameter set dari clf dengan parameter kernel='linear'.
  - Penjelasan untuk baris 9 ialah :  
SVC mengimplementasikan yang namanya data berupa klasifikasi dukungan vektor.
  - Penjelasan untuk baris 10 ialah :  
Membuat prediksi clf dengan parameternya yaitu variabel X test
  - Penjelasan untuk baris 11 ialah :  
set array yang dihasilkan oleh prediksi clf

- Penjelasan untuk baris 12 ialah :  
Melakukan parameter set dari clf dengan parameter kernel='rbf', gamma='scale' dan fit yaitu X dan Y
  - Penjelasan untuk baris 13 ialah :  
SVC mengimplementasikan yang namanya data berupa klasifikasi dukungan vektor.
  - Penjelasan untuk baris 14 ialah :  
Membuat prediksi clf dengan parameteranya yaitu variabel X test
  - Penjelasan untuk baris 15 ialah :  
set array yang dihasilkan oleh prediksi clf
  - Untuk penjelasan uji cobanya sudah selesai.
- \* Multiclass And Multilable Fitting
- Multiclass And Multilable Fitting 1
    1. Penjelasan untuk baris 1 ialah :  
Memasukkan dan memanggil SCV dari sklearn.svm
    2. Penjelasan untuk baris 2 ialah :  
Memasukkan dan memanggil OneVsRestClassifier dari sklearn.svm
    3. Penjelasan untuk baris 3 ialah :  
Memasukkan dan memanggil LabelBinarizer dari sklearn.preprocessing
    4. Penjelasan untuk baris 4 ialah :  
Membuat variabel X dengan beberapa parameter
    5. Penjelasan untuk baris 5 ialah :  
Membuat variabel Y dengan beberapa parameter
    6. Penjelasan untuk baris 6 ialah :  
Membuat variabel classif dengan memanggil OneVsRestClassifier yang didalamnya terdapat 2 parameter yaitu gamma dan random state.
    7. Penjelasan untuk baris 7 ialah :  
Membuat prediksi parameter X dari variabel classif yang berada dalam module fit metode dengan parameter X dan Y
    8. Penjelasan untuk baris 8 ialah :  
set array dari prediksi classif
    9. Penjelasan untuk baris 9 ialah :

- Membuat variabel Y baru dengan memanggil LabelBinarizer tanpa parameter dan fit transform dengan parameter Y
10. Penjelasan untuk baris 10 ialah :  
Membuat prediksi parameter X dari variabel classif yang berada dalam module fit metode dengan parameter X dan Y
  11. Penjelasan untuk baris 11 ialah :  
set array dari prediksi classif
    - Multiclass And Multilable Fitting 2
    1. Penjelasan untuk baris 1 ialah :  
Memasukkan dan memanggil MultiLabelBinarizer dari sklearn.preprocessing
    2. Penjelasan untuk baris 2 ialah :  
Membuat variabel Y dengan beberapa parameter / nilai
    3. Penjelasan untuk baris 3 ialah :  
Membuat variabel Y dengan memanggil MultiLabelBinarizer tanpa parameter dan Fit transform dengan parameter y
    4. Penjelasan untuk baris 4 ialah :  
Membuat prediksi dengan parameter X dari classif pada module fit metode dengan parameter X dan Y
    5. Penjelasan untuk baris 5 ialah :  
set array dari prediksi classif
    6. Untuk penjelasan uji cobanya sudah selesai.

### 1.4.3 Penanganan Error

Terdapat error pada pengujian diatas dan penanganannya, yaitu:

1. Model Persistence
  - Errornya ditandai dengan tidak terdefinisinya module joblib pada komputer
  - Hal itulah yang menyebabkan tidak terprosesnya perintah terkait

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

Figure 1.22: error model persistence

## 2. Penanganan Model Persistence

- Pertama-tama silahkan membuka command prompt
- Kemudian masukkan perintah untuk melakukan instalasi module joblib perintahnya ialah : pip install joblib
- hasilnya akan nampak seperti pada gambar yang ditampilkan

```
C:\WINDOWS\system32>pip install joblib
Collecting joblib
 Downloading https://files.pythonhosted.org/packages/cd/c1/50a758e8247561e58cb87305b1e90b171b8c767b15b12a1734001f41d350
/joblib-0.13.2-py2.py3-none-any.whl (278kB)
100% |████████████████████████████████| 286kB 2.3MB/s
Requirement already up-to-date: numpy >= 1.11 in ./pip-req-build which is not installed.
Installing collected packages: joblib
Successfully installed joblib-0.13.2
You are using pip version 10.0.1, however version 19.0.3 is available.
You should consider upgrading via the 'python -m pip install --upgrade pip' command.
C:\WINDOWS\system32>
```

Figure 1.23: penanganan error model persistence

## 1. Pengujian Penanganan Model Persistence

- Setelah melakukan penginstalan maka kita harus menguji keberhasilan penginstalan
- Caranya dengan mengecek lewat command prompt bahwa module joblibnya telah terdefinisikan di python
- Silahkan ketikkan perintah python, lalu masukkan perintah sebagai berikut :  
from joblib import dump, load
- Maka hasilnya akan nampak seperti pada gambar yang ditampilkan

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

Figure 1.24: penanganan error model persistence

## 1.5 Rahmi Roza/1164085

### 1.5.1 Teori

Teori mencakup resume dari beberapa pembahasan. yaitu :

- Definisi Kecerdasan Buatan.

Kecerdasan Buatan adalah salah satu cabang Ilmu pengetahuan berhubungan dengan pemanfaatan mesin untuk memecahkan persoalan yang rumit dengan cara yang lebih manusiawi. Hal ini biasanya dilakukan dengan mengikuti/mencontoh karakteristik dan analogi berpikir dari kecerdasan/Inteligensia manusia, dan menerapkannya sebagai algoritma yang dikenal oleh komputer.

Agar komputer bisa bertindak seperti dan sebaik manusia, maka komputer juga harus diberi bekal pengetahuan dan mempunyai kemampuan untuk menalar. Untuk itu AI akan mencoba untuk memberikan beberapa metoda untuk membekali komputer dengan kedua komponen tersebut agar komputer bisa menjadi mesin pintar.

- Sejarah Kecerdasan Buatan

Banyak orang percaya kecerdasan buatan akan memusnahkan kelangsungan hidup manusia saat mereka menyadari kekuatan yang dimilikinya. Semua bermula sejak Turing Machine. Berikut perjalanan kecerdasan buatan hingga akhirnya melahirkan robot dan robot seks.

Tahun 1950. Alan Turing memperkenalkan Turing Test dalam jurnal berjudul Computing Machinery and Intelligence. Pada musim panas 1956, Konferensi Dartmouth meluncurkan ide artificial intelligence dan IBM memulai riset tentang AI.

Tahun 1970. Sepanjang 1974 sampai 1980 merupakan gelombang pertama kecerdasan buatan. Pada periode ini pula pengumpulan dana untuk melakukan riset kecerdasan buatan mulai marak.

Tahun 1990. Pada 11 Mei 1997, Deep Blue Computer, kecerdasan buatan besutan IBM berhasil mengalahkan grand master catur asal Rusia, Garry Kasparov

Tahun 2000. Kendaraan bikinan tim peneliti dari Universitas Stanford, Amerika Serikat, berhasil menjadi kampiun dal DARPA Grand Challenge. Mobil swakemudi ini bisa melaju di gurun pasir sejauh 211 kilometer.

Tahun 2010. Watson, kecerdasan buatan besutan IBM, berhasil mengalahkan mantan juara Brad Rutter dan Ken Jennings dalam acara kuis Jeopardy pada pertengahan 2011. Appel, pada 14 Oktober tahun yang sama, memperkenalkan asistem pribadi berbasiskan kecerdasan buatan bernama Siri dalam iPhone 4s. Setahun kemudian, tepatnya Juni, tim dari Google Brain melatih komputer agar bisa mengenali seekor kucing

dari jutaan video di YouTube. ChatBot bikinin Eugene Goostman mengklaim telah memecahkan tes Turing dalam kompetisi yang digelar di Universitas Reading, Inggris. Imbasnya, pada Agustus tahun yang sama, banyak ilmuwan mengusulkan untuk membuat tes Turing yang baru. Sementara itu, terkesan dengan kemampuan Watson, NASA menggunakannya untuk penelitian bidang kedirgantaraan.

Tahun 2017. Google, Februari lalu, kembali membuat gebrakan soal kecerdasan buatan. Tim dari Google Deep Mind mengungkap kecerdasan buatan memiliki tingkat emosi dan kemarahan yang sama dengan manusia. Kecerdasan buatan pun bisa merasakan kalau dirinya ditipu. Tak mau kalah, Microsoft, April lalu, mendeteksi bahwa artificial intelligence ternyata juga bisa rasis. Yang paling fenomenal soal kecerdasan buatan pada tahun ini adalah robot seks bernama Harmony. Tak seperti robot seks pada umumnya, Harmony bisa merasakan cemburu dan mendeteksi penggunanya saat hendak menuju klimaks.

Jika kita berbicara tentang AI atau Artificial Intelligence maka kita tidak bisa melupakan seorang sosok yang sangat terkenal pada bidang tersebut yaitu bapak John McCarthy. McCarthy mendapatkan gelar sarjana matematika dari California Institute of Technology (Caltech) pada September 1948. Dari masa kuliahnya itulah ia mulai mengembangkan ketertarikannya pada mesin yang dapat menirukan cara berpikir manusia. McCarthy kemudian melanjutkan pendidikan ke program doktoral di Princeton University.

McCarthy kemudian mendirikan dua lembaga penelitian kecerdasan buatan. Kedua lembaga AI itu adalah Stanford Artificial Intelligence Laboratory dan MIT Artificial Intelligence Laboratory. Di lembaga-lembaga inilah bermunculan inovasi pengembangan AI yang meliputi bidang human skill, vision, listening, reasoning dan movement of limbs. Bahkan Salah satu lembaga yang didirikan itu, Stanford Artificial Intelligence pernah mendapat bantuan dana dari Pentagon untuk membuat teknologi-teknologi luar angkasa.

#### – Perkembangan Kecerdasan Buatan

Perkembangan kecerdasan buatan atau artificial intelligence (AI) dinilai tidak bisa dihentikan. Keberadaan AI justru dinilai akan semakin mempermudah kehidupan manusia dalam beraktivitas. Dalam pandangan Johnny

Lie sebagai Vice President of Cheetah Mobile, revolusi teknologi tidak bisa dihentikan. Menurutnya, hal itu sudah terjadi sejak puluhan tahun yang lalu. Cheetah Mobile sendiri merupakan perusahaan teknologi mobile yang fokus pada peranti lunak dan pada hari ini menyematkan unsur AI dalam layanannya.

Sebelumnya, beberapa pakar teknologi kenamaan dunia, seperti Elon Musk dan Stephen Hawking gencar memberikan peringatan bahwa AI yang terus berkembang dapat menjadi ancaman bagi umat manusia. Bahkan, keduanya bersama banyak ilmuwan lainnya membuat surat penegasan kepada PBB untuk mengawasi pertumbuhan AI.

Dengan AI, sebuah produk teknologi dapat bekerja lebih cerdas dan mandiri. Misalnya saja, sebuah aplikasi smartphone yang menggunakan AI dapat mempelajari kebiasaan pengguna. Alhasil, Anda tidak perlu repot lagi dalam melakukan suatu pekerjaan di ponsel.

- Tentang Pengertian Terhadap Ilmu Yang Lain

- Supervised learning adalah 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
- Klasifikasi adalah pembagian sesuatu menurut kelas-kelas ( class ). Menurut Ilmu Pengetahuan, Klasifikasi merupakan proses pengelompokan benda berdasarkan ciri-ciri persamaan dan juga perbedaan.
- Regresi adalah metode analisis statistik yang digunakan untuk melihat pengaruh antara dua ataupun lebih variabel.
- Unsupervised Learning berbeda dengan Supervised Learning. Perbedaannya ialah unsupervised learning tidak memiliki data latih, sehingga dari data yang ada kita mengelompokan data tersebut menjadi 2 ataupun 3 bagian dan seterusnya.
- Dataset adalah objek yang merepresentasikan data dan juga relasi yang ada di memory.
- Training Set adalah set digunakan oleh algoritma klasifikasi . Dapat di-contohkan dengan : decision tree, bayesian, neural network dll. Semuanya dapat digunakan untuk membentuk sebuah model classifier.

- Testing Set adalah set yang digunakan untuk mengukur sejauh mana classifier berhasil melakukan klasifikasi dengan benar.

### 1.5.2 Instalasi



Figure 1.25: gambar1

1. Baris 1 = memasukkan dan memanggil svm dari sklearn
2. Baris 2 = membuat variable



Figure 1.26: gambar2

1. Baris 1 = clf dipasang pada model fit metode
  2. Baris 2 = implementasikan klasifikasi dukungan vektor
- 
1. Baris 1 = prediksi nilai baru



Figure 1.27: gambar3



Figure 1.28: gambar4

2. Baris 2 = set array
  
1. Baris 1 = memasukkan dan memanggil datasets dari sklearn
2. Baris 2 = memasukkan dan memanggil svm dari sklearn
3. Baris 3 = membuat variable clf
4. Baris 4 = membuat variable iris
5. Baris 5 = membuat variable x, y
6. Baris 6 = clf dipasang pada model fit metode
7. Baris 7 = implementasikan klasifikasi dukungan vektor
8. Baris 8 = memanggil library pickle
9. Baris 9 = membuat variable s
10. Baris 10 = membuat variable clf2
11. Baris 11 = prediksi nilai baru
12. Baris 12 = set array
  
1. Baris 1 = memasukkan dan memanggil numpy sebagai np



Figure 1.29: gambar5

2. Baris 2 = memasukkan dan memanggil random projection dari sklearn
3. Baris 3 = membuat variable rng
4. Baris 4 = membuat variable rng
5. Baris 5 = membuat variable x
6. Baris 6 = pemanggilan variable x
7. Baris 7 = pemanggilan dtype
8. Baris 8 = membuat variable transformer
9. Baris 9 = membuat variable x new
10. Baris 10 = pemanggilan x new
11. Baris 11 = pemanggilan dtype

### 1.5.3 Penanganan Error

- Skrinsut Error.
- Tuliskan kode eror dan jenis errornya
  - Kode error = NameError: name 'roza' is not defined
  - jenis error = NameError
- Solusi pemecahan masalah error



Figure 1.30: gambar6



Figure 1.31: gambar7

## 1.6 Lusia Violita Aprilian/1164080

### 1.6.1 Artificial Intelligence

#### 1. Pengertian AI

Menurut Minsky, Kecerdasan Buatan ialah suatu ilmu yang mempelajari cara membuat komputer melakukan sesuatu seperti yang dilakukan oleh manusia. Lalu menurut Ensiklopedi Britannica, Kecerdasan Buatan ialah cabang ilmu komputer yang merepresentasi pengetahuan lebih banyak menggunakan symbol-simbol daripada bilangan, dan memproses informasi berdasarkan metode heuristic atau berdasarkan jumlah aturan.

Berdasarkan beberapa teori tentang kecerdasan buatan diatas, penulis menyimpulkan bahwa kecerdasan buatan adalah suatu ilmu yang membuat sebuah mesin menjadi cerdas, sehingga kecerdasan mesin tersebut mirip dengan kecer-

dasan manusia serta dapat mengambil keputusan sendiri untuk menyelesaikan sebuah masalah.

## 2. Sejarah dan Perkembangan

- 1941 ( Era Komputer Elektronik )

Pada era ini, telah ditemukan pertama kali yakni alat penyimpanan dan pemrosesan informasi atau disebut komputer elektronik. Ini juga digunakan untuk dasar pengembangan program ke arah AI.

- 1943 – 1956 ( Era Persiapan AI )

Pada tahun 1943, terdapat dua peneliti yakni Warren McCulloch dan Walter Pitts yang berhasil membuat sebuah model tiruan dari tiap neuron seperti on dan off. Mereka membuktikan bahwa setiap fungsi dapat dihitung dengan suatu jaringan sel saraf dan semua hubungan logis bisa diimplementasikan dengan struktur jaringan yang sederhana.

Pada tahun 1950, Norbert Wiener melakukan penelitian tentang prinsip teori feedback. Bentuk implementasi dari penelitian tersebut salah satunya adalah thermostat.

Pada tahun 1956, John McCarthy mencoba meyakinkan Minsky, Claude Shannon, dan Nathaniel Rochester untuk membantunya dalam melakukan penelitian di bidang automata, jaringan saraf, dan pembelajaran intelijensia. Mereka mengerjakan proyek ini kurang lebih selama 2 bulan di Universitas Dartmouth. Hasilnya adalah berupa program yang mampu berpikir non-numerik dan menyelesaikan masalah pemikiran, yang disebut Principia Mathematica. Berdasarkan hal ini, telah ditentukan bahwa McCarthy disebut sebagai father of Artificial Intelligence/ Bapak Kecerdasan Buatan.

- 1952 – 1969 ( Awal Perkembangan )

Pada tahun 1958, McCarthy di MIT AI Lab mengeluarkan bahasa pemrograman tingkat tinggi yaitu LISP, dimana sekarang sudah mulai sering digunakan dalam pembuatan program-program AI. Lalu, McCarthy membuat program yang disebut programs with common sense. Di program tersebut, dibuat sebuah rancangan untuk menggunakan pengetahuan dalam mencari solusi dari sebuah masalah.

Pada tahun 1959, Program komputer bernama General Problem Solver

berhasil dibuat oleh Herbert A. Simon, J.C. Shaw, dan Allen Newell. Program tersebut dirancang untuk memulai proses penyelesaian masalah secara manusiawi. Pada tahun yg sama Nathaniel Rochester dari IBM dan para mahasiswanya merilis sebuah program AI yaitu geometry theorem prover. Program ini dapat membuktikan bahwa suatu teorema menggunakan axioma-axioma yang ada.

Pada tahun 1963, program yang dibuat oleh James Slagle bisa menyelesaikan masalah integral tertutup untuk mata kuliah Kalkulus.

Pada tahun 1968, program analogi buatan Tom Evan dapat menyelesaikan masalah analogi geometri yang ada pada tes IQ.

- 1966 – 1974 ( Perkembangan AI Lambat )

Perkembangan AI mulai melambat pada tahun 1966 – 1974, yang disebabkan adanya beberapa kesulitan yang dihadapi seperti Program-program AI yang bermunculan hanya mengandung sedikit atau bahkan tidak mengandung sama sekali pengetahuan pada subjeknya, banyak terjadi kegagalan pada pembuatan program AI, serta terdapat beberapa batasan pada struktur dasar yang digunakan untuk menghasilkan perilaku intelijensia.

- 1969 – 1979 ( Sistem berbasis pengetahuan )

Pada tahun 1960, Ed Feigenbaum, Bruce Buchanan, dan Joshua Lederberg mulai merintis proyek bernama DENDRAL yaitu program yang digunakan untuk memecahkan masalah struktur molekul dari informasi yang didapatkan dari spectrometer massa. Dari segi diagnosa medis juga terdapat yang menemukan sistem berbasis Ilmu pengetahuan, yaitu Saul Amarel dalam proyek computer ini biomedicine. Proyek ini diawali dari keinginan untuk mendapatkan diagnosa penyakit berdasarkan pengetahuan yang ada pada mekanisme penyebab proses penyakit.

- 1980 – 1988 ( AI menjadi Industri )

Industrialisasi AI diawali dengan ditemukannya sebuah sistem pakar yang dinamakan R1 yang mampu mengkonfigurasi sistem-sistem komputer baru. Program tersebut mulai dijalankan di Digital Equipment Corporation (DEC), McDermott, pada tahun 1982. Pada tahun 1986, program ini telah berhasil menghemat biaya sebesar US 40 juta per tahun.

Pada tahun 1988, kelompok AI di DEC menjalankan program sistem pakar sebanyak 40 sistem pakar. Hampir semua perusahaan besar di USA mempunyai divisi Ai sendiri yang menggunakan maupun mempelajari sistem

pakar itu sendiri. Industri AI yang sedang ramai diperbincangkan juga melibatkan perusahaan-perusahaan besar seperti Carnegie Group, Inference, IntelliCorp, dan Technoledge yang menawarkan software tools untuk membangun sistem pakar. Perusahaan bidang hardware seperti LISP Machines Inc., Texas Instruments, Symbolics, dan Xerox dan lain-lain juga ikut berperan dalam membangun sebuah workstation yang dioptimasi untuk pembangunan program LISP. Sehingga, perusahaan yang sudah berdiri sejak tahun 1982 hanya menghasilkan beberapa juta US dollar per tahun meningkat menjadi 2 miliar US dollar per tahun pada tahun 1988.

- 1986 – sekarang ( kembalinya jaringan saraf tiruan ) Meskipun bidang ilmu komputer, telah melakukan penolakan terhadap jaringan saraf tiruan setelah diterbitkannya sebuah buku berjudul ‘Perceptrons’ karangan Minsky dan Papert, tetapi para ilmuwan masih terus mempelajari bidang ilmu tersebut dari sudut pandang yang lain, yaitu bidang fisika. Ahli fisika seperti Hopfield (1982) menggunakan teknik-teknik mekanika statistika untuk menganalisa sifat-sifat penyimpanan dan optimasi pada jaringan saraf. ahli psikolog, David Rumelhart dan Geoff Hinton melanjutkan penelitian tersebut tentang model jaringan saraf pada memori. Pada tahun 1985-an sedikitnya empat kelompok riset menemukan algoritma Back-Propagation. Algoritma ini pun akhirnya berhasil diimplementasikan ke dalam ilmu bidang komputer dan psikologi.

## 1.6.2 Supervised Learning dan Data

### 1. Supervised Learning

Sebuah pendekatan dengan kondisi dimana sudah ada terdapat kumpulan data yang dilatih atau ditraining, dan terdapat beberapa variabel yang sudah ditentukan sehingga tujuan dari pendekatan tersebut mengarah ke data yang sudah ada.

### 2. Klasifikasi

Sebuah sejenis program yang dapat menentukan objek yang ada termasuk jenis apa berdasarkan variabel-variabel yang sudah ditentukan.

### 3. Regresi

Sebuah metode yang digunakan untuk menentukan dan memprediksi berdasarkan hubungan sebab – akibat antara satu variabel ke variabel lainnya.

#### 4. Unsupervised Learning

Sebuah pendekatan yang dimana tidak memiliki data yang dilatih, namun ingin di kelompokkan berdasarkan beberapa variabel dengan kemauan sendiri.

#### 5. Data Set

Objek yang merepresentasikan data dan relasi didalam memori.

#### 6. Training Set

Himpunan dari berbagai pasangan objek, kelas yang dapat menunjukkan objek tersebut yang sudah diberi label.

#### 7. Testing Set

Himpunan data yang sudah berlabel lain, yang digunakan untuk mengukur persentase sampel yang diklasifikasikan dengan benar atau persentase sampel mengalami kesalahan.

### 1.6.3 Learning and Predicting

#### 1. Learning and predicting 1

```
from sklearn import svm
clf = svm.SVC(gamma=0.001, C=100.)
```

Figure 1.32: Learning and predicting 1

- Baris 1 = memasukkan dan memanggil svm dari sklearn
- Baris 2 = membuat variable

#### 2. Learning and predicting 2

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

Figure 1.33: Learning and predicting 2

- Baris 1 = clf dipasang pada model fit metode
- Baris 2 = implementasikan klasifikasi dukungan vektor

#### 3. Learning and predicting 3

- Baris 1 = prediksi nilai baru
- Baris 2 = set array

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

Figure 1.34: Learning and predicting 3

#### 1.6.4 Model Persistence

- menjelaskan maksud dari tulisan tersebut dan mengartikan per baris.

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

import pickle
s = pickle.dumps(clf)
clf2 = pickle.loads(s)
clf2.predict(X[0:1])
array([8])
y[0]
0
```

Figure 1.35: Model Persistence

- Baris 1 = memasukkan dan memanggil datasets dari sklearn
- Baris 2 = memasukkan dan memanggil svm dari sklearn
- Baris 3 = membuat variable clf
- Baris 4 = membuat variable iris
- Baris 5 = membuat variable x, y
- Baris 6 = clf dipasang pada model fit metode
- Baris 7 = implementasikan klasifikasi dukungan vektor
- Baris 8 = memanggil library pickle
- Baris 9 = membuat variable s
- Baris 10 = membuat variable clf2
- Baris 11 = prediksi nilai baru
- Baris 12 = set array

#### 1.6.5 Conventions

- menjelaskan maksud dari tulisan tersebut dan mengartikan per baris.

- Baris 1 = memasukkan dan memanggil numpy sebagai np

```

import numpy as np
from sklearn import random_projection

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

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

```

Figure 1.36: Conventions

- Baris 2 = memasukkan dan memanggil random projection dari sklearn
- Baris 3 = membuat variable rng
- Baris 4 = membuat variable rng
- Baris 5 = membuat variable x
- Baris 6 = pemanggilan variable x
- Baris 7 = pemanggilan dtype
- Baris 8 = membuat variable tranformer
- Baris 9 = membuat variable xnew
- Baris 10 = pemanggilan xnew
- Baris 11 = pemanggilan dtype

### 1.6.6 Penanganan Error

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

1. skrinsut error

```

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

```

Figure 1.37: skrinsut error

2. Tuliskan kode eror dan jenis errornya

- Kode error = NameError: name 'lusia' is not defined
- jenis error = NameError

3. Solusi pemecahan masalah error

```
>>> print('lusia')
lusia
```

Figure 1.38: gb 1

# Chapter 2

## Membangun Model Prediksi

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

### 2.1 Fadila/1164072

#### 2.1.1 Teori

Penyelesaian Tugas Harian 3 ( No. 1-7 )

##### 1. Binary Classification Dan Ilustrasi Gambarnya

- Pengertian Binary Classification / Klasifikasi Biner:

Klasifikasi biner atau binomial merupakan tugas untuk mengklasifikasikan elemen-elemen dari himpunan tertentu ke dalam dua kelompok (memprediksi kelompok mana yang masing-masing dimiliki) berdasarkan aturan klasifikasi.

- Ilustrasi Gambar Binary Classification :

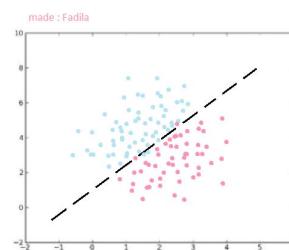


Figure 2.1: binary classification

##### 2. Supervised Learning, Unsupervised Learning, Clustering Dan Ilustrasi Gambar

- Pengertian Supervised Learning :

Sebuah pendekatan dimana terdapat data yang dilatih dan ditargetkan. Lebih singkatnya supervised learning memiliki kategori sehingga tujuan dan outputnya jelas.

- Ilustrasi Gambar Supervised Learning :



Figure 2.2: supervised

Pada contoh gambar diatas dikelompokkan bahwa apabila bentuk dari objek di gambar berbentuk bundar dan berwarna merah maka akan dinamakan atau disebut sebagai Apel. Dan apabila pada gambar terdapat objek berbentuk panjang dan berwarna kuning maka akan dinamakan atau disebut sebagai pisang.

- Pengertian Unsupervised Learning :

Tidak memiliki data latih, sehingga dari data yang tersebut kita bisa mengelompokkannya ke berbagai kelompok 2 seterusnya. Dengan lebih singkatnya ialah unsupervised learning tidak memiliki kategori.

- Ilustrasi Gambar Unsupervised Learning :

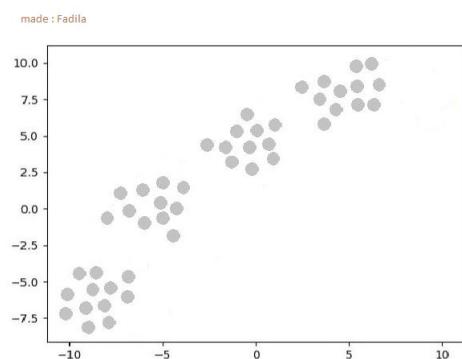


Figure 2.3: unsupervised

Pada gambar dapat dilihat bahwa ada banyak data namun tidak pada pengelompokan yang tepat. Cuman dapat dikelompokkan ke dalam berbagai macam bentuk dan jumlah namun tidak memberikan output yang jelas.

- Pengertian Clustering :

Metode pengelompokan data. Clustering juga merupakan proses partisi satu set objek data ke dalam himpunan bagian yang disebut dengan cluster. Objek dalam cluster tersebut memiliki kemiripan karakteristik antar satu sama lain.

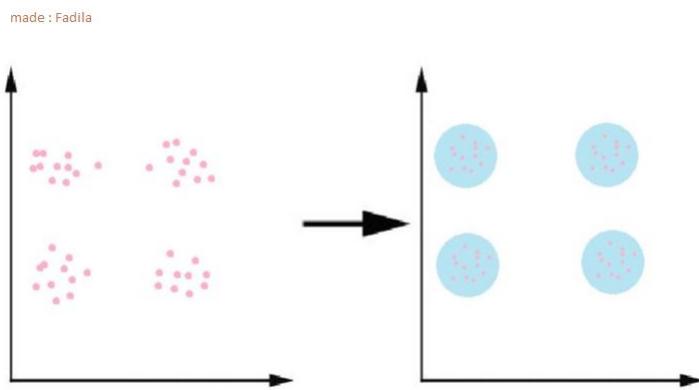


Figure 2.4: clustering

### 3. Evaluasi, Akurasi Dan Ilustrasi Gambar

- Pengertian Evaluasi

Evaluasi digunakan untuk memeriksa/memastikan dan mengevaluasi model dalam bekerja ( seberapa baik ) dengan mengukur keakuratannya. Kita juga dapat menanalis kesalahan yang dibuat pada model yang dijalankan, tingkat kebingungan dan menggunakan matriks kebingungan.

|               | Prediksi "Apel" | Prediksi "Jeruk" |
|---------------|-----------------|------------------|
| Benar "Apel"  | 20              | 5                |
| Benar "Jeruk" | 3               | 22               |
| Made : Fadila |                 |                  |

Figure 2.5: Evaluasi

Pada contoh gambar dapat dilihat bahwa dilakukan evaluasi terhadap kerja dalam penentuan jenis dari objek. Dievaluasi berapa banyak sebuah objek ketika dikelompokkan dan diklasifikasikan kemudian dapat dilihat apakah kerjanya sesuai atau tidak.

- Pengertian Akurasi

Accuracy akan didefinisikan sebagai persentase kasus yang diklasifikasikan dengan benar. Accuracy lebih jelasnya adalah perbandingan kasus yang diidentifikasi benar dengan jumlah semua kasus

Rumus dari accuracy =  $(a+c)/(a+b+c+d)$

Made : Fadila  
**Accuracy / Ketepatan :  $\{ (20 + 22) / 20 + 5 + 3 + 22 \} = 84\%$**

Figure 2.6: Akurasi

Dilakukan perhitungan dengan rumus akurasi terhadap data yang telah diolah pada "Evaluasi". Kemudian didapatkan hasil dari pengolahan data tersebut.

Contoh penggabungan Akurasi Dan Evaluasi

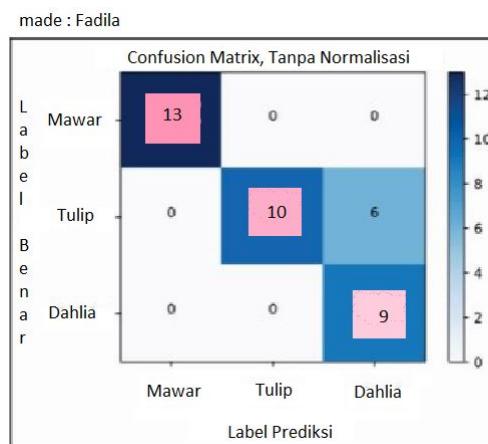


Figure 2.7: Contoh Evaluasi Dan Akurasi Secara Bersamaan

#### 4. Membuat Dan Membaca Confusion Matrix Beserta Contoh

- Pengertian Confusion Matrix

Confusion matrix merupakan suatu metode yang digunakan untuk melakukan perhitungan akurasi pada konsep data mining.

- Pembacaan Confusion Matrix

- Apabila hasil prediksi negatif dan data sebenarnya merupakan negatif.
- Apabila hasil prediksi positif sedangkan nilai sebenarnya merupakan negatif.

(c) Apabila hasil prediksi negatif sedangkan nilai sebenarnya merupakan positif.

(d) Apabila hasil prediksi positif dan nilai sebenarnya merupakan positif.

- Pembuatan Confusion Matrix

(a) Menentukan 4 proses klasifikasi yang akan digunakan dalam confusion matrix.

(b) 4 Istilah tersebut ada True Positive ( TP ), True Negative ( TN ), False Positive ( FP ) dan False Negative ( FN ).

(c) Kelompokkan klasifikasi tersebut bisa menggunakan klasifikasi biner

(d) Akan menghasilkan keluaran berupa 2 Kelas ( Positif dan Negatif ) dan penentuan TP, FP ( 1 klasifikasi positif ), FN dan TN ( 1 klasifikasi negatif ).

(e) Contoh dasarnya nampak seperti langkah diatas

(f) Istilahnya dapat didefinisikan dengan objek lain namun dengan alur yang sama ( sesuai rumus baik klasifikasi dll ).

- Ilustrasi Gambar

| Y | Y Pred | Keluaran Untuk ( threshold ) 0.6 | Recall ( pemanngilan/penarikan ) | Precision ( Presisi ) |
|---|--------|----------------------------------|----------------------------------|-----------------------|
| 0 | 0.5    | 0                                |                                  |                       |
| 1 | 0.9    | 1                                |                                  |                       |
| 0 | 0.7    | 1                                |                                  |                       |
| 1 | 0.7    | 1                                |                                  |                       |
| 1 | 0.3    | 0                                |                                  |                       |
| 0 | 0.4    | 0                                |                                  |                       |
| 1 | 0.5    | 0                                |                                  |                       |

Made : Fadila

Figure 2.8: confusion matrix

– Penjelasan

(a) Recall

Dari semua kelas positif, seberapa banyak yang kami prediksi dengan benar. Itu harus setinggi mungkin.

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}}$$

# made : Fadila

Figure 2.9: recall

(b) Presisi / Precision

Dari semua kelas, seberapa banyak yang kami prediksi dengan benar. Itu harus setinggi mungkin.

$$\text{Precision / Presisi} = \frac{\text{TP}}{\text{TP} + \text{FP}}$$

# made : Fadila

Figure 2.10: precision

(c) F-Ukur ( measure )

Sulit untuk membandingkan dua model dengan presisi rendah dan daya ingat tinggi atau sebaliknya. Jadi untuk membuatnya sebanding, kami menggunakan F-Score. F-score membantu mengukur Recall dan Precision pada saat yang bersamaan

$$\text{F-Measure} = \frac{2 \cdot \text{Recall} \cdot \text{Precision}}{\text{Recall} + \text{Precision}}$$

# made : Fadila

Figure 2.11: f-measure

## 5. Cara Kerja K-Fold Classification Dan Ilustrasi Gambar

- (a) Pertama-tama untuk total instance dibagi menjadi N bagian.
- (b) Fold ke-1 ( atau pertama ) adalah ketika bagian ke-1 menjadi data uji (testing data) dan sisanya menjadi data latih (training data).
- (c) Hitung akurasi ( berdasarkan porsi data tersebut. Persamaanya sebagai berikut :
  - (sigma) data klasifikasi
  - (sigma) total data uji
  - x 100 persen
- (d) Fold ke-2 ( kedua ) adalah ketika bagian ke-2 menjadi data uji (testing data) dan sisanya menjadi data latih (training data).
- (e) Kemudian dihitunglah akurasi berdasarkan porsi data yang telah ditentukan

- (f) Demikian seterusnya hingga mencapai fold ke-K. Hitung rata-rata akurasi dari K buah akurasi di atas. Rata-rata akurasi ini menjadi akurasi final atau akhir.

- Ilustrasi Gambar

| Kategori | $P(c)$        | $P(w_{kj}   c)$ |                |                |                |                |                |                |                |                |
|----------|---------------|-----------------|----------------|----------------|----------------|----------------|----------------|----------------|----------------|----------------|
|          |               | hasilnya        | bagus          | pengen         | ac             | jelek          | banget         | sih            | waw            | Keren          |
| Positif  | $\frac{1}{2}$ | $\frac{2}{27}$  | $\frac{2}{27}$ | $\frac{2}{27}$ | $\frac{1}{27}$ | $\frac{1}{27}$ | $\frac{2}{27}$ | $\frac{1}{27}$ | $\frac{4}{27}$ | $\frac{2}{27}$ |
| Negatif  | $\frac{1}{2}$ | $\frac{1}{27}$  | $\frac{1}{27}$ | $\frac{2}{27}$ | $\frac{2}{27}$ | $\frac{2}{27}$ | $\frac{2}{27}$ | $\frac{1}{27}$ | $\frac{1}{27}$ | $\frac{1}{27}$ |

# made : Fadila

Figure 2.12: k-fold classification 1

| Kategori | $P(c)$        | $P(w_{kj}   c)$ |                |                |                |                |                |                |                |                |
|----------|---------------|-----------------|----------------|----------------|----------------|----------------|----------------|----------------|----------------|----------------|
|          |               | cakep           | esenang        | make           | min            | mangrove       | gamempan       | bohong         | Nih            |                |
| Positif  | $\frac{1}{2}$ | $\frac{2}{27}$  | $\frac{2}{27}$ | $\frac{1}{27}$ |
| Negatif  | $\frac{1}{2}$ | $\frac{1}{27}$  | $\frac{1}{27}$ | $\frac{2}{27}$ |

# made : Fadila

Figure 2.13: k-fold classification 2

## 6. Decision Tree Dan Ilustrasi Gambar

- Pengertian Decision Tree

Decision tree adalah salah satu metode klasifikasi yang paling populer karena mudah diinterpretasikan oleh manusia. Decision tree merupakan metode klasifikasi yang digunakan untuk pengenalan pola dan termasuk dalam pengenalan pola secara statistik. 3 tipe dari decision tree ialah: simpul: simpul root, simpul perantara, dan simpul leaf.

- Ilustrasi Gambar



made : Fadila

Figure 2.14: decision tree

## 7. Information Gain Dan Entropi

- Pengertian Information Gain

Information Gain adalah salah satu attribute selection measure yang digunakan untuk memilih test attribute tiap node pada tree.

Algoritme Information Gain digunakan untuk mengurangi dimensi atribut untuk mendapatkan atribut-atribut yang relevan.

- Ilustrasi Gambar

Tabel 1 Perbandingan Akurasi Kelas Tidak Seimbang dengan 13 fitur dan 6 fitur

| Nilai K | Sebaran Kelas Tidak Seimbang                          |                                                |
|---------|-------------------------------------------------------|------------------------------------------------|
|         | Akurasi Tanpa Menggunakan Information Gain (13 fitur) | Akurasi Menggunakan Information Gain (6 fitur) |
| 5       | 73,08%                                                | 88,46%                                         |
| 15      | 80,77%                                                | 84,62%                                         |
| 25      | 84,62%                                                | 88,46%                                         |
| 35      | 80,77%                                                | 88,46%                                         |
| 45      | 80,77%                                                | 88,46%                                         |
| 55      | 84,62%                                                | 84,62%                                         |
| 65      | 80,77%                                                | 84,62%                                         |
| 75      | 84,62%                                                | 84,62%                                         |
| 85      | 80,77%                                                | 84,62%                                         |
| 95      | 76,92%                                                | 88,46%                                         |

Figure 2.15: informaion gain 1

Tabel 2 Perbandingan Akurasi Kelas Seimbang dengan 13 fitur dan 6 fitur

| Nilai K | Sebaran Kelas Seimbang                                |                                                |
|---------|-------------------------------------------------------|------------------------------------------------|
|         | Akurasi Tanpa Menggunakan Information Gain (13 fitur) | Akurasi Menggunakan Information Gain (6 fitur) |
| 5       | 61,54%                                                | 84,62%                                         |
| 15      | 80,77%                                                | 84,62%                                         |
| 25      | 80,77%                                                | 92,31%                                         |
| 35      | 80,77%                                                | 88,46%                                         |
| 45      | 76,92%                                                | 84,62%                                         |
| 55      | 80,77%                                                | 84,62%                                         |
| 65      | 80,77%                                                | 84,62%                                         |
| 75      | 80,77%                                                | 84,62%                                         |
| 85      | 80,77%                                                | 84,62%                                         |
| 95      | 80,77%                                                | 84,62%                                         |

Figure 2.16: information gain 2

- Penjelasan :

Tabel 1 sampai dengan Tabel 2 menunjukkan bahwa penggunaan seleksi fitur Information Gain menghasilkan nilai akurasi yang lebih baik dibandingkan tanpa menggunakan Information Gain.

Pada saat nilai K sama dengan 5 ( K=5) akurasi yang dihasilkan sistem tanpa menggunakan Information Gain menunjukkan hasil yang kurang

baik pada sebaran kelas seimbang maupun tak seimbang yaitu 61,54 persen pada sebaran kelas seimbang dan 73,08 persen pada sebaran kelas tidak seimbang.

- Pengertian Entropi

Entropi pada umumnya merupakan salah satu besaran yang mengukur energi dalam sistem per satuan temperatur yang tak dapat digunakan untuk melakukan usaha.

Namun, secara spesifik untuk ” Entropi ” sendiri merupakan parameter untuk mengukur tingkat keberagaman (heterogenitas) dari kumpulan data.

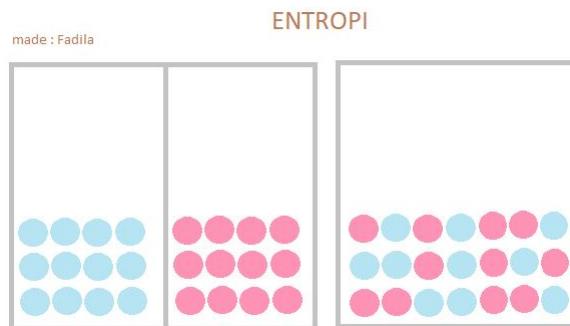


Figure 2.17: entropi

### 2.1.2 Praktek Scikit-Learn

Penyelesaian Tugas Harian 4 ( No. 1-12 )

(a) Pembahasan Codingan Dan Hasilnya

i. Codingan Pertama :

Penjelasan : Codingan pertama ini akan meload ( menampilkan ) data pada file yang ditentukan. Untuk codingan ini file yang dieksekusi ialah ” student-mat.csv ” . Secara jelasnya, dalam codingan dapat dilihat bahwa variabel bakso didefinisikan untuk pembacaan csv dari ” pizza ” dimana untuk pemisahnya yaitu separation berupa ; . Setelah itu variabel bakso di ” print ” dengan perintah menampilkan ” len ” panjang ataupun jumlah dan hasilnya berupa angka 395 .

- Hasil Codingan Pertama :

```
In [56]: import pandas as pizza
...: bakso = pizza.read_csv('dataset/student-mat.csv', sep=';')
...: len(bakso)
Out[56]: 395
```

Figure 2.18: codingan pertama

ii. Codingan Kedua :

Penjelasan : Codingan kedua ini secara keseluruhan menampilkan baris G1, G2 dan G3 ( berdasarkan kriterianya ) untuk kolom PASS pada variabel bakso. Untuk lebih jelasnya, pada codingan terdapat pendefinisian pembacaan "lamda" ( panjang gelombang ) dari baris G1, G2 dan G3. Apabila row-row tersebut bernilai lebih dari 35 maka akan terdefinisikan angka "1" apabila tidak, maka akan terdefinisikan angka "0" pada kolom PASS ( sesuai permintaan awal ). Selanjutnya variabelnya di "print" sehingga menampilkan keluaran. Tidak lupa terdapat juga jumlah dari baris dan kolom yang terubah sesuai dengan baris yang dieksekusi.

- Hasil Codingan Kedua :

```
In [60]: bakso['pass'] = bakso.apply(lambda row: 1 if (row['G1']+row['G2']+row['G3']) >= 35 else 0, axis=1)
...: bakso = bakso.drop(['G1', 'G2', 'G3'], axis=1)
...: bakso.head()
Out[60]:
 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 2.19: codingan kedua

iii. Codingan Ketiga :

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

- Hasil Codingan Ketiga :

```

In [61]: bakso = pizza.get_dummies(bakso, columns=['sex', 'school', 'address', 'famsize', 'Pstatus', 'Mjob', 'Fjob',
...: 'reason', 'guardian', 'schoolsup', 'famsup', 'paid', 'activities',
...: 'nursery', 'higher', 'internet', 'romantic'])
...: bakso.head()
Out[61]:
 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 2.20: codingan ketiga

#### iv. Codingan Keempat :

Penjelasan : Secara keseluruhan codingan ini difungsikan untuk mendefinisikan pembagian data yang berupa training dan testing data. Secara jelasnya pertama-tama variabel bakso akan mendefinisikan sampel yang akan digunakan ( berupa shuffle row ) . Nah kemudian masin2 parameter yaitu bakso train dan bakso test akan berjumlah 500 data ( telah dibagi untuk training dan testing ). Selanjutnya dilakukan pengeksekusian untuk kolom Pass, apabila sesuai dengan "axis=1" maka eksekusi fungsi berhasil. Selain itu juga disertakan jumlah dari peserta yang lolos dari semua nilai data setnya.

- Hasil Codingan Keempat :

```

In [62]: bakso = bakso.sample(frac=1)
...: # split training and testing data
...: bakso_train = bakso[:500]
...: bakso_test = bakso[500:]
...:
...: bakso_train_att = bakso_train.drop(['pass'], axis=1)
...: bakso_train_pass = bakso_train['pass']
...:
...: bakso_test_att = bakso_test.drop(['pass'], axis=1)
...: bakso_test_pass = bakso_test['pass']
...:
...: bakso_att = bakso.drop(['pass'], axis=1)
...: bakso_pass = bakso['pass']
...:
...: # number of passing students in whole dataset:
...: import numpy as np
...: print("Passing: %d out of %d (%.2f%%)" % (np.sum(bakso_pass), len(bakso_pass),
100*float(np.sum(bakso_pass)) / len(bakso_pass)))
Passing: 166 out of 395 (42.03%)

```

Figure 2.21: codingan keempat

#### v. Codingan Kelima :

Penjelasan : Secara keseluruhan, codingan ini hanya membuktikan pengujian dari Klasifikasi Decision Tree yang ada, apakah true atau tidak dan hasilnya true. Apabila dibahas secara lengkap maka pada codingan ini di definisikan library sklearn untuk mengimpor atau menampilkan tree. Variabel sate difungsikan untuk membaca klasifikasi decision tree dari tree itu sendiri dengan 2 parameternya yaitu kriteria="entropy" dan max depth=5. Maka selanjutnya variabel sate akan masuk dan terbaca dalam module fit dengan 2 parameter yaitu bakso trai att dan bakso train pass.

- Hasil Codingan Kelima :

```
In [63]: from sklearn import tree
...: sate = tree.DecisionTreeClassifier(criterion="entropy", max_depth=5)
...: sate = sate.fit(bakso_train_att, bakso_train_pass)
```

Figure 2.22: codingan kelima

vi. Codingan Keenam :

Penjelasan : Codingan ini memberikan gambaran dari klasifikasi decision tree dari pengolahan parameter yang dieksekusi kedalam variabel sate. Tentunya dengan pemanfaatan library graphviz yang telah diimport dan difungsikan.

Untuk gambarnya terlalu besar ketika di Console sehingga hanya sebagian yang bisa di screenshoot untuk dijadikan hasil.

- Hasil Codingan Keenam :

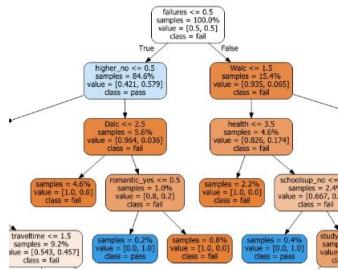


Figure 2.23: codingan keenam

vii. Codingan Ketujuh :

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

- Hasil Codingan Ketujuh :

```
In [66]: tree.export_graphviz(sate, out_file="student-performance.dot", label="all", impurity=False,
proportion=True,
...: feature_names=list(bakso_train_att), class_names=["fail", "pass"],
...: filled=True, rounded=True)
```

Figure 2.24: codingan ketujuh

viii. Codingan Kedelapan :

Penjelasan : Secara keseluruhan, codingan ini membaca score dari variabel sate dimana terdapat 2 parameter yang dihitung dan diuji yaitu bakso test att dan bakso test pass. Untuk hasilnya sendiri mengapa berupa angka, dikarenakan pada parameter yang dieksekusi memang memiliki data sehingga dieksekusi dan menghasilkan keluaran dari score tersebut.

- Hasil Codingan Kedelapan :

```
In [33]: sate.score(bakso_test_att, bakso_test_pass)
Out[33]: 0.6308724832214765
```

Figure 2.25: codingan kedelapan

ix. Codingan Kesembilan :

Penjelasan : Secara keseluruhan, codingan ini membahas mengenai pengkesekusian fungsi dan variabel dari library yang didefinisikan dan yang diimport. Penjelasan lebih jelasnya ialah codingan ini mendefinisikan library sklearn.model.selection kemudian mengimpor cross val score. Kemudian variabel score mendefinisikan cross val score yang telah diimport tadi dengan 4 parameter yaitu sate, bakso att, bakso pass dan cv=5 untuk dieksekusi. Setelah semua pemrosesan tersebut maka hasil yang di "print" ialah rata2 perhitungan dari variabel score dimana dan standar dari (+/-) tentunya dengan ketentuan parameter Accuracy .

- Hasil Codingan Kesembilan :

```
In [69]: from sklearn.model_selection import cross_val_score
...: scores = cross_val_score(sate, bakso_att, bakso_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 2.26: codingan kesembilan

x. Codingan Ke-10 :

Penjelasan : Codingan ini mendefinisikan max depth dalam jarak angka antara parameter 1 dan 20. Variabel sate mendefinisikan klas-

fier decision tree dengan 2 parameter. Kemudian variabel score mengeksusi parameter lainnya yaitu seperti sate, bakso att, bakso pass dan  $cv=5$  . Hasil yang ditampilkan ialah dari max depth, accuracy dan  $(+/-)$  dan akhirnya hasilnya nampak seperti pada gambar.

- Hasil Codingan Ke-10 :

```
In [70]: for max_depth in range(1, 20):
 ...:
 state = tree.DecisionTreeClassifier(criterion='entropy', max_depth=max_depth)
 state.fit(X_train, y_train)
 state.score(X_test, y_test)
 print("Max depth: %d, Accuracy: %.2f (%-.2f)" % (max_depth, scores.mean()), scores.std()) * 2
```

Figure 2.27: codingan ke-10

### xi. Codingan Ke-11 :

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

- Hasil Codingan Ke-11 :

```
In [5]: fadila = np.empty((19,3), float)
.... i = 0
.... for max_depth in range(1,20):
.... tree = tree.DecisionTreeClassifier(criterion='entropy',
.... max_depth=max_depth)
.... scores = cross_val_score(tree, bakso_1st, bakso_pass, cv=5)
.... fadila[i,0] = scores.mean()
.... fadila[i,1] = scores.std()**2
.... i = i + 1
....
.... fadila
Out[5]: array([[1.00000000e+00, 5.7751784e-01, 6.3076858e-01],
 [2.00000000e+00, 4.9644267e-01, 1.1165462e-01],
 [3.00000000e+00, 4.7289959e-01, 9.8738567e-02],
 [4.00000000e+00, 4.6000000e+00, 1.0899999e-01],
 [5.00000000e+00, 4.5500000e+00, 1.0899999e-01],
 [6.00000000e+00, 4.7456244e-01, 8.4362271e-01],
 [7.00000000e+00, 4.6500000e+00, 1.0899999e-01],
 [8.00000000e+00, 5.5811273e-01, 1.0825254e-01],
 [9.00000000e+00, 4.4457349e-01, 8.6731064e-01],
 [1.00000000e+01, 4.4457349e-01, 8.6731064e-01],
 [1.10000000e+01, 4.5202447e-01, 1.3516115e-01],
 [1.20000000e+01, 5.4590521e-01, 1.2381284e-01],
 [1.30000000e+01, 5.4590521e-01, 1.2381284e-01],
 [1.40000000e+01, 5.1598521e-01, 9.1794871e-02],
 [1.50000000e+01, 4.4868866e-01, 8.4572327e-02],
 [1.60000000e+01, 4.4454446e-01, 8.4572327e-02],
 [1.70000000e+01, 4.4454446e-01, 9.1899999e-02],
 [1.80000000e+01, 4.1212121e-01, 1.2267772e-01],
 [1.90000000e+01, 4.5955552e-01, 1.2267772e-01],
 [2.00000000e+01, 4.5955552e-01, 1.2267772e-01]])
```

Figure 2.28: codingan ke-11

### xii. Codingan Ke-12 :

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

- Hasil Codingan Ke-12 :

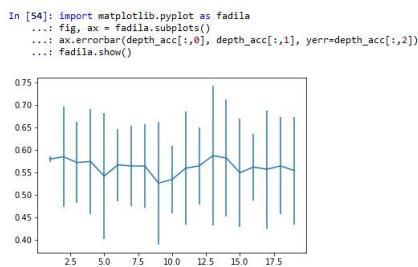


Figure 2.29: codingan ke-12

### 2.1.3 Penanganan Error

#### Pembahasan dan Penyelesaian Error

- (a) Error 1 :

```
File "pandas_libs\parsers.pyx", line 384, in
pandas._libs.parsers.TextReader._cinit_
File "pandas_libs\parsers.pyx", line 695, in
pandas._libs.parsers.TextReader._setup_parser_source
FileNotFoundError: File b'student-mat.csv' does not exist
```

Figure 2.30: error 1

- Penjelasan :

- Pada error tersebut dikatakan bahwa untuk file -b"student-mat.csv" tidak ada. Mengapa? karena file codingan yang dieksekusi yaitu student-performance.py tidak berada pada folder yang sama jadi tidak dapat terpanggil dan tereksekusi.
- Nah, untuk penyelesaiannya yaitu dengan menambahkan fungsi yang mendefinisikan folder tempat file "student-mat.csv" berada.
- Silahkan tambahkan perintah " dataset/student-mat.csv " pada codingannya

- Dengan penambahan perintah tersebut maka tidak akan terjadi error lagi.

(b) Error 2 :

```
In [65]: import graphviz
...: dot_data = tree.export_graphviz(sate, out_file=None, label="all",
...: impurity=False, proportion=True,
...: feature_names=list(bakso_train_att),
...: class_names=["fail", "pass"],
...: rounded=True, filled=True)
...: graph = graphviz.Source(dot_data)
...: graph
Traceback (most recent call last):
File "<ipython-input-65-583fa3dfc87a>", line 1, in <module>
 import graphviz
ModuleNotFoundError: No module named 'graphviz'
```

Figure 2.31: error 2

• Penjelasan :

- Pada error tersebut dikatakan bahwa tidak terdapat module graphviz sehingga codingan tidak dapat dieksekusi.
- Penanganannya yaitu dengan melakukan penginstalan module graphviz itu sendiri
- Penginstalan bisa dilakukan pada Anaconda Prompt ataupun Command Prompt
- Pada prompt tersebut masukkan perintah ” conda install graphviz ”
- Silahkan tunggu sampai instalasi berhasil
- Setelah selesai, maka cobalah RUN kembali codingan terkait
- Maka tidak akan terjadi error lagi.

(c) Error 3 :

```
In [86]: import graphviz
...: dot_data = tree.export_graphviz(bandung, out_file=None, label="all",
...: impurity=False, proportion=True,
...: feature_names=list(jakarta_train_att),
...: class_names=["fail", "pass"],
...: rounded=True, filled=True)
...: graph = graphviz.Source(dot_data)
...: graph
graphERROR: execution aborted
```

Figure 2.32: error 3

• Penjelasan :

- Pada error tersebut dikatakan bahwa eksekusinya dilarang atau di aborted.
- Penanganannya sebenarnya sederhana
- Tidak terdapat kesalahan dalam pembuatan dan penyesuaian codingan

- Mungkin saja terjadi error tersebut karena pengeksekusian fungsi yang berulang-ulang pada codingan yang berbeda
- Silahkan di Restart atau dijalankan kembali aplikasi spydernya
- Run kembali maka errornya tidak akan muncul lagi dan hasilnya akan muncul seperti pada gambar pengujian no 6

(d) Error 4 :

```
File "C:\ProgramData\Anaconda\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\Anaconda\lib\site-packages\sklearn\utils\validation.py", line 582, in check_array
 context)
ValueError: unsupported format character 'b' (0x62) at index 18
```

Figure 2.33: error 4

- Penjelasan :

- Pada error tersebut dikatakan terdapat error karakter b nya gak kebaca
- Penanganannya sebenarnya sederhana, dari codingan tidak ada yang salah
- Disarankan agar mengulang kembali penamaan untuk variabel terkait dari awal codingan hingga akhir sehingga lebih teratur
- Silahkan dicoba run codingan tersebut kembali
- Maka codingannya akan berhasil dan muncul seperti pada contoh codingan no 8 diatas

## 2.2 Lusia Violita Aprilian

### 2.2.1 Teori

#### 1. binary classification dilengkapi ilustrasi gambar

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

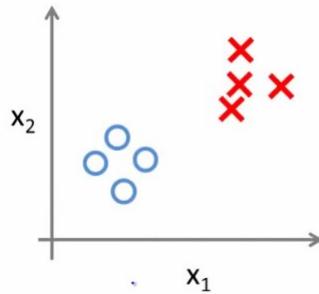


Figure 2.34: Binary Classification

2. supervised learning dan unsupervised learning dan clustering dengan ilustrasi gambar

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

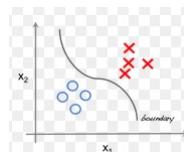


Figure 2.35: Supervised Learning

(b) Unsupervised learning adalah istilah yang digunakan untuk pembelajaran bahasa Ibrani, yang terkait dengan pembelajaran tanpa guru, juga dikenal sebagai organisasi mandiri dan metode pemodelan kepadatan probabilitas input. Analisis cluster sebagai cabang pembelajaran mesin yang

mengelompokkan data yang belum diberi label, diklasifikasikan atau dikategorikan. Alih-alih menanggapi umpan balik, analisis klaster mengidentifikasi kesamaan dalam data dan bereaksi berdasarkan ada tidaknya kesamaan di setiap potongan data baru.

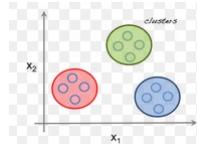


Figure 2.36: Unsupervised Learning

- (c) Cluster analysis or clustering adalah tugas pengelompokan sekumpulan objek sedemikian rupa sehingga objek dalam kelompok yang sama (disebut klaster) lebih mirip (dalam beberapa hal) satu sama lain daripada pada kelompok lain (kluster). Ini adalah tugas utama penambangan data eksplorasi, dan teknik umum untuk analisis data statistik, yang digunakan di banyak bidang, termasuk pembelajaran mesin, pengenalan pola, analisis gambar, pengambilan informasi, bioinformatika, kompresi data, dan grafik komputer. Analisis Cluster sendiri bukan merupakan salah satu algoritma spesifik, tetapi tugas umum yang harus dipecahkan. Ini dapat dicapai dengan berbagai algoritma yang berbeda secara signifikan dalam pemahaman mereka tentang apa yang merupakan sebuah cluster dan bagaimana cara menemukannya secara efisien. Gagasan populer mengenai cluster termasuk kelompok dengan jarak kecil antara anggota cluster, area padat ruang data, interval atau distribusi statistik tertentu. Clustering karena itu dapat dirumuskan sebagai masalah optimasi multi-objektif. Algoritma pengelompokan dan pengaturan parameter yang sesuai (termasuk parameter seperti fungsi jarak yang akan digunakan, ambang kepadatan atau jumlah cluster yang diharapkan) tergantung pada set data individual dan penggunaan hasil yang dimaksudkan. Analisis kluster bukan merupakan tugas otomatis, tetapi proses berulang penemuan pengetahuan atau optimasi multi-objektif interaktif yang melibatkan percobaan dan kegagalan. Seringkali diperlukan untuk memodifikasi praproses data dan parameter model hingga hasilnya mencapai properti yang diinginkan.

3. evaluasi dan akurasi dari buku dan disertai ilustrasi contoh dengan gambar

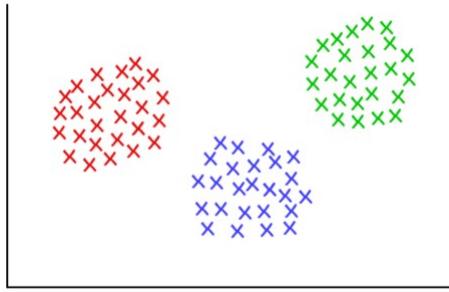


Figure 2.37: Cluster

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

|               | Predicted "apple" | Predicted "orange" |
|---------------|-------------------|--------------------|
| True "apple"  | 20                | 5                  |
| True "orange" | 3                 | 22                 |

Figure 2.38: Evaluasi dan Akurasi

4. bagaimana cara membuat dan membaca confusion matrix, buat confusion matrix
  - (a) Cara membuat dan membaca confusion matrix :
    - i. Tentukan pokok permasalahan dan atributanya, misal gaji dan listik.
    - ii. Buat pohon keputusan
    - iii. Lalu data testingnya
    - iv. Lalu mencari nilai a, b, c, dan d. Semisal a = 5, b = 1, c = 1, dan d = 3.
    - v. Selanjutnya mencari nilai recall, precision, accuracy, serta dan error rate.
  - (b) 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 =  $(1+1)/(5+1+1+3) = 0,2$

5. bagaimana K-fold cross validation bekerja dengan gambar ilustrasi

Cara kerja K-fold cross validation :

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



Figure 2.39: K-fold cross validation

6. decision tree dengan gambar ilustrasi

Decision tree adalah model visual yang terdiri dari node dan cabang, seperti Gambar dijelaskan secara rinci nanti dalam artikel ini. Untuk saat ini, amati bahwa ia tumbuh dari kiri ke kanan, dimulai dengan simpul keputusan root (kuadrat, juga disebut simpul pilihan) yang cabang-cabangnya mewakili dua atau lebih opsi bersaing yang tersedia bagi para pembuat keputusan. Pada akhir cabang awal ini, ada simpul akhir (segitiga, juga disebut simpul nilai) atau simpul ketidakpastian (lingkaran, juga disebut simpul peluang). Node

akhir mewakili nilai tetap. Cabang lingkaran mewakili hasil yang mungkin bersama dengan probabilitasnya masing-masing (yang berjumlah 1,0). Di luar cabang-cabang node ketidakpastian awal ini, mungkin ada lebih banyak bujur sangkar dan lebih banyak lingkaran, yang umumnya bergantian sampai setiap jalur berakhir di simpul akhir.

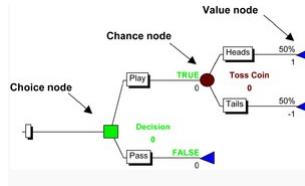


Figure 2.40: Decision Tree

## 7. Information gain dan entropi dengan gambar ilustrasi

- (a) Information gain (IG) mengukur seberapa banyak informasi fitur memberi kita tentang kelas. - Fitur yang sempurna mempartisi harus memberikan informasi maksimal. - Fitur yang tidak terkait seharusnya tidak memberikan informasi.

| Feature         | Explanation                                | IG Rank |
|-----------------|--------------------------------------------|---------|
| <i>is.obese</i> | The subject is obese (yes, no)             | 0.0203  |
| <i>whr</i>      | Waist hip ratio                            | 0       |
| <i>hc</i>       | Hip circumference (cm)                     | 0       |
| <i>bmi</i>      | Body mass index ( $\text{kg}/\text{m}^2$ ) | 0       |
| <i>wc</i>       | Waist circumference (cm)                   | 0       |
| <i>age</i>      | Age (years)                                | 0       |
| <i>id</i>       | Subject ID                                 | 0       |

Figure 2.41: Information gain

- (b) Entropi merupakan kemurnian dalam koleksi contoh yang sewenang-wenang.

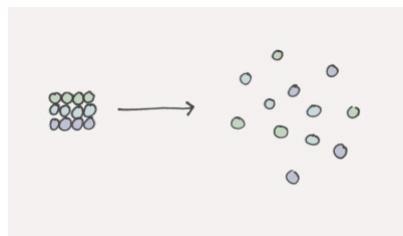


Figure 2.42: Entropi

```
In [5]: import pandas as pd
...: nanas = pd.read_csv('student-mat.csv', sep=';')
...: len(nanas)
Out[5]: 395
```

Figure 2.43: load dataset

## 2.2.2 scikit-learn

### 1. load dataset

Pada gambar tersebut dijelaskan bahwa in dengan mengimport pandas dengan diperumpamakan sebagai pd. Pada baris ke dua, dibuat variable nanas untuk memanggil pd sehingga dapat membaca file csv. Dan pada baris ke 3 variable dipanggil. Sehingga menghasilkan output 395.

### 2. generate binary label (pass/fail) based on G1+G2+G3

```
In [6]: nanas['pass'] = nanas.apply(lambda row: 1 if (row['G1']+row['G2']+row['G3']) >= 35 else 0, axis=1)
...: nanas = nanas.drop(['G1', 'G2', 'G3'], axis=1)
...: nanas.head()
Out[6]:
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 2.44: generate binary label

Pada gambar tersebut dijelaskan bahwa in pada baris pertama, dibuat variable nanas untuk memanggil nanas dengan apply lamda sehingga baris selanjutnya nanas meng-drop G1, G2, dan G3. Dan pada baris ke 3 variable dipanggil. Sehingga menghasilkan output sebuah data 5 rows x 31 columns.

### 3. use one-hot encoding on categorical columns

```
In [7]: nanas = pd.get_dummies(nanas, columns=['sex', 'school', 'address', 'famsize',
'Pstatus', 'Mjob', 'Fjob',
...,
'paid', 'activities',
...,
...: nanas.head())
Out[7]:
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]
In [8]:
```

Figure 2.45: use one-hot encoding on categorical columns

Pada gambar tersebut dijelaskan bahwa in pada baris pertama, dibuat variable nanas untuk memanggil pd dengan get dummies nanas columns. Dan pada baris selanjutnya variable dipanggil. Sehingga menghasilkan output sebuah data 5 rows x 57 columns.

#### 4. shuffle rows, split training dan testing data

```
In [8]: nanas = nanas.sample(frac=1)
...: # split training and testing data
...: nanas_train = nanas[:500]
...: nanas_test = nanas[500:]
...:
...: nanas_train_att = nanas_train.drop(['pass'], axis=1)
...: nanas_train_pass = nanas_train['pass']
...:
...: nanas_test_att = nanas_test.drop(['pass'], axis=1)
...: nanas_test_pass = nanas_test['pass']
...:
...: nanas_att = nanas.drop(['pass'], axis=1)
...: nanas_pass = nanas['pass']
...:
...: # number of passing students in whole dataset:
...: import numpy as np
...: print("Passing: %d out of %d (%.2f%%)" % (np.sum(nanas_pass), len(nanas_pass),
100*float(np.sum(nanas_pass)) / len(nanas_pass)))
Passing: 166 out of 395 (42.03%)
```

Figure 2.46: shuffle rows, split training dan testing data

Pada gambar tersebut dijelaskan bahwa in pada baris pertama, dibuat variable nanas untuk memanggil nanas sebagai sample. Lalu pada baris selanjutnya dilakukan split training dan testing data. Setelah data dilakukan split training da testing, import file numpy sebagai np dan kemudian melakukan prerintah print. Sehingga menghasilkan passing: 116 out of 395.

#### 5. fit a decision tree

```
In [11]: from sklearn import tree
...: duku = tree.DecisionTreeClassifier(criterion="entropy", max_depth=5)
...: duku = duku.fit(nanas_train_att, nanas_train_pass)
In [12]:
```

Figure 2.47: fit a decision tree

Pada gambar tersebut dijelaskan bahwa in pada baris pertama, import tree dari sklearn. Pada baris selanjutnya variable duku memanggil tree dengan decision tree clasifier dengan criteria entropy dan max dept = 5. Dan pada baris terakhir, variable duku memanggil duku dengan fit nanas train att dan nanas train pass.

#### 6. visualize tree

Pada gambar tersebut menjelaskan import graphviz dengan variabelm dot data, class name, class name, dan graph. Sehingga menampilkan gambar sebagai berikut :

```
In [2]: import graphviz
...: dot_data = tree.export_graphviz(duku, out_file=None, label="all",
...: proportion=True,
...: feature_names=list(nanas_train_att),
...: class_names=["fail", "pass"],
...: filled=True, rounded=True)
...: graph = graphviz.Source(dot_data)
...: graph
```

Figure 2.48: visualize tree

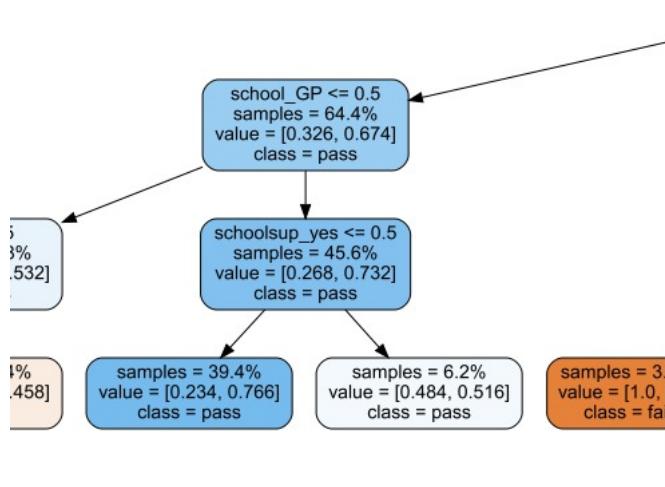


Figure 2.49: visualize tree

## 7. save tree

```
In [15]: tree.export_graphviz(duku, out_file="student-performance.dot", label="all",
...: impurity=False, proportion=True,
...: feature_names=list(nanas_train_att),
...: class_names=["fail", "pass"],
...: filled=True, rounded=True)
```

Figure 2.50: save tree

Pada gambar tersebut dijelaskan bahwa in pada baris pertama, Tree export graphviz duku dengan out file student-performance.dot dan label all. pada baris selanjutnya dijelaskan featura names = list dan class name fail atau pas. Apabila bernilai true akan filled dan rounded.

## 8. t.score

```
In [101]: duku.score(nanas_test_att, nanas_test_pass)
```

Figure 2.51: t.score

Pada gambar tersebut dijelaskan bahwa in pada baris pertama, parameter dengan hasilnya adalah duku.score yang dimana terdapat perhitungan nanas test at dan nana test pass.

## 9. show average score and +/- two standard deviations away

```
In [14]: from sklearn.model_selection import cross_val_score
...: scores = cross_val_score(duku, nanas_att, nanas_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: 0.59 (+/- 0.10)
In [15]:
```

Figure 2.52: show average score

Pada gambar tersebut dijelaskan bahwa in pada baris pertama memanggil cross vla score dari sklearn.model selection. Pada baris selanjutnya membuat variabel scores. Dan pada baris terakhir score akan ditampilkan.

## 10. for max depth in range

```
In [16]: for max_depth in range(1, 20):
...: duku = tree.DecisionTreeClassifier(criterion="entropy",
...: max_depth=max_depth)
...: scores = cross_val_score(duku, nanas_att, nanas_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 (+/- 0.03)
Max depth: 3, Accuracy: 0.61 (+/- 0.11)
Max depth: 4, Accuracy: 0.60 (+/- 0.10)
Max depth: 5, Accuracy: 0.59 (+/- 0.08)
Max depth: 6, Accuracy: 0.60 (+/- 0.10)
Max depth: 7, Accuracy: 0.60 (+/- 0.08)
Max depth: 8, Accuracy: 0.55 (+/- 0.06)
Max depth: 9, Accuracy: 0.56 (+/- 0.04)
Max depth: 10, Accuracy: 0.57 (+/- 0.06)
Max depth: 11, Accuracy: 0.59 (+/- 0.06)
Max depth: 12, Accuracy: 0.57 (+/- 0.09)
Max depth: 13, Accuracy: 0.54 (+/- 0.13)
Max depth: 14, Accuracy: 0.57 (+/- 0.07)
Max depth: 15, Accuracy: 0.58 (+/- 0.09)
Max depth: 16, Accuracy: 0.55 (+/- 0.12)
Max depth: 17, Accuracy: 0.55 (+/- 0.09)
Max depth: 18, Accuracy: 0.56 (+/- 0.06)
Max depth: 19, Accuracy: 0.58 (+/- 0.06)
In [17]:
```

Figure 2.53: for max depth in range

Pada gambar tersebut dijelaskan bahwa in pada baris pertama mendefinisikan range dari 1 hingga 20. Lalu, variabel duku memdefinisikan tree dengan decision tree clasification kriteria entropy. Dan variable scores mendefinisikan duku, nanas att, nanas pass dan cv = 5. Baris terakhir adalah menampilkan hasil.

## 11. depth acc

Pada gambar tersebut dijelaskan bahwa variabel depth acc mendefinisikan np empty dengan i = 0 dari range 1-20. Dimana pada range tersebut terdapat variabel duku dan score. Variabel duku memdefinisikan tree dengan decision tree clasification kriteria entropy. Dan variable scores mendefinisikan duku, nanas att, nanas pass dan cv = 5. Dan pada baris selanjutnya adalah pendefinisian depth acc. Sehingga didapatkan hasil berupa array.

```

In [88]: depth_acc = np.empty((19,3), float)
...: i = 0
...: for max_depth in range(1, 20):
...: duku = tree.DecisionTreeClassifier(criterion="entropy", max_depth=max_depth)
...: scores = cross_val_score(duku, nanas_att, nanas_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[88]:
array([[1.00000000e+00, 5.7975170e-01, 6.3076859e-03],
 [2.00000000e+00, 5.7981499e-01, 4.59786402e-02],
 [3.00000000e+00, 5.72091853e-01, 3.22002693e-02],
 [4.00000000e+00, 5.79881532e-01, 8.26691511e-02],
 [5.00000000e+00, 5.87665533e-01, 1.00629211e-01],
 [6.00000000e+00, 5.75006491e-01, 1.06656313e-01],
 [7.00000000e+00, 5.80039760e-01, 8.19527208e-02],
 [8.00000000e+00, 5.90197988e-01, 9.29887851e-02],
 [9.00000000e+00, 5.90037326e-01, 9.46882094e-02],
 [1.00000000e+01, 5.87568160e-01, 1.13990705e-01],
 [1.10000000e+01, 5.77473223e-01, 1.11095161e-01],
 [1.20000000e+01, 5.77507303e-01, 8.03612112e-02],
 [1.30000000e+01, 6.02601426e-01, 4.38647624e-02],
 [1.40000000e+01, 5.95037326e-01, 5.40261640e-02],
 [1.50000000e+01, 5.77752515e-01, 5.74778007e-02]]).

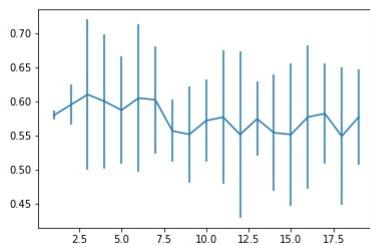
```

Figure 2.54: depth acc

```

In [18]: import matplotlib.pyplot as plt
...: fig, ax = plt.subplots()
...: ax.errorbar(depth_acc[:,0], depth_acc[:,1], yerr=depth_acc[:,2])
...: plt.show()

```



```
In [19]:
```

Figure 2.55: matplotlib

## 12. matplotlib

Pada gambar tersebut dijelaskan bahwa in pada baris pertama mendefinisikan pemanggilan matplotlib.pyplot sebagai plt. Lalu pendefinisian figure dari plt serta error bar. Dan pada baris terakhir menampilkan plt. Maka akan muncul sebuah gambar berupa giagram.

### 2.2.3 Penanganan Error

#### 1. Skrinsut error

```
File "pandas_libs\parsers.pyx", line 384, in
pandas._libs.parsers.TextReader.__cinit__
File "pandas_libs\parsers.pyx", line 695, in
pandas._libs.parsers.TextReader._setup_parser_source
FileNotFoundError: File b'student-por.csv' does not exist

TAE
```

Figure 2.56: error

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

Kode dan jenis error yang didapatkan adalah :

- Kode error = FileNotFoundError: File b'student-por.csv' does not exist
- Jenis error = FileNotFoundError

#### 3. Solusi pemecahan masalah error tersebut

Berikut adalah solusi dari permasalahan pada no. 1

```
In [5]: import pandas as pd
...: nanas = pd.read_csv('student-mat.csv', sep=',')
...: len(nanas)
Out[5]: 395
```

Figure 2.57: penyelesaian

## 2.3 Rahmi Roza/1164085

### 2.3.1 Teori

Penyelesaian Tugas Harian 3 ( No. 1-7 )

#### 1. Binary Classification Dan Ilustrasi Gambarnya

- Pengertian Binary Classification / Klasifikasi Biner:

Klasifikasi biner atau binomial adalah tugas mengklasifikasikan elemen-elemen dari himpunan yang diberikan ke dalam dua kelompok (memprediksi kelompok mana yang masing-masing dimiliki) berdasarkan aturan klasifikasi.

- Ilustrasi Gambar Binary Classification :

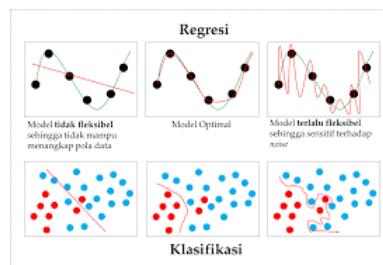


Figure 2.58: binary classification

- Supervised Learning, Unsupervised Learning, Clustering Dan Ilustrasi Gambar
  - Pengertian Supervised Learning :

Supervised learning adalah 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.
- Ilustrasi Gambar Supervised Learning :

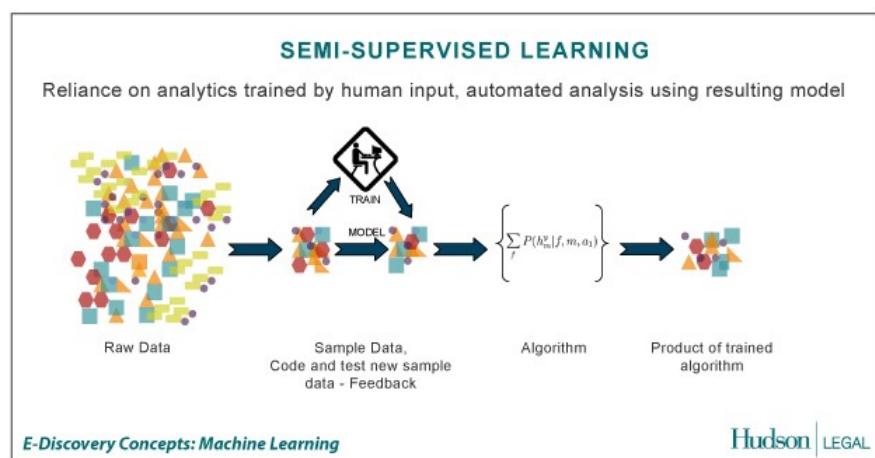


Figure 2.59: supervised

- Pengertian Unsupervised Learning :

unsupervised learning tidak memiliki data latih, sehingga dari data yang ada, kita mengelompokan data tersebut menjadi 2 bagian atau 3 bagian dan seterusnya.

- Ilustrasi Gambar Unsupervised Learning :

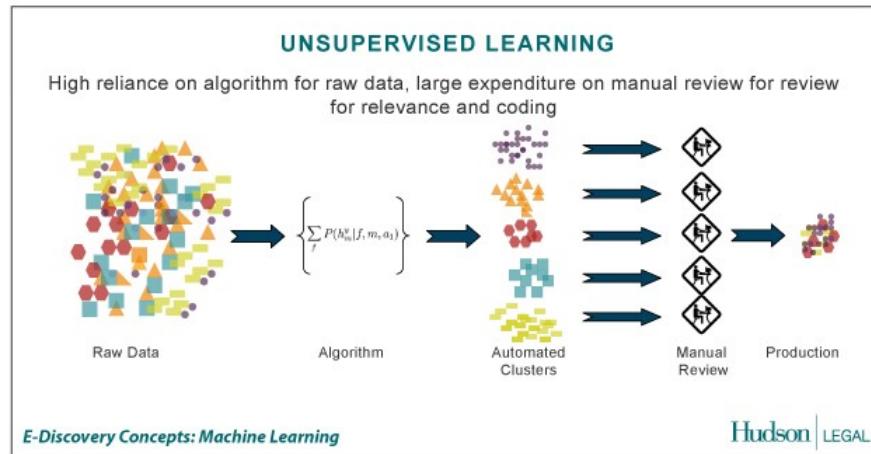


Figure 2.60: unsupervised

- Pengertian Clustering :

Metode pengelompokan data. Clustering juga merupakan proses partisi satu set objek data ke dalam himpunan bagian yang disebut dengan cluster. Objek dalam cluster tersebut memiliki kemiripan karakteristik antar satu sama lain.

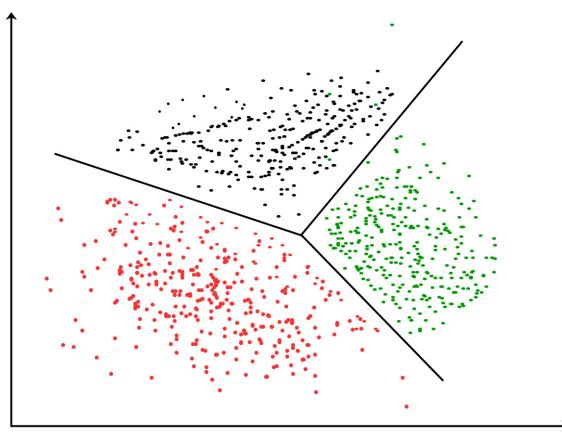


Figure 2.61: clustering

## 2. Evaluasi, Akurasi Dan Ilustrasi Gambar

- Pengertian Evaluasi

Evaluasi digunakan untuk memeriksa/memastikan dan mengevaluasi model dalam bekerja ( seberapa baik ) dengan mengukur keakuratannya. Kita juga dapat menanalis kesalahan yang dibuat pada model yang dijalankan, tingkat kebingungan dan menggunakan matriks kebingungan.

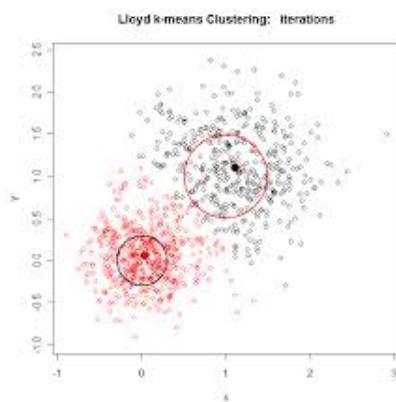


Figure 2.62: Evaluasi

- Pengertian Akurasi

Accuracy akan didefinisikan sebagai presentasi kasus yang diklasifikasikan dengan benar. Accuracy lebih jelasnya adalah perbandingan kasus yang diidentifikasi benar dengan jumlah semua kasus

Rumus dari accuracy=  $(a+c)/(a+b+c+d)$

Made : Fadila  
Accuracy / Ketepatan :  $((20 + 22) / 20 + 5 + 3 + 22) = 84$

Figure 2.63: Akurasi

Dilakukan perhitungan dengan rumus akurasi terhadap data yang telah diolah pada " Evaluasi ". Kemudian di dapatkan hasil dari pengolahan data tersebut.

Contoh penggabungan Akurasi Dan Evaluasi

## 3. Membuat Dan Membaca Confusion Matrix Beserta Contoh

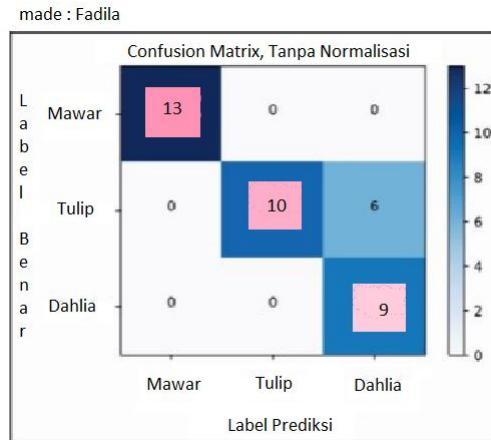


Figure 2.64: Contoh Evaluasi Dan Akurasi Secara Bersamaan

- Pengertian Confusion Matrix

Confusion matrix adalah suatu metode yang biasanya digunakan untuk melakukan perhitungan akurasi pada konsep data mining. Rumus ini melakukan perhitungan dengan 4 keluaran, yaitu: recall, precision, accuracy dan error rate.

- Pembacaan Confusion Matrix

- Apabila hasil prediksi negatif dan data sebenarnya merupakan negatif.
- Apabila hasil prediksi positif sedangkan nilai sebenarnya merupakan negatif.
- Apabila hasil prediksi negatif sedangkan nilai sebenarnya merupakan positif.
- Apabila hasil prediksi positif dan nilai sebenarnya merupakan positif.

- Pembuatan Confusion Matrix

- Menentukan 4 proses klasifikasi yang akan digunakan dalam confusion matrix.
- 4 Istilah tersebut ada True Positive ( TP ), True Negative ( TN ), False Positive ( FP ) dan False Negative ( FN ).
- Kelompokkan klasifikasi tersebut bisa menggunakan klasifikasi biner
- Akan menghasilkan keluaran berupa 2 Kelas ( Positif dan Negatif ) dan penentuan TP, FP ( 1 klasifikasi positif ), FN dan TN ( 1 klasifikasi negatif ).
- Contoh dasarnya nampak seperti langkah diatas

- (f) Istilahnya dapat didefinisikan dengan objek lain namun dengan alur yang sama ( sesuai rumus baik klasifikasi dll ).

- Ilustrasi Gambar

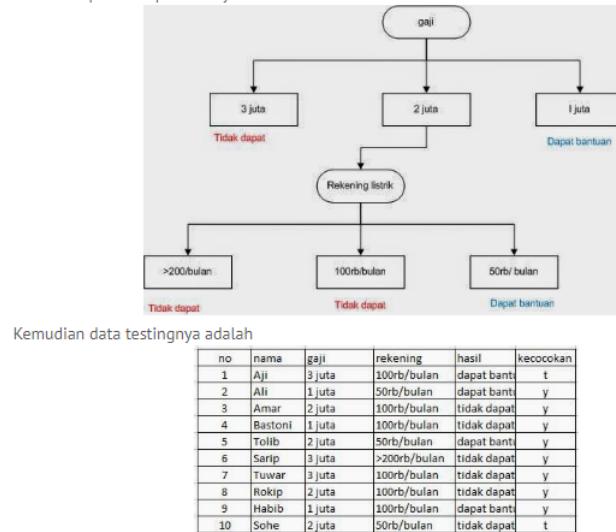


Figure 2.65: confusion matrix

#### 4. Cara Kerja K-Fold Classification Dan Ilustrasi Gambar

- Pertama-tama untuk total instance dibagi menjadi N bagian.
- Fold ke-1 ( atau pertama ) adalah ketika bagian ke-1 menjadi data uji (testing data) dan sisanya menjadi data latih (training data).
- Hitung akurasi ( berdasarkan porsi data tersebut. Persamaannya sebagai berikut :
 
$$\frac{\text{(sigma) data klasifikasi}}{\text{(sigma) total data uji}} \times 100 \text{ persen}$$
- Fold ke-2 ( kedua ) adalah ketika bagian ke-2 menjadi data uji (testing data) dan sisanya menjadi data latih (training data).
- Kemudian dihitunglah akurasi berdasarkan porsi data yang telah ditentukan
- Demikian seterusnya hingga mencapai fold ke-K. Hitung rata-rata akurasi dari K buah akurasi di atas. Rata-rata akurasi ini menjadi akurasi final atau akhir.

- Ilustrasi Gambar

| Dataset:                                   |  |  |  |  |
|--------------------------------------------|--|--|--|--|
| “   K1   K2   K3   K4   K5                 |  |  |  |  |
| -----                                      |  |  |  |  |
| Data Eksperimen:                           |  |  |  |  |
| “   Eksperimen Ke   Data Latih   Data Test |  |  |  |  |
| -----                                      |  |  |  |  |
| 1   K2,K3,K4,K5   K1                       |  |  |  |  |
| -----                                      |  |  |  |  |
| 2   K1,K3,K4,K5   K2                       |  |  |  |  |
| -----                                      |  |  |  |  |
| 3   K1,K2,K4,K5   K3                       |  |  |  |  |
| -----                                      |  |  |  |  |
| 4   K1,K2,K3,K5   K4                       |  |  |  |  |
| -----                                      |  |  |  |  |
| 5   K1,K2,K3,K4   K5                       |  |  |  |  |
| -----                                      |  |  |  |  |

Figure 2.66: k-fold classification 1

- Decision Tree Dan Ilustrasi Gambar

- Pengertian Decision Tree

Decision Tree (Pohon Keputusan) adalah pohon dimana setiap cabangnya menunjukkan pilihan diantara sejumlah alternatif pilihan yang ada, dan setiapdaunnya menunjukkan keputusan yang dipilih. Decision tree biasa digunakan untuk mendapatkan informasi untuk tujuan-pengambilan sebuah keputusan. Decision tree dimulai dengan sebuah root node(titik awal) yang dipakai oleh user untuk mengambil tindakan. Dari node root ini, user memecahnya sesuai dengan algoritma decision tree. Hasil akhirnya adalah sebuah decision tree dengan setiap cabangnya menunjukkan kemungkinan sekenario dari keputusan yang diambil serta hasilnya.

- Ilustrasi Gambar

- Information Gain Dan Entropi

- Pengertian Information Gain

Information gain adalah salah satu attribute selection measure yang digunakan untuk memilih test attribute tiap node pada tree. Atribut dengan information gain tertinggi dipilih sebagai test attribute dari suatu node. Ada 2 kasus berbeda pada saat penghitungan Information Gain, pertama untuk kasus penghitungan attribute tanpa missing value dan kedua, penghitungan attribute dengan missing value.

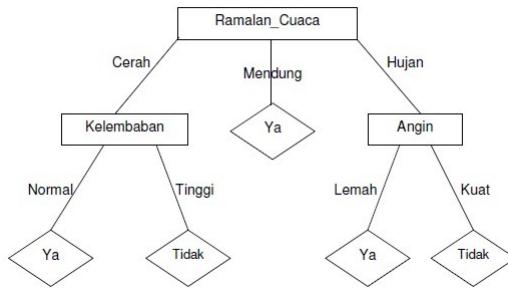


Figure 2.67: decision tree

- Ilustrasi Gambar

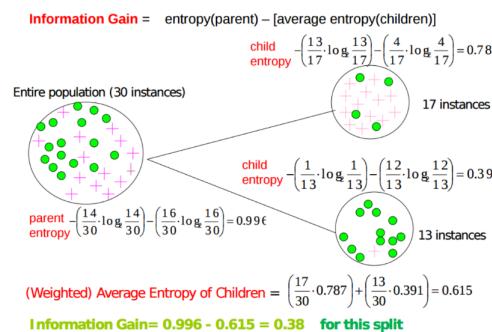


Figure 2.68: information gain 1

- Pengertian Entropi

Adalah suatu parameter untuk mengukur tingkat keberagaman (heterogenitas) dari kumpulan data. Semakin heterogen, nilai entropi semakin besar.

### 2.3.2 Scikit-learn

Penyelesaian Tugas Harian 4 ( No. 1-12 )

- Pembahasan Codingan Dan Hasilnya

- (a) Gambar Pertama :

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

– Hasil Gambar Pertama :

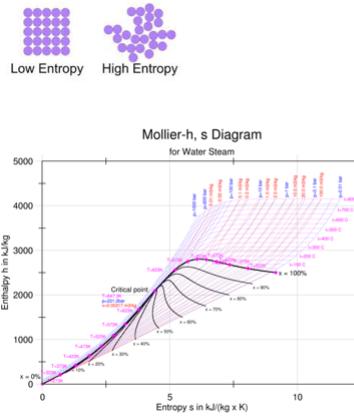


Figure 2.69: entropi

```
In [40]: import pandas as solok
...: solok = solok.read_csv('student-mat.csv', sep=';')
...: len(solok)
Out[40]: 395
```

Figure 2.70: Gambar pertama

(b) Gambar Kedua :

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

– Hasil Gambar Kedua :

```
In [41]: solok['pass'] = solok.apply(lambda row: 1 if (row['G1']+row['G2']+row['G3']) >=
35 else 0, axis=1)
...: solok = solok.drop(['G1', 'G2', 'G3'], axis=1)
...: solok.head()
Out[41]:
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 2.71: Gambar kedua

(c) Gambar Ketiga :

Penjelasan : Variabel solok 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 [42]: solok = pd.get_dummies(solok, columns=['sex', 'school', 'address', 'famsize', 'Pstatus', 'Mjob',
...: 'Fjob', ...: 'reason', 'guardian', 'schoolsup', 'famsup', 'paid', 'activities', 'nursery', 'higher', 'internet', 'romantic'])
...: solok.head()
Out[42]:
 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 2.72: Gambar Ketiga

(d) Gambar Keempat :

Penjelasan : Penejelasan pada gambar keempat adalah variabel solok 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 [43]: solok = solok.sample(frac=1)
...: # split training and testing data
...: solok_train = solok[:500]
...: solok_test = solok[500:]
...:
...: solok_train_att = solok_train.drop(['pass'], axis=1)
...: solok_train_pass = solok_train['pass']
...:
...: solok_test_att = solok_test.drop(['pass'], axis=1)
...: solok_test_pass = solok_test['pass']
...:
...: solok_att = solok.drop(['pass'], axis=1)
...: solok_pass = solok['pass']
...:
...: # number of passing students in whole dataset:
...: import numpy as np
...: print("Passing: %d out of %d (%.2f%%)" % (np.sum(solok_pass), len(solok_pass),
100*float(np.sum(solok_pass)) / len(solok_pass)))
Passing: 166 out of 395 (42.03%)
```

Figure 2.73: Gambar Keempat

(e) 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 [44]: from sklearn import tree
...: solo = tree.DecisionTreeClassifier(criterion="entropy", max_depth=5)
...: solo = solo.fit(solok_train_att, solok_train_pass)
```

Figure 2.74: Gambar Kelima

(f) Gambar Keenam :

Penjelasan : Pada gambar nomor 6, tejadi kesalah error yaitu pada import graphviz.

– Hasil Gambar Keenam :

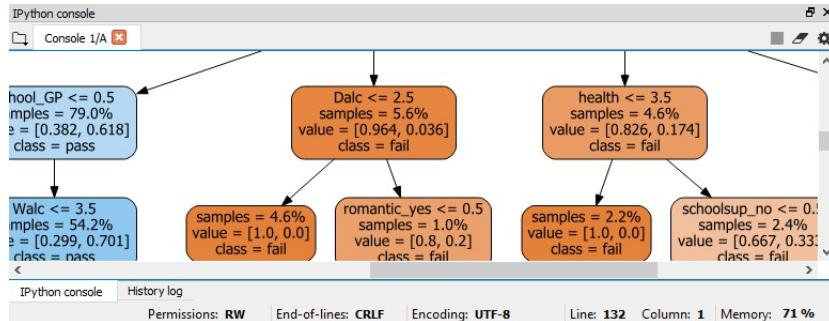


Figure 2.75: Gambar Keenam

(g) 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 [47]: tree.export_graphviz(solo, out_file="student-performance.dot", label="all", impurity=False,
proportion=True,
...: feature_names=list(solok_train_att), class_names=["fail", "pass"],
...: filled=True, rounded=True)
```

Figure 2.76: Gambar Ketujuh

(h) Gambar Kedelapan :

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

– Hasil Gambar Kedelapan :

```
In [34]: solo.score(solok_test_att, solok_test_pass)
Out[34]: 0.6577181208053692
```

Figure 2.77: Gambar Kedelapan

(i) Gambar Kesembilan:

Penjelasan : Pada gambar 9, 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.

```
In [49]: from sklearn.model_selection import cross_val_score
...: scores = cross_val_score(solo, solok_att, solok_pass, cv=5)
...: # mean accuracy, and a 95.0% confidence interval
...: # (mean - 2*std, mean + 2*std)
...: print("Accuracy: %0.2f (+/- %0.2f)" % (scores.mean(), scores.std() * 2))
Accuracy: 0.53 (+/- 0.11)
```

Figure 2.78: Gambar Kesembilan

– Hasil Gambar Kesembilan :

(j) 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 [50]: for max_depth in range(1, 20):
...: solo = tree.DecisionTreeClassifier(criterion="entropy", max_depth=max_depth)
...: scores = cross_val_score(solo, solok_att, solok_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 (+/- 0.07)
Max depth: 3, Accuracy: 0.59 (+/- 0.14)
Max depth: 4, Accuracy: 0.54 (+/- 0.02)
Max depth: 5, Accuracy: 0.53 (+/- 0.11)
Max depth: 6, Accuracy: 0.57 (+/- 0.11)
Max depth: 7, Accuracy: 0.56 (+/- 0.11)
Max depth: 8, Accuracy: 0.55 (+/- 0.15)
Max depth: 9, Accuracy: 0.57 (+/- 0.09)
Max depth: 10, Accuracy: 0.56 (+/- 0.11)
Max depth: 11, Accuracy: 0.57 (+/- 0.15)
Max depth: 12, Accuracy: 0.58 (+/- 0.09)
Max depth: 13, Accuracy: 0.57 (+/- 0.09)
Max depth: 14, Accuracy: 0.55 (+/- 0.11)
Max depth: 15, Accuracy: 0.57 (+/- 0.09)
Max depth: 16, Accuracy: 0.57 (+/- 0.09)
Max depth: 17, Accuracy: 0.54 (+/- 0.14)
Max depth: 18, Accuracy: 0.55 (+/- 0.09)
Max depth: 19, Accuracy: 0.55 (+/- 0.09)
```

Figure 2.79: Gambar Kesepuluh

(k) 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 solo, solok att, solok pass, cv=5. Yang mana hasil tampilan dari kodingannya adalah outputan seperti gambar 11.

– Hasil Gambar Kesebelas :

(l) Gambar Keduabelas :

```

In [51]: depth_acc = np.empty((19,3), float)
...: i = 0
...: for max_depth in range(1, 20):
...: solo = tree.DecisionTreeClassifier(criterion="entropy", max_depth=max_depth)
...: scores = cross_val_score(solo, solok_att, solok_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[51]:
array([[1.00000000e+00, 5.79751704e-01, 6.30768599e-03],
[2.00000000e+00, 5.90073028e-01, 7.19635495e-02],
[3.00000000e+00, 5.87446446e-01, 1.37192167e-01],
[4.00000000e+00, 5.44337877e-01, 2.41201036e-02],
[5.00000000e+00, 5.29500162e-01, 1.05510772e-01],
[6.00000000e+00, 5.85200422e-01, 1.22237161e-01],
[7.00000000e+00, 5.64788218e-01, 9.32673367e-02],
[8.00000000e+00, 5.37129998e-01, 1.21107315e-01],
[9.00000000e+00, 5.69980526e-01, 1.27258587e-01],
[1.00000000e+01, 5.52448888e-01, 1.49180519e-01],
[1.10000000e+01, 5.67672022e-01, 1.59022626e-01],
[1.20000000e+01, 5.82702045e-01, 1.36814986e-01],
[1.30000000e+01, 5.49758195e-01, 1.52087945e-01],
[1.40000000e+01, 5.67417235e-01, 1.37785627e-01],
[1.50000000e+01, 5.64978903e-01, 1.11136389e-01],

```

Figure 2.80: Gambar Kesebelas

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 :

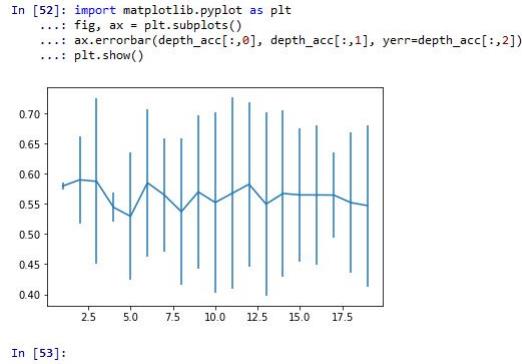


Figure 2.81: Gambar Keduabelas

### 2.3.3 Penanganan Error

(a) Skrinsut Error

(b) Kode Error dan Jenis Errornya

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

Jenis Error: Pada Grafik

(c) Penanganan

Melakukan install ulang pada graphiz

```
In [65]: import graphviz
...: dot_data = tree.export_graphviz(solo, out_file=None, label="all", impurity=False, proportion=True,
...: feature_names=list(solo_k_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 2.82: Gambar Ketigabelas

# Chapter 3

## Prediksi dengan Random Forest

### 3.1 Lusia Violita Aprilian/1164080

#### 3.1.1 Teori

##### 1. Random Forest

- Random Forest diperkenalkan dan diselidiki untuk memprediksi aktivitas biologis kategoris atau kategoris suatu senyawa berdasarkan pada deskripsi kuantitatif dari struktur molekul senyawa. Random Forest adalah ensemble dari pohon klasifikasi atau regresi yang tidak ditandai yang dibuat dengan menggunakan sampel bootstrap dari data pelatihan dan pemilihan fitur acak dalam induksi pohon. Prediksi dibuat dengan menggabungkan (suara terbanyak atau rata-rata) prediksi ensemble.
- Gambar Random Forest

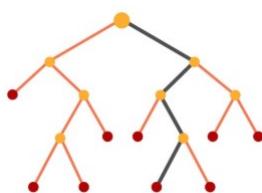


Figure 3.1: Random Forest

##### 2. Membaca Dataset

- Berikut adalah cara membaca dataset :
  - (a) Buka Anaconda Navigator lalu jalankan Spyder, kemudian import libraries yang dibutuhkan.

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

Figure 3.2: Kode Python

- (b) Masukkan kode python untuk membaca file csv, lalu jalankan.
- (c) Maka pada window console akan menampilkan pesan berikut :

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

Figure 3.3: Pesan Window Console

- (d) Dari explorer dapat terlihat dataset yang terimport.

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

Figure 3.4: Dataset

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

| Index | Country | Age | Salary | Purchased |
|-------|---------|-----|--------|-----------|
| 0     | France  | 44  | 72000  | No        |
| 1     | Spain   | 27  | 48000  | Yes       |
| 2     | Germany | 38  | 54000  | No        |
| 3     | Spain   | 38  | 61000  | No        |
| 4     | Germany | 40  | nan    | Yes       |
| 5     | France  | 35  | 58000  | Yes       |
| 6     | Spain   | nan | 52000  | No        |
| 7     | France  | 48  | 79000  | Yes       |
| 8     | Germany | 50  | 83000  | No        |
| 9     | France  | 37  | 67000  | Yes       |

Figure 3.5: Hasil Dataset Cell

- (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: Perintah

- (i) Perintah yang telah dibuat di atas akan membuat sebuah global environment baru dan muncul dataset.
- (j) Klikdataset tersebut maka muncul tabel berisi dataset.

### 3. Cross Validation

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

### 4. Menjelaskan Arti Score

- Maksud arti score 44% pada random forest adalah hasil akurasi.
- Maksud arti score 27% pada decision tree adalah persentasi hasil dari perhitungan dataset.
- Maksud arti 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 didapat outputnya 44% untuk hutan acak, 27% untuk pohon keputusan, dan 29% untuk SVM.

### 5. Cara Membaca Confusion Matriks

- Mari kita lihat contoh klasifikasi biner berikut ini, yang menunjukkan beberapa kali model telah membuat prediksi objek yang benar:

|              |               | Predicted "apple" | Predicted "orange" |
|--------------|---------------|-------------------|--------------------|
| True "apple" | True "apple"  | 20                | 5                  |
|              | True "orange" | 3                 | 22                 |

Figure 3.7: Tabel Confusion Matriks

- Dalam tabel tersebut, baris apple dan orange mengacu pada kasus di mana objek tersebut sebenarnya sebuah apel atau sebuah jeruk. Kolom merujuk pada prediksi yang dibuat oleh model. Kita melihat bahwa dalam contoh

ada 20 apel yang diprediksi dengan benar, sementara ada 5 apel yang salah diidentifikasi sebagai jeruk. Idealnya, confusion matriks harus memiliki semua nilai nol, kecuali untuk diagonal. 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:

$$\text{Accuracy} = (20 + 22)/20 + 5 + 3 + 22) = 84$$

Figure 3.8: Rumus Confusion Matriks

- Sehingga dari perhitungan tersebut kita mendapat akurasi 84
- Berikut adalah gambar klasifikasi biner :

|                   | Class 1<br>Predicted | Class 2<br>Predicted |
|-------------------|----------------------|----------------------|
| Class 1<br>Actual | TP                   | FN                   |
| Class 2<br>Actual | FP                   | TN                   |

Figure 3.9: Confusion Matriks

## 6. Voting Pada Random Forest

- Voting pada random forest merupakan metode yang paling umum digunakan setelah classifier membuat keputusan.
- Gambar voting pada random forest

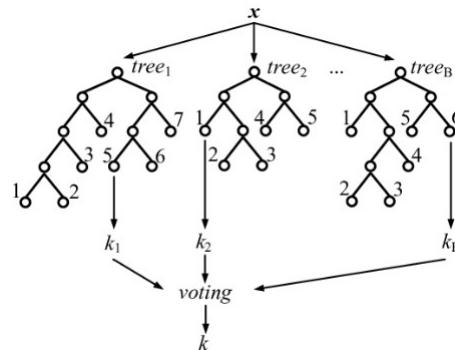


Figure 3.10: Voting Pada Random Forest

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


```

Figure 3.11: Aplikasi Pandas

### 3.1.2 Praktek

#### 1. Aplikasi Sederhana Menggunakan Pandas

Penjelasan kodingan :

- (a) Memanggil library.
- (b) Membuat variable dengan data frame.
- (c) Menampilkan hasil

Sehingga menghasilkan :

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

Figure 3.12: Hasil Pandas

#### 2. Aplikasi Sederhana Menggunakan Numpy

```
import numpy as np
x = np.random.normal(size=5)
print(x)
```

Figure 3.13: Aplikasi Numpy

Penjelasan kodingan :

- (a) Memanggil library
- (b) Membuat variable dengan value random dan size 5
- (c) Menampilkan hasil value

Sehingga menghasilkan :

```
In [45]: import numpy as np
...: x = np.random.normal(size=5)
...: print(x)
[1.94 0.19 0.57 -0.3 0.33]
```

Figure 3.14: Hasil Numpy

```

import matplotlib.pyplot as plt
Data to plot
languages = ['Java', 'Python', 'PHP', 'JavaScript', 'C#', 'C++'
popularity = [22.2, 17.6, 8.8, 8, 7.7, 6.7]
colors = ["#1f77b4", "#ff7f0e", "#2ca02c", "#d62728", "#9467bd", "#8c564b"]
explode 1st slice
explode = (0.1, 0, 0, 0, 0)
Plot
plt.pie(popularity, explode=explode, labels=languages, colors=colors,
 autopct='%1.1f%%', shadow=True, startangle=140)
plt.axis('equal')
plt.show()

```

Figure 3.15: Aplikasi Matplotlib

### 3. Aplikasi Sederhana Menggunakan Matplotlib

Penjelasan kodingan :

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

Sehingga menghasilkan :

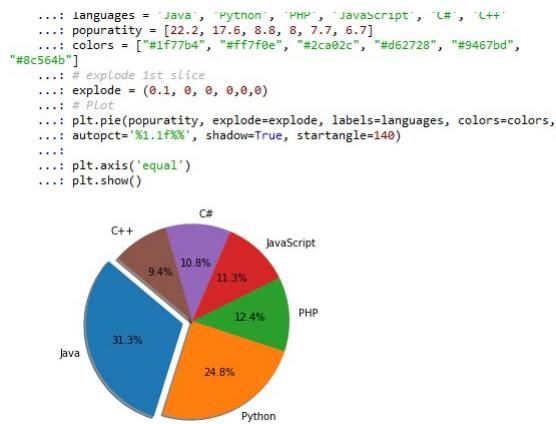


Figure 3.16: Hasil Matplotlib

### 4. Program Klasifikasi Random Fores

- Yang pertama dataset akan dibaca.

```
In [9]: import pandas as pd
...
...: # some lines have too many fields (?), so skip bad lines
...: imgatt = pd.read_csv("CUB_200_2011/attributes/image_attribute_labels.txt",
...: sep='\t', header=None, error_bad_lines=False,
warn_bad_lines=False,
...: usecols=[0,1,2], names=['imgid', 'attid', 'present'])
...
...: # description from dataset README:
...: #
...: # The set of attribute labels as perceived by MTurkers for each image
...: # is contained in the file attributes/image_attribute_labels.txt, with
...: # each line corresponding to one image/attribute/worker triplet:
...: #
...: # <image_id> <attribute_id> <is_present> <certainty_id> <time>
...: #
...: # where <image_id>, <attribute_id>, <certainty_id> correspond to the IDs
...: # in images.txt, attributes/attributes.txt, and attributes/certainties.txt
...: # respectively. <is_present> is 0 or 1 (1 denotes that the attribute is
...: # present). <time> denotes the time spent by the MTurker in seconds.
```

Figure 3.17: Membaca Data File

```
In [10]: imgatt.head()
Out[10]:
 imgid attid present
0 1 1 0
1 1 2 0
2 1 3 0
3 1 4 0
4 1 5 1
```

Figure 3.18: Melihat Data Sebagian

- Selanjutnya sebagian data awal akan dilihat dengan menggunakan listing.
- Selanjutnya jumlah data dilihat dengan menggunakan listing.

```
In [11]: imgatt.shape
Out[11]: (3677856, 3)
```

Figure 3.19: Melihat Jumlah Data

- Lalu atribut diubah menjadi kolom dengan menggunakan perintah pivot.

```
In [12]: imgatt2 = imgatt.pivot(index='imgid', columns='attid', values='present')
```

Figure 3.20: Mengubah menjadi kolom

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

```
In [14]: imgatt2.head()
Out[14]:
 attid 1 2 3 4 5 6 7 ... 306 307 308 309 310 311 312
imgid
1 0 0 0 0 1 0 0 ... 0 0 1 0 0 0 0
2 0 0 0 0 0 0 0 ... 0 0 0 0 0 0 0
3 0 0 0 0 1 0 0 ... 0 0 1 0 0 1 0
4 0 0 0 0 1 0 0 ... 1 0 0 1 0 0 0
5 0 0 0 0 1 0 0 ... 0 0 0 0 0 0 0
```

[5 rows x 312 columns]

Figure 3.21: Lihat sebagian data awal

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

```
In [15]: imgatt2.shape
Out[15]: (11788, 312)
```

Figure 3.22: Melihat jumlah data

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

```
In [17]: imglabels = pd.read_csv("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.
```

Figure 3.23: Mengelompokkan burung

- Lalu melakukan pivot dimana imgid menjadi index.

```
In [18]: imglabels.head()
Out[18]:
 label
imgid
1 1
2 1
3 1
4 1
5 1
```

Figure 3.24: Melalukan pivot

- Selanjutnya imgid, sebagian data awalnya akan dilihat dengan menggunakan listing untuk mengecek data.

```
In [19]: imglabels.shape
Out[19]: (11788, 1)
```

Figure 3.25: Melihat data awal imgid

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

```
In [20]: df = imgatt2.join(imglabels)
...: df = df.sample(frac=1)
```

Figure 3.26: 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.

```
In [21]: df_att = df.iloc[:, :312]
...: df_label = df.iloc[:, 312:]
```

Figure 3.27: Data ciri label dari join

```
In [22]: df_att.head()
Out[22]:
 1 2 3 4 5 6 7 ... 306 307 308 309 310 311 312
imgid
3376 0 0 0 0 0 0 0 ... 0 0 0 0 0 1 0
960 0 0 0 0 0 0 0 ... 0 0 0 0 1 0 0
3297 0 0 0 0 0 0 0 ... 0 0 0 0 0 0 1
9899 0 0 0 0 0 0 1 ... 1 0 0 0 0 1 0
5179 0 0 0 0 1 0 0 ... 0 0 0 0 1 0 0
[5 rows x 312 columns]
```

Figure 3.28: Mengubah menjadi kolom

```
In [23]: df_label.head()
Out[23]:
 label
imgid
3376 59
960 18
3297 57
9899 169
5179 89
```

Figure 3.29: Melihat isi data frame

- Lalu cek kembali isinya dengan perintah listing.
- Kemudian data dibagi menjadi dua bagian, dimana 8000 row pertama merupakan data training dan sisanya adalah data testing.

```
In [23]: df_label.head()
Out[23]:
 label
imgid
3376 59
960 18
3297 57
9899 169
5179 89

In [24]: df_train_att = df_att[:8000]
...: df_train_label = df_label[:8000]
...: df_test_att = df_att[8000:]
...: df_test_label = df_label[8000:]
...:
...: df_train_label = df_train_label['label']
...: df_test_label = df_test_label['label']
```

Figure 3.30: Membagi data

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

```
In [25]: from sklearn.ensemble import RandomForestClassifier
...: clf = RandomForestClassifier(max_features=50, random_state=0, n_estimators=100)
```

Figure 3.31: Kelas Random Forest

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

```
In [26]: clf.fit(df_train_att, df_train_label)
```

Figure 3.32: Membangun Random forest

```
In [27]: print(clf.predict(df_train_att.head()))
[59 18 57 169 89]
```

Figure 3.33: Melihat hasil

- Lalu akan terlihat hasil score dari klasifikasi.

```
In [28]: clf.score(df_test_att, df_test_label)
Out[28]: 0.4498416050686378
```

Figure 3.34: Lihat hasil score

## 5. Program Klasifikasi Confusion Matrix

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

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

Figure 3.35: Memetakan ke confusion matrix

- Lalu melihat hasilnya.

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

Figure 3.36: Melihat hasil

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

Figure 3.37: Melakukan Plot

```
In [33]: birds = pd.read_csv("CUB_200_2011/classes.txt",
...: sep="\s+", header=None, usecols=[1],
...: names=['birdname'])
...: birds = birds['birdname']
...: birds
Out[33]:
0 001.Black_footed_Albatross
1 002.Laysan_Albatross
2 003.Sooty_Albatross
3 004.Groove_billed_Ani
4 005.Crested_Auklet
5 006.Least_Auklet
6 007.Parakeet_Auklet
7 008.Rhinoceros_Auklet
8 009.Brewer_Blackbird
9 010.Red_winged_Blackbird
10 011.Rusty_Blackbird
11 012.Yellow_headed_Blackbird
12 013.Bobolink
13 014.Indigo_Bunting
14 015.Lazuli_Bunting
15 016.Painted_Bunting
16 017.Cardinal
17 018.Spotted_Catbird
18 019.Gray_Catbird
```

Figure 3.38: Plotting nama data

```
In [34]: import numpy as np
...: np.set_printoptions(precision=2)
...: plt.figure(figsize=(60,60), dpi=300)
...: plot_confusion_matrix(cm, classes=birds, normalize=True)
...: plt.show()
Normalized confusion matrix
[[0.35 0. 0.2 ... 0. 0.05 0.]
 [0. 0.63 0. ... 0. 0. 0.]
 [0.04 0. 0.46 ... 0. 0. 0.]
 ...
 [0. 0. 0.05 ... 0.21 0. 0.]
 [0. 0. 0. ... 0. 0.38 0.]
 [0. 0. 0. ... 0. 0. 0.88]]
```

Figure 3.39: Melakukan perintah plot

```
In [35]: from sklearn import tree
...: clftree = tree.DecisionTreeClassifier()
...: clftree.fit(df_train_att, df_train_label)
...: clftree.score(df_test_att, df_test_label)
In [36]: Out[35]: 0.270855332629355850ut[36]: 0.27560718057022177
```

Figure 3.40: Klasifikasi menggunakan decision tree

```

In [37]: from sklearn import svm
...: clfsvm = svm.SVC()
...: clfsvm.fit(df_train_att, df_train_label)
...: clfsvm.score(df_test_att, df_test_label)
C:\Users\lsvapr\Anaconda3\lib\site-packages\sklearn\svm\base.py:196:
FutureWarning: The default value of gamma will change from 'auto' to 'scale' in
version 0.22 to account better for unscaled features. Set gamma explicitly to
'auto' or 'scale' to avoid this warning.
"avoid this warning.", FutureWarning)

Out[37]: 0.28299894403379094C:\Users\lsvapr\Anaconda3\lib\site-packages\sklearn
\svm\base.py:196: FutureWarning: The default value of gamma will change from
'auto' to 'scale' in version 0.22 to account better for unscaled features. Set
gamma explicitly to 'auto' or 'scale' to avoid this warning.
"avoid this warning.", FutureWarning)

```

Figure 3.41: Klasifikasi menggunakan SVM

(b) Program Klasifikasi SVM

Mengklasifikasikan dataset yang sama menggunakan SVM.

7. Program Cross Validation

- Melakukan pengecekan cross validation untuk random forest.

```

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

```

Figure 3.42: Pengecekan cross validation random forest

- Melakukan pengecekan cross validation untuk decision tree.

```

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

```

Figure 3.43: Pengecekan cross validation decision tree

- Melakukan pengecekan cross validation untuk SVM.

```

...: print("Accuracy: %0.2f (+/- %0.2f)" % (scoressvm.mean(), scoressvm.std()
* 2))
C:\Users\lsvapr\Anaconda3\lib\site-packages\sklearn\svm\base.py:196:
FutureWarning: The default value of gamma will change from 'auto' to 'scale' in
version 0.22 to account better for unscaled features. Set gamma explicitly to
'auto' or 'scale' to avoid this warning.
"avoid this warning.", FutureWarning)
C:\Users\lsvapr\Anaconda3\lib\site-packages\sklearn\svm\base.py:196:
FutureWarning: The default value of gamma will change from 'auto' to 'scale' in
version 0.22 to account better for unscaled features. Set gamma explicitly to
'auto' or 'scale' to avoid this warning.
"avoid this warning.", FutureWarning)
C:\Users\lsvapr\Anaconda3\lib\site-packages\sklearn\svm\base.py:196:
FutureWarning: The default value of gamma will change from 'auto' to 'scale' in
version 0.22 to account better for unscaled features. Set gamma explicitly to
'auto' or 'scale' to avoid this warning.
"avoid this warning.", FutureWarning)
C:\Users\lsvapr\Anaconda3\lib\site-packages\sklearn\svm\base.py:196:
FutureWarning: The default value of gamma will change from 'auto' to 'scale' in
version 0.22 to account better for unscaled features. Set gamma explicitly to
'auto' or 'scale' to avoid this warning.
"avoid this warning.", FutureWarning)
C:\Users\lsvapr\Anaconda3\lib\site-packages\sklearn\svm\base.py:196:
FutureWarning: The default value of gamma will change from 'auto' to 'scale' in
version 0.22 to account better for unscaled features. Set gamma explicitly to
'auto' or 'scale' to avoid this warning.
"avoid this warning.", FutureWarning)
Accuracy: 0.26 (+/- 0.02)

```

Figure 3.44: Pengecekan cross validation SVM

## 8. Program Pengamatan Komponen Informasi

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

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

Figure 3.45: Pengamatan Komponen

- Melakukan plot informasi agar bisa dibaca.

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

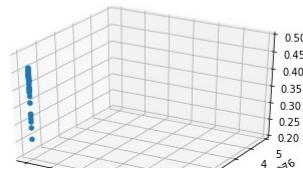


Figure 3.46: Plot informasi

### 3.1.3 Penanganan Error

#### 1. Skrinsut Error

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

Figure 3.47: 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 :

```
import pandas as pd
some lines have too many fields (?), so skip bad lines
imgatt = pd.read_csv("CUB_200_2011/attributes/image_attribute_labels.txt",
 sep='\s+', header=None, error_bad_lines=False,
 usecols=[0,1,2], names=['imgid', 'attid', 'present'])
```

Figure 3.48: Penyelesaian

Sehingga didapat hasil seperti berikut :

```
In [9]: import pandas as pd
...
...: # some lines have too many fields (?), so skip bad lines
...: imgatt = pd.read_csv("CUB_200_2011/attributes/image_attribute_labels.txt",
...: sep='\s+', header=None, error_bad_lines=False,
...: warn_bad_lines=False,
...: usecols=[0,1,2], names=['imgid', 'attid', 'present'])
...
...: # description from dataset README:
...: #
...: # The set of attribute labels as perceived by MTurkers for each image
...: # is contained in the file attributes/image_attribute_labels.txt, with
...: # each Line corresponding to one image/attribute/worker triplet:
...: #
...: # <image_id> <attribute_id> <is_present> <certainty_id> <time>
...: #
...: # where <image_id>, <attribute_id>, <certainty_id> correspond to the IDs
...: # in images.txt, attributes/attributes.txt, and attributes/certainties.txt
...: # respectively. <is_present> is 0 or 1 (1 denotes that the attribute is
...: # present). <time> denotes the time spent by the MTurker in seconds.
```

Figure 3.49: Hasil

## 3.2 Rahmi Roza/1164085

### 3.2.1 Teori

Penyelesaian Tugas Harian 5 ( No. 1-6 )

#### 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 :

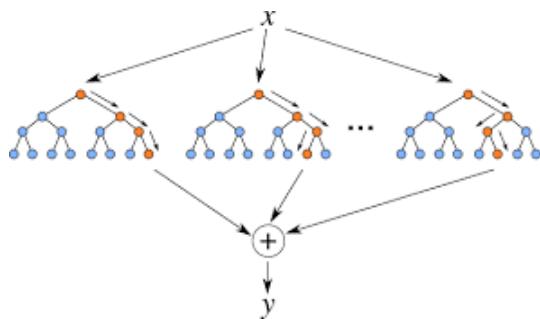


Figure 3.50: Random Forest

2. Cara Membaca Dataset, Dan artikan makna setiap file dan isinya.

- Cara Membaca Dataset :

- (a) Buka Anaconda Navigator lalu jalankan Spyder, kemudian import libraries yang dibutuhkan.
- (b) Masukkan kode python untuk membaca file csv, lalu jalankan.

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

Figure 3.51: (b)

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

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

Figure 3.52: (c)

- (d) Dari explorer dapat terlihat dataset yang terimport.

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

Figure 3.53: (d)

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

Figure 3.54: (e)

- (e) Lalu klik dataset cell, maka akan muncul seperti berikut :
- (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 3.55: (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 atau bisa disebut estimasi rotasi ,adalah sebuah teknik validasi model untuk menilai bagaimana hasil statistik analisis akan menggeneralisasi kumpulan data independen. Teknik ini utamanya digunakan untuk melakukan prediksi model dan memperkirakan seberapa akurat sebuah model prediktif ketika dijalankan dalam praktiknya.

Dalam sebuah masalah prediksi, sebuah model biasanya diberikan kumpulan data (dataset) yang diketahui untuk digunakan dalam menjalankan pelatihan (dataset pelatihan), serta kumpulan data yang tidak diketahui (atau data yang pertama kali dilihat) terhadap model yang diuji (penguji dataset).

Tujuan dari validasi silang adalah untuk mendefinisikan dataset untuk "menguji" model dalam tahap pelatihan (yaitu, validasi data), dalam rangka untuk membatasi masalah seperti terjadinya overfitting, memberikan wawasan tentang bagaimana model akan menggeneralisasi independen dataset (yaitu, tidak diketahui dataset, misalnya dari masalah nyata), dll.

#### 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, decision tree menjadi lebih buruk. Jika kita menggunakan Support Vector Machine (SVM), yang merupakan neural pendekatan jaringan, outputnya 29%. Jadi 29% pada SVM merupakan hasil outputnya.

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 :

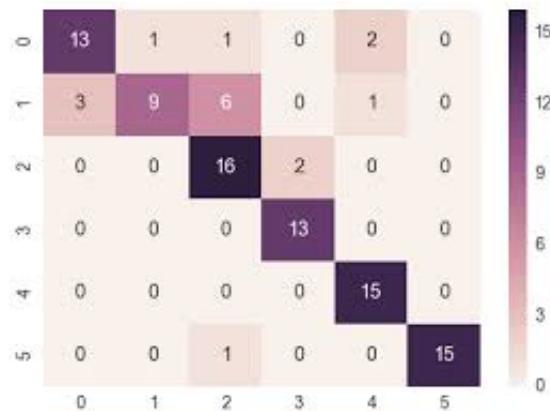


Figure 3.56: Confussion Matrik

## 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. 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 :

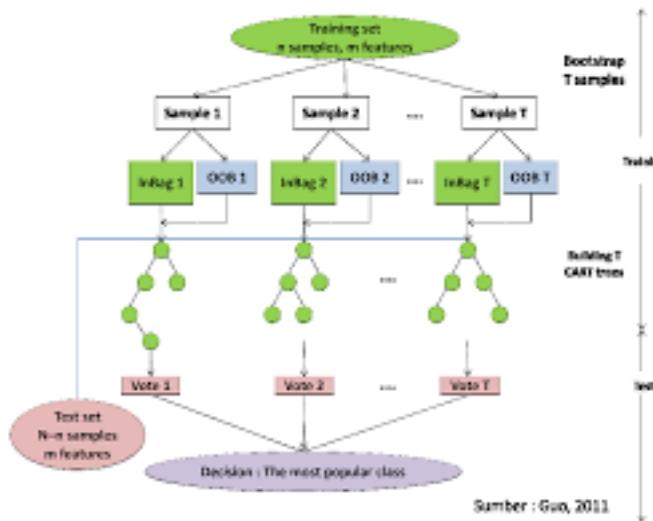


Figure 3.57: Voting Random forest

### 3.2.2 Praktek

Penyelesaian Tugas Harian 6 ( No. 1-8 )

#### 1. Aplikasi Sederhana Pandas dan Penjelasan Code ( Perbaris )

- Pandas:
  - Penjelasan Code Baris 1 : Mengimport library pandas dari python dengan inisiasi pd.
  - Penjelasan Code Baris 2 : Variabel s1 mendefinisikan data menggunakan pandas series.
  - Penjelasan Code Baris 3 : Mencetak output
  - Penjelasan Code Baris 4 : Menampilkan data s1
  - Penjelasan Code Baris 5 : Mencetak output
  - Penjelasan Code Baris 6 : Mengubah data karakter menjadi numerik
  - Penjelasan Code Baris 7 : Mempulkan data s2
- Hasil:
- Aplikasi Sederhana Numpy dan Penjelasan Code ( Perbaris )
  - Code Numpy:
    - \* Penjelasan Code Baris 1 : Import library numpy

```

Original Data Series:
0 100
1 200
2 python
3 300.12
4 400
dtype: object
Change the said data type to numeric:
0 100.00
1 200.00
2 NaN
3 300.12
4 400.00
dtype: float64

```

Figure 3.58: Pandas

- \* Penjelasan Code Baris 2 : Menampilkan versi dari numpy.
- \* Penjelasan Code Baris 3 : Menampilkan konfigurasi dari numpy.
- Hasil:

```

1.15.4
mkl_info:
 libraries = ['mkl_rt']
 library_dirs = ['C:/ProgramData/Anaconda3/Library/lib']
 define_macros = [('SCIPY_MKL_H', None), ('HAVE_CBLAS', None)]
 include_dirs = ['C:\Program Files (x86)\IntelSWTools\compilers_and_libraries_2019.0.117\windows\mkl', 'C:\Program Files (x86)\IntelSWTools\compilers_and_libraries_2019.0.117\windows\mkl\include', 'C:\Program Files (x86)\IntelSWTools\compilers_and_libraries_2019.0.117\windows\mkl\lib', 'C:/ProgramData/Anaconda3/Library/include']
blas_mkl_info:
 libraries = ['mkl_rt']
 library_dirs = ['C:/ProgramData/Anaconda3/Library/lib']
 define_macros = [('SCIPY_MKL_H', None), ('HAVE_CBLAS', None)]
 include_dirs = ['C:\Program Files (x86)\IntelSWTools\compilers_and_libraries_2019.0.117\windows\mkl', 'C:\Program Files (x86)\IntelSWTools\compilers_and_libraries_2019.0.117\windows\mkl\include', 'C:\Program Files (x86)\IntelSWTools\compilers_and_libraries_2019.0.117\windows\mkl\lib', 'C:/ProgramData/Anaconda3/Library/include']
blas_opt_info:
 libraries = ['mkl_rt']
 library_dirs = ['C:/ProgramData/Anaconda3/Library/lib']
 define_macros = [('SCIPY_MKL_H', None), ('HAVE_CBLAS', None)]
 include_dirs = ['C:\Program Files (x86)\IntelSWTools\compilers_and_libraries_2019.0.117\windows\mkl', 'C:\Program Files (x86)\IntelSWTools\compilers_and_libraries_2019.0.117\windows\mkl\include', 'C:\Program Files (x86)\IntelSWTools\compilers_and_libraries_2019.0.117\windows\mkl\lib', 'C:/ProgramData/Anaconda3/Library/include']

```

Figure 3.59: Numpy

- Aplikasi Sederhana Matplotlib dan Penjelasan Code ( Perbaris )

- \* Code Matplotlib:
  - Penjelasan Code Baris 1 : Mengimport library matplotlib dari python dengan inisiasi plt.
  - Penjelasan Code Baris 2 : Variabel X
  - Penjelasan Code Baris 3 : Variabel Y

- Penjelasan Code Baris 4 : Menampilkan nilai dari X
- Penjelasan Code Baris 5 : Menampilkan value dari X
- Penjelasan Code Baris 6 : Menampilkan value dari variabel Y
- Penjelasan Code Baris 7 : Menampilkan nilai dari variabel Y
- Penjelasan Code Baris 8 : Mengatur label sumbu x dari sumbu saat ini.
- Penjelasan Code Baris 9 : Mengatur label sumbu y dari sumbu saat ini.
- Penjelasan Code Baris 10 : Menetapkan judul atau title.
- Penjelasan Code Baris 11 : Menampilkan figure atau gambar.

\* Hasil:

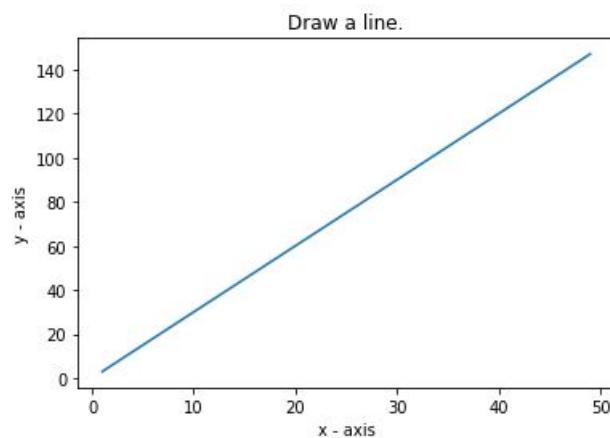


Figure 3.60: Matplotlib

– Program Aplikasi Random Forest dan Penjelasan Keluarannya :

\* Code Random Forest 1 :

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

Figure 3.61: Gambar1

```
In [31]: imgatt.head()
Out[31]:
 imgid attid present
0 1 1 0
1 1 2 0
2 1 3 0
3 1 4 0
4 1 5 1
```

Figure 3.62: Gambar2

- 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 :
  - 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 :

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

Figure 3.63: Gambar3

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

Figure 3.64: Gambar 4

- Penjelasan : Pada gambar di atas menunjukkan load dari jawabannya yang berisi ” apakah burung tersebut ( subjek pada dataset ) termasuk dalam spesies yang mana ?. Kolom yang digunakan adalah imgid dan label, kemudian melakukan pivot yang mana imgid menjadi index yang artinya unik sehubungan dengan dataset yang telah dieksekusi.
- \* Code 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 ).
- \* Code Random Forest 12 :

```

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

[5 rows x 312 columns]

```

Figure 3.65: Gambar 5

```

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

```

Figure 3.66: Gambar 6

- 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.
- \* Code Random Forest 17 :
  - Penjelasan : Pada gambar di atas menunjukkan hasil dari cetakan variabel dftrainatt.head.
- \* Code Random Forest 18 :

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

Figure 3.67: Gambar 7

```
[In [50]: imglabels.head()]
Out[50]:
 imgid label
0 1 1
1 2 1
2 3 1
3 4 1
4 5 1
```

Figure 3.68: Gambar 8

- 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 4 :
    - Penjelasan : Pada gambar di atas merupakan kodingan untuk menyesuaikan sumbu dengan nama datanya makanya datset nya di lakukan dengan perintah di atas.
  - \* Code Confusion Matrix 5 :

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

Figure 3.69: Gambar 9

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

Figure 3.70: Gambar 10

- Penjelasan : Pada gambar di atas merupakan perintah plot dari gambar sebelumnya.
- Program Klasifikasi SVM dan Decision Tree Beserta Penjelasan Keluarannya :
  - \* Code SVM :
    - Penjelasan : Pada gambar di atas cara untuk mencoba klasifikasi dengan SVM dengan dataset yang sama.
  - \* 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 [43]: df_att = df.iloc[:, :312]
...: df_label = df.iloc[:, 312:]
```

Figure 3.71: Gambar 11

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

Figure 3.72: Gambar 12

\* 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.

## 3.3 Fadila/1164072

### 3.3.1 Teori

Penyelesaian Tugas Harian 5 ( No. 1-6 )

#### 1. Random Forest Dan Ilustrasi Gambarnya

- Pengertian Random Forest:

Random forest ( hutan acak ) atau biasa juga disebut dengan istilah random decision forest (hutan keputusan acak ) merupakan metode pembelajaran ensembel untuk klasifikasi, regresi yang beroperasi dengan membangun banyak pohon keputusan pada waktu pelatihan dan menghasilkan kelas yang merupakan mode kelas (klasifikasi) atau prediksi rata-rata (regresi) dari masing-masing pohon.

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

Figure 3.73: Gambar 13

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

Figure 3.74: Gambar 14

- Ilustrasi Gambar Random Forest :
2. Cara Membaca Dataset, Dan artikan makna setiap file dan isinya.
- Cara Membaca Dataset :
- Membaca Dataset
- (a) Langkah-langkah cara membaca dataset :
- Pertama-tama kita harus membuka Aplikasi yang digunakan untuk membuka datasetnya
  - Yang saya gunakan ialah spyder dari Anaconda Navigator
  - Selanjutnya import libraries seperti numpy, pandas, matplotlib dll ( sesuai kebutuhan ).
  - Silahkan untuk memasukkan kode python yang digunakan untuk membaca file csv yang berada pada dataset
- ```
dataset = pd.read_csv('Data.csv')
```
- Kemudian jalankan script python tersebut
 - Perhatikan pada bagian " window consolenya ", tampilan akan seperti berikut :
 - Dari window 'explorer' tersebut, kita dapat melihat dataset yang terimport.
- Akan ada kolom nama, tipe, size dan values
- Lalu klik dataset cell, maka akan muncul seperti berikut :

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

Figure 3.75: Gambar 15

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

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

Figure 3.76: Gambar 16

- Selanjutnya seperti yang dapat dilihat pada gambar tersebut untuk dataset ini memiliki Kolom Country, Age, dan Salary sebagai independent variablenya dan kolom Purchased sebagai dependent variablenya.
- Kemudian pada code buatlah 2 matrix of features yang berisi values / nilai-nilai dari independent variable dan dependent variable yang sesuai.
- Setelah semuanya selesai, maka silahkan tuliskan perintah berikut :

```
dataset = read.csv('Data.csv')
```
- Perintah/ script yang telah dibuat di atas akan membuat sebuah global environment
- Global environmentnya berupa environment baru dan muncullah dataset.
- Silahkan untuk meng-Klik dataset tersebut, kemudian akan muncul tabel berisi dataset.

3. Cross Validation Dan Ilustrasi Gambar :

- Pengertian Cross Validation :

Cross Validation (validasi silang) biasa disebut dengan estimasi rotasi, atau pengujian out-of-sample dimana merupakan salah satu dari berbagai teknik validasi model yang serupa untuk menilai bagaimana hasil analisis

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

Figure 3.77: Gambar 17

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

Figure 3.78: Gambar 18

statistik akan digeneralisasikan ke set data independen. Hal ini (terutama) digunakan dalam pengaturan di mana tujuannya adalah prediksi, dan orang ingin memperkirakan seberapa akurat model prediksi akan dilakukan dalam praktek.

Untuk satu putaran validasi silang melibatkan mempartisi sampel data ke dalam himpunan bagian pelengkap, melakukan analisis pada satu subset (disebut set pelatihan), dan juga memvalidasi analisis pada subset lainnya (disebut set validasi atau set pengujian).

Penjelasan : Pada contoh gambar, didefinisikan bahwa terjadi penilaian hasil analisis statistik yang telah berubah ke set data independen dan merupakan sebuah prediksi. Apabila prediksinya akurat maka tampilannya akan cross seperti gambar tersebut.

4. Penjelasan / Maksud Dari Score pada :

- Random forest (44 %)

Maksudnya adalah akurasi. Dimana Hasil dari eksekusi variabel, parameter dll dari perhitungan code dan decision tree terkait akan diproses lagi untuk menghasilkan nilai akurasi. Dan untuk nilai akurasinya berupa 44 %

- Decision Tree (27 %)

Untuk score dari Decision Tree yang berupa 27 % merupakan presentasi hasil dari perhitungan dataset yang telah dibaca dan diproses sebelumnya.

- SVM (29 %)

Untuk nilai 29 % dari SCV merupakan hasil pendekatan jaringan saraf. Jaringan saraf sendiri merupakan komponen jaringan utama dari sistem saraf. Sistem tersebut mengatur dan mengontrol fungsi tubuh dan aktivitas dan terdiri dari dua bagian: (SSP) yang terdiri dari otak dan sumsum

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

Figure 3.79: Gambar 19

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

Figure 3.80: Gambar 20

tulang belakang, dan percabangan saraf perifer dari sistem saraf tepi (SST) yang terdapat dalam pengolahan dataset terkait.

5. Confusion Matrix Dan Ilustrasinya :

- Pengertian Confusion Matrix :

Pertama-tama confusion matrix ialah suatu metode yang digunakan untuk melakukan perhitungan akurasi pada konsep data mining.

- Cara Membaca Confusion Matrix :

Untuk pembacaan dan pemahaman pada confusion matrix dapat memperhatikan penjelasan berikut

- Perhatikan : Apabila hasil prediksi negatif dan data sebenarnya merupakan negatif.
- Perhatikan : Apabila hasil prediksi positif sedangkan nilai sebenarnya merupakan negatif.
- Perhatikan : Apabila hasil prediksi negatif sedangkan nilai sebenarnya merupakan positif.
- Perhatikan : Apabila hasil prediksi positif dan nilai sebenarnya merupakan positif.

- Pembacaan Lanjutan (lebih jelas) Untuk Confusion Matrix :

- Pastikan definisi dari 4 proses klasifikasi yang akan digunakan ada dalam confusion matrix.
- 4 Istilah tersebut ada TP, TN, FP dan FN
- Cara bacanya yaitu True Positive (TP), True Negative (TN), False Positive (FP) dan False Negative (FN).

```

In [54]: import matplotlib.pyplot as plt
...: import itertools
...: def plot_confusion_matrix(cm, classes,
...:                         normalize=False,
...:                         title='Confusion matrix',
...:                         cmap=plt.cm.Blues):
...:
...:     """
...:     This function prints and plots the confusion matrix.
...:     Normalization can be applied by setting `normalize=True`.
...:
...:     if normalize:
...:         cm = cm.astype('float') / cm.sum(axis=1)[:, np.newaxis]
...:         print("Normalized confusion matrix")
...:     else:
...:         print('Confusion matrix, without normalization')
...:
...:     print(cm)
...:
...:     plt.imshow(cm, interpolation='nearest', cmap=cmap)
...:     plt.title(title)
...:     #plt.colorbar()
...:     tick_marks = np.arange(len(classes))
...:     plt.xticks(tick_marks, classes, rotation=90)
...:     plt.yticks(tick_marks, classes)
...:
...:     fmt = '.2f' if normalize else 'd'
...:     thresh = cm.max() / 2.
...:     #for i, j in itertools.product(range(cm.shape[0]), range(cm.shape[1])):

```

Figure 3.81: Gambar 21

- (d) Ada pendefinisian biner pada klasifikasi confusion matrix
- (e) Klasifikasi tersebut menghasilkan keluaran berupa 2 Kelas (Positif dan Negatif) dan penentuan TP, FP (1 klasifikasi positif) , FN dan TN (1 klasifikasi negatif).

- Ilustrasi Gambar

Penjelasan : Pada gambar dapat dilihat bahwa cara membaca keluaran dari klasifikasi contoh tersebut dibaca 1 dan 0 (yaitu iya atau tidak). Untuk perhitungan lainnya pada klasifikasi untuk confusion marix memang bersifat mutlak atau hanya berada pada 2 pilihan.

6. Voting Random Forest Dan Ilustrasi Gambarnya.

- Pengertian Voting pada Random Forest :

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

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

```

In [56]: birds = pd.read_csv("CUB_200_2011/classes.txt",
...:                         sep='\s+', header=None, usecols=[1], names=['birdname'])
...: birds = birds['birdname']
...: birds
Out[56]:
0      001.Black_footed_Albatross
1      002.Laysan_Albatross
2      003.Sooty_Albatross
3      004.Groove_billed_Ani
4      005.Crested_Auklet
5      006.Least_Auklet
6      007.Parakeet_Auklet
7      008.Rhinoceros_Auklet
8      009.Brewer_Blackbird
9      010.Red_winged_Blackbird
10     011.Rusty_Blackbird
11     012.Yellow_headed_Blackbird
12     013.Bobolink
13     014.Indigo_Bunting
14     015.Lazuli_Bunting
15     016.Painted_Bunting
16     017.Cardinal
17     018.Spotted_Catbird
18     019.Gray_Catbird
19     020.Yellow_breasted_Chat
20     021.Eastern_Towhee
21     022.Chuck_will_Widow
22     023.Brandt_Cormorant

```

Figure 3.82: Gambar 22

```

In [37]: import numpy as np
...: np.set_printoptions(precision=2)
...: plt.figure(figsize=(60,60), dpi=300)
...: plot_confusion_matrix(cm, classes=birds, normalize=True)
...: plt.show()
Normalized confusion matrix
[[0.27 0. 0.18 ... 0. 0. 0. ]
 [0. 0.73 0. ... 0. 0. 0. ]
 [0.08 0. 0.42 ... 0. 0. 0. ]
 ...
 [0. 0. 0. ... 0.2 0. 0. ]
 [0. 0. 0. ... 0. 0.3 0. ]
 [0. 0. 0. ... 0. 0. 0.88]]

```

Figure 3.83: Gambar 23

- Ilustrasi Gambar Voting Random Forest :

Penjelasan : Pada gambar terlihat bahwa terdapat mayoritas voting dimana datanya berasal dari pengolahan decision tree 1, 2 dan 3 . Ketiga decision tree tersebut telah dikelompokkan kedalam class yang berbeda. Selanjutnya setelah dikelompokkan kedalam voting maka akan dieksekusi lagi sehingga menghasilkan class akhir untuk random forest dari decision tree tersebut.

3.3.2 Praktek

Penyelesaian Tugas Harian 6 (No. 1-8)

1. Aplikasi Sederhana Pandas dan Penjelasan Code (Perbaris)

- Code Pandas:

```

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

```

Figure 3.84: SVM

```

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

```

Figure 3.85: Decision Tree

```

import pandas as fadila
np_array = np.array([10, 20, 30, 40, 50])
print("NumPy array:")
print(np_array)
new_series = fadila.Series(np_array)
print("Converted Pandas series dari Fadila:")
print(new_series)

```

- Penjelasan Code Baris 1 :
 - Mengimport library / module pandas sebagai fadila
- Penjelasan Code Baris 2 :
 - variabel np_array mendefinisikan variabel np.array dimana terdapat beberapa parameter yang akan dieksekusi pada variabel tersebut
- Penjelasan Code Baris 3 :
 - Mendefinisikan perintah print dengan tulisan ” Numpy array ”
- Penjelasan Code Baris 4 :
 - Melakukan perintah print / pencetakan untuk variabel np_array
- Penjelasan Code Baris 5 :
 - Mendefinisikan variabel baru yaitu new_series dimana mengeksekusi fadila.series dengan variabel parameternya ialah np_array
- Penjelasan Code Baris 6 :
 - Mendefinisikan perintah print dengan tulisan ” Converted pandas series dari fadila ”
- Penjelasan Code Baris 7 :
 - Melakukan perintah print / pencetakan terhadap variabel new_series

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

Figure 3.86: Cross Validation 1

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

Figure 3.87: Cross Validation 2

- Hasil Eksekusi :
2. Aplikasi Sederhana Numpy dan Penjelasan Code (Perbaris)

- Code Numpy:

```
import numpy as fadila
x = fadila.ones((3,3))
print("Array Original:")
print(x)
print("0 berada di border sedangkan 1 berada dalam array")
x = fadila.pad(x, pad_width=1, mode='constant', constant_values=0)
print(x)
```

- Penjelasan Code Baris 1 :

Mengimport library / module numpy sebagai fadila

- Penjelasan Code Baris 2 :

Mendefinisikan variabel x dengan mengeksekusi fadila.ones dengan 2 parameter yaitu (3,3)

- Penjelasan Code Baris 3 :

Mendefinisikan perintah print dengan parameter tulisan yaitu (” Array Original ”)

- Penjelasan Code Baris 4 :

Melakukan Perintah print variabel X

- Penjelasan Code Baris 5 :

Mendefinisikan perintah print dengan parameter tulisan yaitu (” 0 berada di border sedangkan 1 berada dalam array ”)

- Penjelasan Code Baris 6 :

Variabel x mengeksekusi fadila.pad dengan 4 variabel parameter yaitu (x, pad_width=1 , mode='constant'], constant_values=0)

```

C:\ProgramData\Anaconda3\lib\site-packages\sklearn\svm\base.py:196: FutureWarning:
The default value of gamma will change from 'auto' to 'scale' in version 0.22 to
account better for unscaled features. Set gamma explicitly to 'auto' or 'scale' to
avoid this warning.
    "avoid this warning.", FutureWarning)
C:\ProgramData\Anaconda3\lib\site-packages\sklearn\svm\base.py:196: FutureWarning:
The default value of gamma will change from 'auto' to 'scale' in version 0.22 to
account better for unscaled features. Set gamma explicitly to 'auto' or 'scale' to
avoid this warning.
    "avoid this warning.", FutureWarning)
C:\ProgramData\Anaconda3\lib\site-packages\sklearn\svm\base.py:196: FutureWarning:
The default value of gamma will change from 'auto' to 'scale' in version 0.22 to
account better for unscaled features. Set gamma explicitly to 'auto' or 'scale' to
avoid this warning.
    "avoid this warning.", FutureWarning)
C:\ProgramData\Anaconda3\lib\site-packages\sklearn\svm\base.py:196: FutureWarning:
The default value of gamma will change from 'auto' to 'scale' in version 0.22 to
account better for unscaled features. Set gamma explicitly to 'auto' or 'scale' to
avoid this warning.
    "avoid this warning.", FutureWarning)
C:\ProgramData\Anaconda3\lib\site-packages\sklearn\svm\base.py:196: FutureWarning:
The default value of gamma will change from 'auto' to 'scale' in version 0.22 to
account better for unscaled features. Set gamma explicitly to 'auto' or 'scale' to
avoid this warning.
    "avoid this warning.", FutureWarning)
Accuracy: 0.27 (+/- 0.02)

```

Figure 3.88: Cross Validation 3

– Penjelasan Code Baris 7 :

Melakukan perintah print / pencetakan variabel X

- Hasil Eksekusi :

3. Aplikasi Sederhana Matplotlib dan Penjelasan Code (Perbaris)

- Code Matplotlib:

```

import matplotlib.pyplot as fadila
# line 1 points
x1 = [10,20,30]
y1 = [20,40,10]
# plotting the line 1 points
fadila.plot(x1, y1, label = "line 1")
# line 2 points
x2 = [10,20,30]
y2 = [40,10,30]
# plotting the line 2 points
fadila.plot(x2, y2, label = "line 2")
fadila.xlabel('x - axis')
# Set the y axis label of the current axis.
fadila.ylabel('y - axis')
# Set a title of the current axes.
fadila.title('Dua atau line lebih pada plot yang sama
              dengan suitable legends ')
# show a legend on the plot
fadila.legend()

```

```

In [49]: max_features_opts = range(5, 50, 5)
...: n_estimators_opts = range(10, 200, 20)
...: rf_params = np.empty((len(max_features_opts)*len(n_estimators_opts),4),
float)
...: i = 0
...: for max_features in max_features_opts:
...:     for n_estimators in n_estimators_opts:
...:         clf = RandomForestClassifier(max_features=max_features,
n_estimators=n_estimators)
...:         scores = cross_val_score(clf, df_train_att, df_train_label, cv=5)
...:         rf_params[i,0] = max_features
...:         rf_params[i,1] = n_estimators
...:         rf_params[i,2] = scores.mean()
...:         rf_params[i,3] = scores.std() * 2
...:         i += 1
...: print("Max features: %d, num estimators: %d, accuracy: %0.2f (+/- %0.2f)" %
(max_features, n_estimators, scores.mean(), scores.std() * 2))
Max features: 5, num estimators: 10, accuracy: 0.26 (+/- 0.02)
Max features: 5, num estimators: 30, accuracy: 0.35 (+/- 0.03)
Max features: 5, num estimators: 50, accuracy: 0.39 (+/- 0.03)
Max features: 5, num estimators: 70, accuracy: 0.41 (+/- 0.02)
Max features: 5, num estimators: 90, accuracy: 0.42 (+/- 0.03)
Max features: 5, num estimators: 110, accuracy: 0.43 (+/- 0.03)
Max features: 5, num estimators: 130, accuracy: 0.43 (+/- 0.03)
Max features: 5, num estimators: 150, accuracy: 0.44 (+/- 0.03)

```

Figure 3.89: Program Pengamatan Komponen Informasi 1

```

# Display a figure .
fadila.show()

```

- Penjelasan Code Baris 1 :
Mengimport library / module matplotlib.pyplot sebagai fadila
- Penjelasan Code Baris 2 :
Mendefinisikan / membuat variabel x1 dengan pengeksekusian 3 parameter yaitu (10,20,30)
- Penjelasan Code Baris 3 :
Mendefinisikan / membuat variabel y1 dengan pengeksekusian 3 parameter yaitu (10,20,30)
- Penjelasan Code Baris 4 :
Mendefinisikan perintah fadila.plot dengan 3 parameter (x1,y1 , label = 'line 1')
- Penjelasan Code Baris 5 :
Mendefinisikan variabel x1 dengan pengeksekusian 3 parameter yaitu (10,20,30)
- Penjelasan Code Baris 6 :
Mendefinisikan / membuat variabel x2 dengan pengeksekusian 3 parameter yaitu (10,20,30)
- Penjelasan Code Baris 7 :

```
In [9]: import matplotlib.pyplot as plt
...: from mpl_toolkits.mplot3d import Axes3D
...: from matplotlib import cm
...: fig = plt.figure()
...: fig.clf()
...: ax = fig.gca(projection='3d')
...: x = rf_params[:,0]
...: y = rf_params[:,1]
...: z = rf_params[:,2]
...: ax.scatter(x, y, z)
...: ax.set_zlim(0.2, 0.5)
...: ax.set_xlabel('Max features')
...: ax.set_ylabel('Num estimators')
...: ax.set_zlabel('Avg accuracy')
...: plt.show()
```

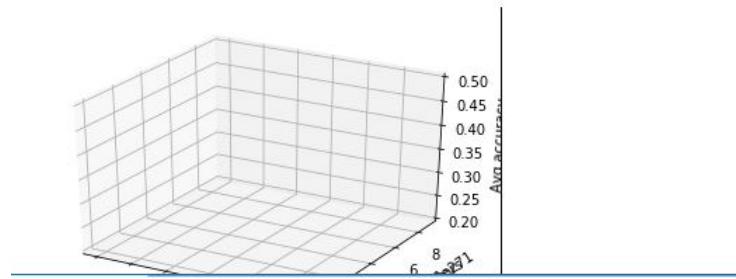


Figure 3.90: Program Pengamatan Komponen Informasi 2

Mendefinisikan / membuat variabel y2 dengan pengeksekusian 3 parameter yaitu (40,10,30)

– Penjelasan Code Baris 8 :

Mendefinisikan perintah fadila.plot dengan 3 parameter yaitu (x2,y2 , label = 'line 2")

– Penjelasan Code Baris 9 :

Mendefinisikan perintah fadila.xlabel dengan parameter (x-axis)

– Penjelasan Code Baris 10 :

Mendefinisikan perintah fadila.ylabel dengan parameter (y-axis)

– Penjelasan Code Baris 11 :

Mendefinisikan perintah fadila.title dengan parameter tulisan yaitu (" dua atau line lebih pada plot yang sama dengan suitable legends ")

– Penjelasan Code Baris 12 :

Mendefinisikan perintah fadila.legend tanpa parameter

– Penjelasan Code Baris 13 :

Mendefinisikan perintah fadila.show untuk menampilkan hasil eksekusi dari variabel fadila

• Hasil Eksekusi :

4. Program Aplikasi Random Forest dan Penjelasan Keluarannya :

```

parser_f
    return _read(filepath_or_buffer, kwds)

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

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

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

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

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

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

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

```

Figure 3.91: Error

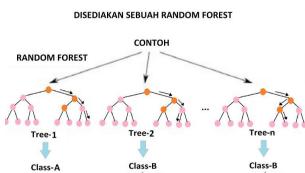


Figure 3.92: random forest

- Code Random Forest 1 :

- Penjelasan : Codingan tersebut akan menghasilkan sebuah variabel baru yaitu ”imgatt” (yang telah dibuat sebelumnya) dengan tipe dataframe dan berupa matrix 2 dimensi. Terdapat 3 kolom dan 3677856 baris data. Bisa di check isi datanya karena berupa data frame. Codingan tersebut ”Membaca” dan ”Menampilkan” data yang berasal dari file ”image_attribute_labels.txt”.

- Code Random Forest 2 :

- Penjelasan : Hasil dari codingan tersebut berfungsi untuk melihat data awal dari data/dataset yang dibaca dan diproses. Lebih jelasnya hanya untuk melihat isi paling atas dari data dan ditampilkan di console saat codingan dieksekusi.

- Code Random Forest 3 :

- Penjelasan : Codingan diatas akan menghasilkan dan juga mena-

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

Figure 3.93: random forest 1

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

Figure 3.94: random forest 2

mpilkan dari jumlah data pada dataset yang telah dieksekusi. Jumlah data/ sizenya akan berupa baris dan kolom dari data yang ada.

- Code Random Forest 4 :
 - Penjelasan : Codingan tersebut menghasilkan tampilan variabel baru yaitu ”imgatt2” . Fungsinya yaitu juga untuk merubah atribut menjadi kolom dengan menggunakan pivot layaknya excel. Yang membedakan antara ”imgatt2” dan ”imgatt1” ialah, jumlah dari data yang ditampilkan berbeda.
- Code Random Forest 5 :
 - Penjelasan : Hasil dari codingan tersebut berfungsi untuk melihat data awal dari data/dataset yang dibaca dan diproses pada variabel” imgatt2 ”. Lebih jelasnya hanya untuk melihat isi paling atas dari data dan ditampilkan di console saat codingan dieksekusi.
- Code Random Forest 6 :
 - Penjelasan : Codingan diatas akan menghasilkan dan juga menampilkan dari jumlah data pada dataset yang telah dieksekusi pada variabel ” imgatt2 ”. Jumlah data/ sizenya akan berupa baris dan kolom dari data yang ada.
- Code Random Forest 7 :
 - Penjelasan : Hasil dari codingan diatas menampilkan atau menunjukkan load dari jawabannya yang berisi ” apakah burung tersebut (subjek pada dataset) termasuk dalam spesies yang mana ?. Kedua kolom yang digunakan adalah imgid dan label, kemudian melakukan pivot yang mana imgid menjadi index yang artinya unik sehubungan dengan dataset yang telah dieksekusi.
- Code Random Forest 8 :

name: A Type: DataFrame Row: 6 Column names: Country, Age, Salary, Purchased

Figure 3.95: random forest 3

Index	Country	Age	Salary	Purchased
0	France	44	72000	No
1	Spain	27	48000	Yes
2	Germany	38	54000	No
3	Spain	38	61000	No
4	Germany	48	nan	YES
5	France	35	36000	Yes

Figure 3.96: random forest 4

- Penjelasan : Hasil dari codingan tersebut berfungsi untuk melihat data awal dari data/dataset yang dibaca dan diproses pada variabel ”imglabels ”. Lebih jelasnya hanya untuk melihat isi paling atas dari data dan ditampilkan di console saat codingan dieksekusi.

- Code Random Forest 9 :

- Penjelasan : Codingan diatas akan menghasilkan dan juga menampilkan dari jumlah data pada dataset yang telah dieksekusi pada variabel ”imglabels ”. Jumlah data/ sizenya akan berupa baris dan kolom dari data yang ada.

- Code Random Forest 10 :

- Penjelasan : Codingan 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 : Codingan ini menghasilkan pemisahan dan pemilihan tabel (memisahkan dan memilih tabel). Pada codingan ini dilakukan kegiatan untuk drop label yang berada di depan kemudian menggunakan label yang berada pada posisi paling belakang yang selanjutnya melakukan join terhadap 2 variabel tersebut yang telah dieksekusi.

- Code Random Forest 12 :

- Penjelasan : Codingan ini menunjukkan hasil pengecekan dari isi variabel df.att bagian head (tampilan awal) dari dataset yang dieksekusi.

Figure 3.97: cross validation

Y	Y Pred	Keluaran Untuk (threshold) 0.6	Recall (pemanggilan/penarikan)	Precision (Presisi)
0	0.5	0		
1	0.9	1		
0	0.7	1		
1	0.7	1	1 / 2	4 / 7
1	0.3	0		
0	0.4	0		
1	0.5	0		

Figure 3.98: confusion matrix

- Code Random Forest 13 :
 - Penjelasan : Codingan ini menunjukkan hasil pengecekan dari isi variabel df.label bagian head (tampilan awal) dari dataset yang dieksekusi.
 - Code Random Forest 14 :
 - Penjelasan : Hasil codingannya menunjukkan pembagian untuk data training dan data testing pada dataset. Dilakukan pembagian data menjadi dua bagian dimana untuk row/baris data pertama sebanyak 8000 dijadikan sebagai data training , kemudian 8000 data sisanya sebagai data testing pada pengeksekusianya.
 - Code Random Forest 15 :
 - Penjelasan : Hasil dari codingan tersebut yaitu instalasi kelas random forest. Dimana pada pemrosesannya, dilakukan pemanggilan kelas Random Forest Classifier (Klasifikasi Random Forest). Pada codingan terdapat max features yang diartikan sebagai berapa banyak kolom pada setiap tree yang dieksekusi pada dataset.
 - Code Random Forest 16 :
 - Penjelasan : Codingan tersebut menunjukkan fitting random forest dengan dataset training. Dimana pengeksekusianya dilakukan fit untuk membangun random forest yang sudah ditentukan dengan maksimum fitur sebanyak 50 untuk perpohnnya (dari dataset yang dieksekusi).

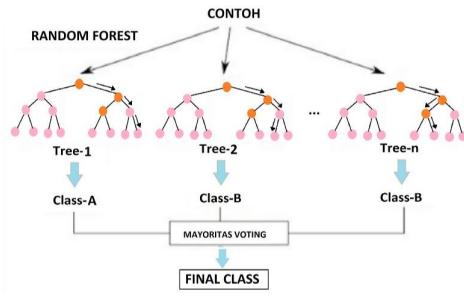


Figure 3.99: voting random forest

```

In [28]: import pandas as fadila
...: np_array = np.array([10, 20, 30, 40, 50])
...: print("NumPy array:")
...: print(np_array)
...: new_series = fadila.Series(np_array)
...: print("Converted Pandas series dari Fadila:")
...: print(new_series)
NumPy array:
[10 20 30 40 50]
Converted Pandas series dari Fadila:
0    10
1    20
2    30
3    40
4    50
dtype: int32
  
```

Figure 3.100: pandas

- Code Random Forest 17 :
 - Penjelasan : Codingan tersebut menunjukkan hasil dari prediksi dataset yang dieksekusi.
- Code Random Forest 18 :
 - Penjelasan : Codingan tersebut menunjukkan score perolehan dari klasifikasi dataset yang dieksekusi. Terdapat hasil besaran akurasi dari code yang dieksekusi sesuai dengan data yang digunakan.

5. Program Aplikasi Confusion Matrix dan Penjelasan Keluarannya :

- Code Confusion Matrix 1 :
 - Penjelasan : Codingan diatas menunjukkan proses dari pembuatan confusion matrix dimana dilakukan pemetaan dan pemetakan confusion matrix terhadap data yang dieksekusi.
- Code Confusion Matrix 2 :
 - Penjelasan : Codingan ini menunjukkan hasil dari pembuatan pemetaan data yang dilakukan sebelumnya (Code confusion matrix 1). Hasilnya berupa matrix yang didalamnya terdapat angka dari hasil eksekusi confusion matrix itu sendiri. Memunculkan array dan dtype dari hasil pembuatan confusion matrix.

```
In [30]: import numpy as fadila
...: x = fadila.ones((3,2))
...: print("Array Original:")
...: print(x)
...: print("0 berada di border sedangkan 1 berada dalam array")
...: x = fadila.pad(x, pad_width=1, mode='constant', constant_values=0)
...: print(x)
Array Original:
[[1. 1.]
 [1. 1. 1.]
 [1. 1. 1. 1.]]
0 berada di border sedangkan 1 berada dalam array
[[0. 0. 0. 0. 0.]
 [0. 1. 1. 1. 0.]
 [0. 1. 1. 1. 0.]
 [0. 1. 1. 1. 0.]
 [0. 0. 0. 0. 0.]]
```

Figure 3.101: numpy

```
In [34]: import matplotlib.pyplot as fadila
...: x1 = [10, 20, 30]
...: y1 = [20, 25, 30]
...: # Plotting the line 1 points
...: fadila.plot(x1, y1, label = 'line 1')
...: x2 = [15, 25, 30]
...: y2 = [10, 20, 30]
...: # Plotting the line 2 points
...: fadila.plot(x2, y2, label = 'line 2')
...: # Set the x-axis label of the current axes.
...: fadila.xlabel('x-axis of the current axes')
...: # Set the y-axis label of the current axes.
...: fadila.ylabel('y-axis of the current axes')
...: # Set the title of the plot
...: fadila.title('Dua atau line lebih pada plot yang sama dengan suitable legends')
...: fadila.legend()
...: fadila.show()
```

Figure 3.102: matplotlib

- Code Confusion Matrix 3 :

- Penjelasan : Codingan diatas melakukan plotting confusion matrix dimana dieksekusi beberapa parameter yang disesuaikan dengan data yang dieksekusi. Code ini hanya melakukan proses plotting dan tidak menunjukkan hasil plotting confusion matrixnya.

- Code Confusion Matrix 4 :

- Penjelasan : Codingan tersebut menunjukkan hasil dari pembacaan file classes.txt yang dieksekusi. Dilakukan plotting terhadap sumbu sesuai dengan nama data dan dilakukan penyettingan (set) sehingga memberikan hasil seperti pada gambar.

- Code Confusion Matrix 5 :

- Penjelasan : Codingan ini menghasilkan plot dari hasil perubahan label. Dimana proses plot yang dilakukan akan merubah label dari classes / data yang dieksekusi.

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

- Code SVM :

Figure 3.103: random forest

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

Figure 3.104: random forest 2

- Penjelasan : Codingan ini menunjukkan klasifikasi dengan decision tree dengan dataset yang sama. Tentunya pada pemrosesannya decision tree digunakan dan menghasilkan output dari klasifikasi tersebut.

- Code Decision Tree :

- Penjelasan : Codingan ini menunjukkan klasifikasi dengan SVM dengan dataset yang sama. Tentunya pada pemrosesannya SVM digunakan dan menghasilkan output dari klasifikasi tersebut.

7. Program Cross Validation dan Penjelasan Keluarannya :

- Code Cross Validation 1 :

- Penjelasan : Codingan tersebut menunjukkan hasil dari Cross Validation Random Forest. Pada pemrosesannya dilakukan pengecekan terhadap Cross Validation untuk Random Forest sesuai dengan data yang dieksekusi.

- Code Cross Validation 2 :

- Penjelasan : Codingan tersebut menunjukkan hasil dari Cross Validation Decision Tree. Pada pemrosesannya dilakukan pengecekan terhadap Cross Validation untuk Decision Tree sesuai dengan data yang dieksekusi.

- Code Cross Validation 3 :

- Penjelasan : Codingan tersebut menunjukkan hasil dari Cross Validation SVM. Pada pemrosesannya dilakukan pengecekan terhadap Cross Validation untuk SVM sesuai dengan data yang dieksekusi.

8. Program Pengamatan Komponen Informasi dan Penjelasan Keluarannya :

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

Figure 3.105: random forest 3

```
In [5]: imgatt2 = imgatt.plot(index='imgID', columns='attID', values='present')
```

Figure 3.106: random forest 4

- Code Pengamatan Komponen Informasi 1 :

- Penjelasan : Codingan ini memberikan hasil pengamatan komponen informasi dimana pada pengeksekusianya dapat diketahui berapa banyak tree yang dibuat, berapa banyak atribut yang dipakai dan informasi lainnya terkait kode yang digunakan.

- Code Pengamatan Komponen Informasi 2 :

- Penjelasan : Codingan ini menunjukkan plot dari komponen informasi agar bisa dibaca dimana pada pemrosesannya plotting dilakukan pada informasi dengan menggunakan kode.

3.3.3 Penanganan Error

Penyelesaian Tugas Harian 6 (Penanganan Error)

(a) Menyelesaikan dan Membahas Penanganan Error :

- Error 1 :

- Penjelasan Error 1 :

Error tersebut disebabkan karena pada bagian codingan ”directory” file yang akan dieksekusi atau dipanggil tidak ada.

```
FileNotFoundException: File b'/data/CUB_200_2011/attributes/
image\_attribute\_labels.txt' does not exist
```

- Penyelesaian Error 1 :

- i. Pertama-tama perhatikan file yang akan dieksekusi yaitu ”image_attribute_labels.txt”

- ii. Pastikan file tersebut berada satu tempat (directory) dengan file bird-identifier.py yang digunakan sebagai codingan untuk pengeksekusian file

- iii. Selanjutnya hapus codingan dan sesuai seperti contoh codingan berikut :

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

Figure 3.107: random forest 5

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

Figure 3.108: random forest 6

```
imgatt = pd.read_csv('image_attribute_labels.txt')
```

iv. Maka tidak akan terjadi error lagi

- Penjelasan Error 2 :

Error tersebut disebabkan karena pada bagian codingan ”directory” file yang akan dieksekusi atau dipanggil tidak ada.

```
FileNotFoundException: File b'/data/CUB_200_2011/attributes/
image\_class\_labels.txt' does not e
```

- Penyelesaian Error 2 :

- Pertama-tama perhatikan file yang akan dieksekusi yaitu ”image_class_labels.txt”

- Pastikan file tersebut berada satu tempat (directory) dengan file bird-identifier.py yang digunakan sebagai codingan untuk pengeksekusian file

- Selanjutnya hapus codingan dan sesuai seperti contoh codingan berikut :

```
imglabels = pd.read_csv('image_class_labels.txt')
```

iv. Maka tidak akan terjadi error lagi

- Penjelasan Error 3 :

Error tersebut disebabkan karena pada bagian codingan ”directory” file yang akan dieksekusi atau dipanggil tidak ada.

```
FileNotFoundException: File b'/data/CUB_200_2011/attributes/cla
```

- Penyelesaian Error 3 :

- Pertama-tama perhatikan file yang akan dieksekusi yaitu ”classes.txt”

- Pastikan file tersebut berada satu tempat (directory) dengan file bird-identifier.py yang digunakan sebagai codingan untuk pengeksekusian file

Figure 3.109: random forest 7

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

Figure 3.110: random forest 8

- iii. Selanjutnya hapus codingan dan sesuai seperti contoh codingan berikut :

```
birds = pd.read_csv('classes.txt')
```

- iv. Maka tidak akan terjadi error lagi

```
In [16]: imglabel1.shape
```

Figure 3.111: random forest 9

```
In [17]: df.att = df.att.drop(['label'])
```

Figure 3.112: random forest 10

```
In [18]: df.att = df.att.drop(['label'])
```

Figure 3.113: random forest 11

```
In [19]: df.att = df.att.drop(['label'])
```

Figure 3.114: random forest 12

```
In [20]: df.att.head()
```

Figure 3.115: random forest 13

```
In [21]: df.att.att = df.att.att.drop(['label'])
```

Figure 3.116: random forest 14

```
In [22]: from sklearn.ensemble import RandomForestClassifier
```

Figure 3.117: random forest 15

```
In [23]: clf.fit(df_train_att, df_train_label)
```

Figure 3.118: random forest 16

```
In [24]: print(clf.predict(df_train_att.head()))
```

Figure 3.119: random forest 17

```
In [25]: clf.score(df_text_att, df_text_label)
```

Figure 3.120: random forest 18

```
In [26]: from sklearn.metrics import confusion_matrix
```

Figure 3.121: confusion matrix 1

```
In [27]: cm
```

Figure 3.122: confusion matrix 2

Figure 3.123: confusion matrix 3

Figure 3.124: confusion matrix 4

```
In [37]: import numpy as np
...     np.set_printoptions(precision=2)
...     np.random.seed(0)
...     plt.imshow(cifar10[0][0], cmap='gray')
...     plt.confusion_matrix(c, classes=birds, normalize=True)
...     plt.show()

Normalized confusion matrix
[[ 0.27  0.18  0.  0.  0. ]
 [ 0.05  0.73  0.  0.  0. ]
 [ 0.08  0.42  0.  0.  0. ]
 [ 0.08  0.42  0.  0.  0. ]
 [ 0.  0.  0.  0.  1. ]]
```

Figure 3.125: confusion matrix 5

Figure 3.126: svm

```
In [50]: from sklearn import tree
...: cftree = tree.DecisionTreeClassifier()
...: cftree.fit(df_train_att, df_train_label)
...: cftree.score(df_test_att, df_test_label)
```

Figure 3.127: decision tree

```
In [46]: from sklearn.model_selection import cross_val_score
...: scores = cross_val_score(clf, df_train[att], df_train_label, cv=5)
...: # Show average score and +/- two standard deviations away (covering 95%)
scores
...: #> [0.44 +/- 0.02]
```

Figure 3.128: cross validation 1

```
In [42]: scoretree = cross_val_score(clftree, df_train_att, df_train_label, cv=5)
scoretree
```

Ergonomics 2020, 3, 120

```
C:\Program Files\Java\jdk1.8.0_111\bin>java -jar lib\package\aliveness\onjava-1.0.0.jar Futurealgos
The default value of goal will change from "had" to "held" on version 0.2.0. To
keep the old behavior, set goal explicitly to "had" or held to avoid this warning.
  (Futurealgos, Futurealgos)
C:\Program Files\Java\jdk1.8.0_111\bin>java -jar lib\package\aliveness\onjava-1.0.0.jar Futurealgos
The default value of goal will change from "had" to "held" on version 0.2.0. To
keep the old behavior, set goal explicitly to "had" or held to avoid this warning.
  (Futurealgos, Futurealgos)
C:\Program Files\Java\jdk1.8.0_111\bin>java -jar lib\package\aliveness\onjava-1.0.0.jar Futurealgos
The default value of goal will change from "had" to "held" on version 0.2.0. To
keep the old behavior, set goal explicitly to "had" or held to avoid this warning.
  (Futurealgos, Futurealgos)
C:\Program Files\Java\jdk1.8.0_111\bin>java -jar lib\package\aliveness\onjava-1.0.0.jar Futurealgos
The default value of goal will change from "had" to "held" on version 0.2.0. To
keep the old behavior, set goal explicitly to "had" or held to avoid this warning.
  (Futurealgos, Futurealgos)
C:\Program Files\Java\jdk1.8.0_111\bin>java -jar lib\package\aliveness\onjava-1.0.0.jar Futurealgos
The default value of goal will change from "had" to "held" on version 0.2.0. To
keep the old behavior, set goal explicitly to "had" or held to avoid this warning.
  (Futurealgos, Futurealgos)
```

Figure 3.130: cross validation 3

Figure 3.131: pengamatan komponen informasi 1

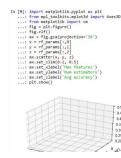


Figure 3.132: pengamatan komponen informasi 2

Figure 3.133: error 1

Figure 3.134: error 2

Figure 3.135: error 3

Chapter 4

Klasifikasi Teks

brief of experiment and result.

4.1 Lusia Violita Aprilian/1164080

4.1.1 Teori

1. Klasifikasi teks

Klasifikasi Dokumen / Teks adalah salah satu tugas penting dan tipikal dalam supervised machine learning (ML). Menetapkan kategori pada dokumen, yang dapat berupa halaman web, buku perpustakaan, artikel media, galeri, dll. Memiliki banyak aplikasi seperti mis. penyaringan spam, perutean email, analisis sentimen dll.

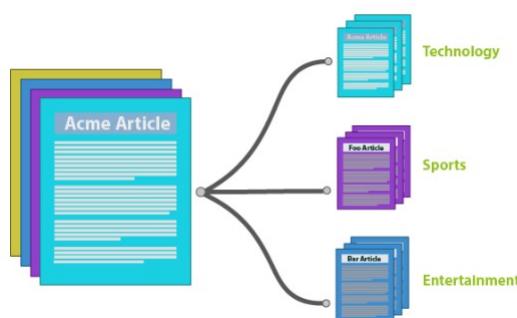


Figure 4.1: Lusia-Klasifikasi teks

2. Klasifikasi Bunga tidak dapat menggunakan machine learning

Klasifikasi bunga tidak dapat menggunakan machine learning karena memiliki masalah input yang serupa namun output yang berbeda atau 'noise'. Yang

dimaksud dengan noise adalah contoh output yang direkam bukan seperti seharusnya. Misalnya saja kita secara implisit berasumsi bahwa contoh bunga kita telah diklasifikasikan dengan benar. Tetapi ini harus dilakukan dengan seseorang yang tepat, seperti seorang ahli botani. Seorang ahli botani ahli harus melihat contoh bunga dan berkata: "ini adalah setosa ... ini adalah virginica", dan dengan demikian bertindak sebagai "guru" yang memungkinkan mesin untuk belajar. Tetapi bagaimana jika guru itu melakukan kesalahan? Selain itu, selalu ada peluang untuk memperkenalkan kesalahan saat merekam data. Noise juga ditemukan dalam pengukuran, yang selalu sedikit bermasalah karena alat dan sensor kami tidak sempurna dan hanya bekerja pada tingkat presisi tertentu.

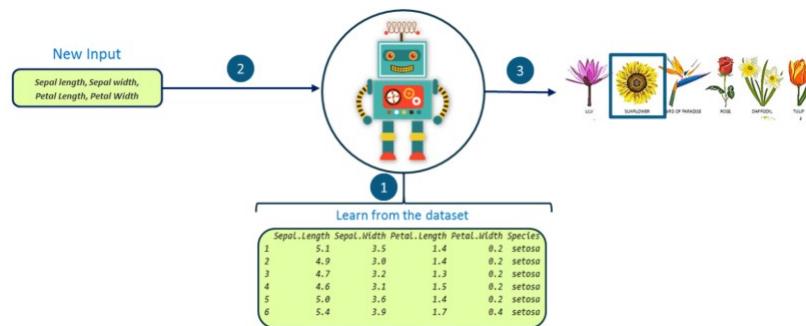


Figure 4.2: Lusia-Klasifikasi bunga

3. Teknik pembelajaran mesin pada teks YouTube

Dengan menggunakan kasus seperti rekomendasi video yang terdapat pada fiturnya, 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.

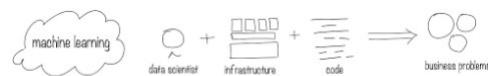


Figure 4.3: Lusia-Teknik YouTube

4. Vectorisasi Data

- Maksud dari Vectorisasi Data merupakan Pemecahan dan Pembagian Data.

5. Bag of word

Bag-of-words adalah cara untuk merepresentasikan data teks saat memodelkan teks dengan algoritma pembelajaran mesin.

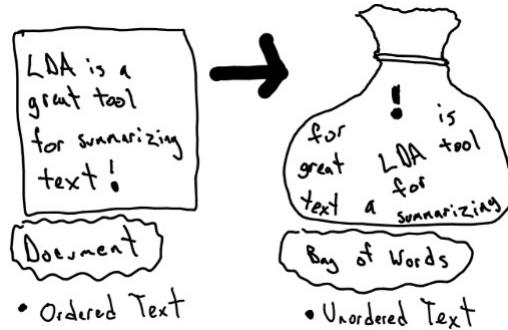


Figure 4.4: Lusia-Bag of Word

6. TF-IDF

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

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.

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

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

Inverse document frequency
of documents

$$\log \frac{1 + n_{\text{docs}}}{1 + df(d, t)} + 1$$

Document frequency of the term t

Figure 4.5: Lusia-TF IDF

4.1.2 Praktek

1. Aplikasi menggunakan pandas

Berikut adalah aplikasi yang dibuat menggunakan pandas :

```
import pandas as pd #memanggil library panda sebagai pd
mas = pd.read_csv('mushrooms.csv', sep=',') #membuat variable untuk membaca file csv
#membuat data frame
df = pd.DataFrame(mas, columns = ['class','cap-shape','cap-surface','cap-color','bruises','odor','g
#%
dummy = pd.get_dummies(df['stalk-surface-above-ring']) #membuat dummy atau memanggil
dummy.head() #memunculkan data teratas
f = df.join(dummy) #join atau menggabung
```

Figure 4.6: Lusia-Pandas

- (a) 1 = memanggil library pandas sebagai pd
- (b) 2 = membuat variable mas untuk membaca file csv
- (c) 3 = membuat variable untuk membuat data frame
- (d) 4 = membuat variable dummy untuk mengubah kategori menjadi integer
- (e) 5 = untuk memunculkan data teratas
- (f) 6 = untuk menjoinkan atau menggabungkan data frame dengan dummy

Berikut adalah hasilnya :

app	DataFrame	(501, 23)	Column names: class, cap-shape, cap-surface, cap-color, bruises, odor, ...
df	DataFrame	(501, 23)	Column names: class, cap-shape, cap-surface, cap-color, bruises, odor, ...
dummy	DataFrame	(501, 2)	Column names: f, s
f	DataFrame	(501, 23)	Column names: class, cap-shape, cap-surface, cap-color, bruises, odor, ...
mas	DataFrame	(501, 23)	Column names: class, cap-shape, cap-surface, cap-color, bruises, odor, ...

Figure 4.7: Lusia-Hasil Pandas

2. Memecah data frame

Berikut untuk memecah data frame menjadi dua :

```
d_train= mas[ :450] #split data training 0-450 data pertama
d_test= mas[450: ] #split data test sisa dari 0-450 data pertama
```

Figure 4.8: Lusia-Pecah data

- (a) 1 = split data training 0-450 data pertama
- (b) 2 = split data test sisa dari 0-450 data pertama

d_test	DataFrame	(51, 23)	Column names: class, cap-shape, cap-surface, cap-color, bruises, odor, ...
d_train	DataFrame	(450, 23)	Column names: class, cap-shape, cap-surface, cap-color, bruises, odor, ...

Figure 4.9: Lusia-Hasil Pecah data

Berikut adalah hasilnya :

3. Vektorisasi dan klasifikasi Decission Tree Katty Perry

- Berikut adalah vektorisasi dan klasifikasi Katty Perry

```
...: import pandas as pd
...: d = pd.read_csv("youtube02-KatyPerry.csv")
...:
...: from sklearn.feature_extraction.text import CountVectorizer
...: vectorizer = CountVectorizer()
...:
...: dvec = vectorizer.fit_transform(d['CONTENT'])
...: dvec
...:
...: daptarkata=vectorizer.get_feature_names()
```

Figure 4.10: Lusia-Vektorisasi dan klasifikasi

Maksud dari gambar vektorisasi dan klasifikasi Katty Perry adalah hasil dari impor dataset, lalu import countvectorizer dari sklearn. Modul sklearn feature extraction digunakan untuk mengekstrak fitur dalam format yang didukung oleh algoritma pembelajaran mesin dari kumpulan data yang terdiri dari format seperti teks dan gambar. Lalu membuat variabel Dan CountVectorizer mengimplementasikan tokenization dan penghitungan kejadian dalam satu kelas. lalu membuat variabel dvec untuk mempelajari dataset. Membuat variabel daptarkata yang berfungsi untuk pemetaan array dari indeks integer fitur ke nama fitur.

- Berikut adalah Decission Tree Katty Perry

```
In [72]: from sklearn import tree
...: clf = tree.DecisionTreeClassifier(criterion="entropy", max_depth=5)
...: clf.fit(d_train_att, d_train_label)
...: clf.predict(d_test_att)
...: clf.score(d_test_att, d_test_label)
Out[72]: 0.86
```

Figure 4.11: Lusia-Decission Tree Katty Perry

Dalam gambar Decission Tree dijelaskan bahwa library tree dari sklearn. Dan mendefinisikan variable untuk memanggil Decission Tree Classifisier yang kemudian dilakukan fit atau pengujian. Lalu menggunakan prediksi dengan fungsi predict untuk memprediksi test. Yang terakhir memunculkan akurasi prediksi yaitu 0,86.

```
In [75]: from sklearn import svm
...: clfsvm = svm.SVC()
...: clfsvm.fit(d_train_att, d_train_label)
...: clfsvm.score(d_test_att, d_test_label)
Out[75]: 0.42
```

Figure 4.12: Lusia-Hasil klasifikasi SVM

4. Klasifikasikan dari data vektorisasi dengan klasifikasi SVM

Berikut adalah klasifikasikan dari data vektorisasi dengan klasifikasi SVM

Dalam gambar SVM dijelaskan bahwa mula-mula file svm diimpor dari sklearn, lalu melakukan fit dari d train att dan d train label atau disebut dengan pengujian. Selanjutnya variable didifinisikan variable untuk melakukan prediksi dataset dengan SVM. Dan yang terakhir muncul hasilnya yaitu 0,42.

5. Klasifikasikan dari data vektorisasi dengan klasifikasi Decission Tree

Maksud dari gambar vektorisasi adalah hasil dari impor dataset

Berikut adalah Decission Tree

```
In [72]: from sklearn import tree
...: clf = tree.DecisionTreeClassifier(criterion="entropy", max_depth=5)
...: clf.fit(d_train_att, d_train_label)
...: clf.predict(d_test_att)
...: clf.score(d_test_att, d_test_label)
Out[72]: 0.86
```

Figure 4.13: Lusia-Decission Tree

Dalam gambar Decission Tree dijelaskan bahwa library tree dari sklearn. Dan mendefinisikan variable untuk memanggil Decission Tree Classifisier yang kemudian dilakukan fit atau pengujian. Lalu menggunakan prediksi dengan fungsi predict untuk memprediksi test. Yang terakhir memunculkan akurasi prediksi yaitu 0,86.

6. Plot confusion matrix

Berikut adalah hasil dari ploting confusion matrix

```
In [30]: import numpy as np
...: np.set_printoptions(precision=2)
...: plt.figure(figsize=(60,60), dpi=300)
...: plot_confusion_matrix(cm, classes=yt, normalize=True)
...: plt.show()
Normalized confusion matrix
[[1. 0.]
 [0.1 0.9]]
```

Figure 4.14: Lusia-ploting confusion matrix

Dari gambar dijelaskan bahwa library numpy di import sebagai np. Lalu Opsi set printoption untuk menentukan cara angka floating point, array dan objek

NumPy lainnya ditampilkan. Selanjutnya matplotlib pyplot figure untuk membuat figur atau gambar baru. Selanjutnya confution matrix dinormalisasikan. Dan yang terakhir hasil ditampilkan.

7. Program cross validation

Berikut adalah hasil dari program cross validation

```
In [24]: from sklearn.model_selection import cross_val_score
...: scores = cross_val_score(clf,d_train_att,d_train_label, cv=5)
...:
...: skorata2=scores.mean()
...: skoresd=scores.std()
```

Figure 4.15: Lusia-Program cross validation

Maksud dari hasil gambar tersebut adalah untuk mendefinisikan dataset untuk 'menguji' model.

8. Program pengamatan komponen informasi

Berikut adalah hasil dari program pengamatan komponen informasi

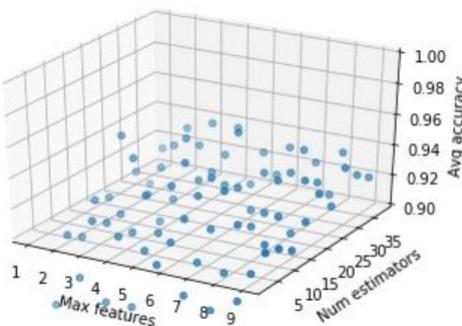


Figure 4.16: Lusia-Program pengamatan komponen informasi

Maksud dari hasil gambar tersebut adalah diagram informasi dari dataset yang digunakan.

4.1.3 Penanganan Error

1. skrinsut error
2. Tuliskan kode eror dan jenis errornya
 - Kode error = KeyError: 'test preparation course'
 - Jenis error = KeyError

```

  File "pandas\_libs\index.pyx", line 140, in
pandas._libs.index.IndexEngine.get_loc
  File "pandas\_libs\index.pyx", line 162, in
pandas._libs.index.IndexEngine.get_loc
  File "pandas\_libs\hashtable_class_helper.pxi", line 1492, in
pandas._libs.hashtable.PyObjectHashTable.get_item
  File "pandas\_libs\hashtable_class_helper.pxi", line 1500, in
pandas._libs.hashtable.PyObjectHashTable.get_item
KeyError: 'test preparation course'

```

Figure 4.17: Lusia-skrinsut error

3. Solusi pemecahan masalah error

Solusinya adalah dengan memasukkan salah satu atribut dari dataset file csv yang digunakan.

4.2 Rahmi Roza-1164085

4.2.1 Teori

Penjelasan Tugas Harian 7 (No 1-6)

1. Pengertian Klasifikasi Teks Dan Ilustrasi Gambar

- Pengertian Klasifikasi Teks

Klasifikasi teks adalah proses pengelompokan benda berdasarkan ciri-ciri persamaan dan perbedaan dengan pemberian tag atau kategori ke teks sesuai dengan isinya.

- Ilustrasi Gambar

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

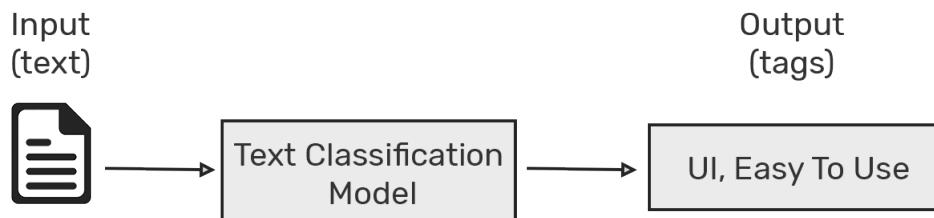


Figure 4.18: Klasifikasi Teks Roza

2. Mengapa Klasifikasi Bunga Tidak Bisa Menggunakan Machine Learning Dan Ilustrasi Gambar

- Mengapa Klasifikasi Bunga Tidak Bisa Menggunakan Machine Learning

Penjelasan : Klasifikasi bunga tidak bisa digunakan pada machine learning karena terdapat masalah input yang sama dan output yang berbeda. Output atau error disebut noise.

- Ilustrasi Gambar

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

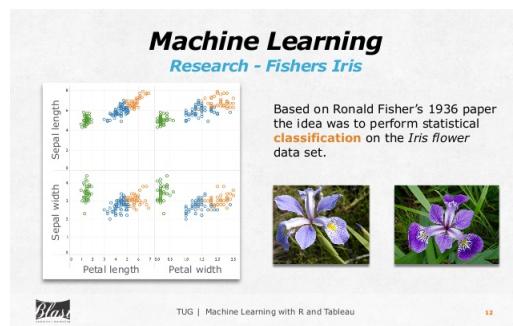


Figure 4.19: Klasifikasi Bunga Roza

3. Teknik Pembelajaran Mesin Pada Teks Pada Kata-Kata Yang Digunakan Di Youtube Dan Ilustrasi Gambar

- Teknik Pembelajaran Mesin Pada Teks Pada Kata-Kata Yang Digunakan Youtube

Penjelasan : Penggunaan Machine Learning pada Youtube yaitu contohnya pada saat kita melakukan searching vidio atau pun bahan lainnya di youtube maka yang akan ditampilkan adalah vidio dari keyword yang kita ketikkan di kotak pencarian. Dengan kata lain Youtube akan memfilter vidio dengan keyword yang telah digunakan. Dan pada saat kita menonton Youtube pada bagian sebelah kanan tampilan youtuber terdapat vidio yang berkaitan dengan keyword yang kita cari tadi. Dimana disitulah Machine Learning pada Youtube menyimpan data.

- Ilustrasi Gambar

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

```

In [8]: totalMails = mails['message'].shape[0]
trainIndex, testIndex = list(), list()
for i in range(mails.shape[0]):
    if np.random.uniform(0, 1) < 0.75:
        trainIndex += [i]
    else:
        testIndex += [i]
trainData = mails.loc[trainIndex]
testData = mails.loc[testIndex]

In [9]: trainData.reset_index(inplace = True)
trainData.drop(['index'], axis = 1, inplace = True)
trainData.head()

Out[9]:
```

	message	label
0	Go until jurong point, crazy.. Available only ...	0
1	Free entry in 2 a wkly comp to win FA Cup fina...	1
2	U dun say so early hor... U c already then say...	0
3	FreeMsg Hey there darling it's been 3 week's n...	1
4	As per your request 'Melle Melle (Oru Minnamin...	0

Figure 4.20: Youtube Roza

4. Vektorisasi Data

- Maksud Dari Vektorisasi Data

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

5. Pengertian Bag Of Words Dan Ilustrasi Gambar

- Pengertian Bag Of Words

Bag Of-Words adalah sebuah konsep yang diambil dari analisis teks yaitu mempresentasikan dokumen sebagai sebuah kantung informasi-infromasi penting tanpa mengurutkan setiap katanya.

- Ilustrasi Gambar

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

6. Pengertian TF-IDF Dan Ilustrasi Gambar

- Pengertian TF-IDF

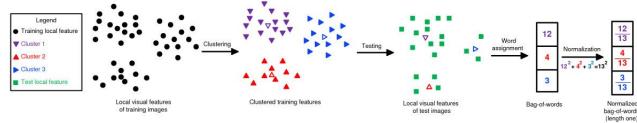


Figure 4.21: Bag of Words Roza

TF-IDF (Term Frequency – Inverse Document Frequency) adalah sebuah algoritma adalah salah satu algoritma yang dapat digunakan untuk menganalisa hubungan antara sebuah frase/kalimat dengan sekumpulan dokumen.

Inti utama dari algoritma ini adalah melakukan perhitungan nilai TF dan nilai IDF dari sebuah setiap kata kunci terhadap masing-masing dokumen. Nilai TF dihitung dengan rumus $TF = \frac{\text{jumlah frekuensi kata terpilih}}{\text{jumlah kata}}$ dan nilai IDF dihitung dengan rumus $IDF = \log(\frac{\text{jumlah dokumen}}{\text{jumlah frekuensi kata terpilih}})$. Selanjutnya adalah melakukan perkalian antara nilai TF dan IDF untuk mendapatkan jawaban akhir.

- Ilustrasi Gambar

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

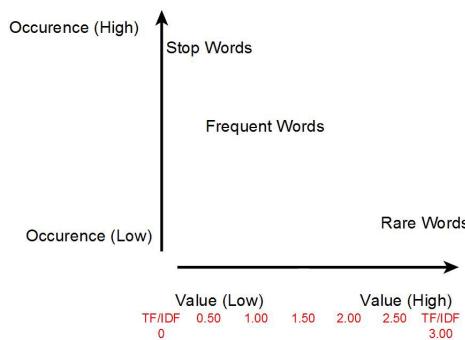


Figure 4.22: TF-IDF Rroza

4.2.2 Praktek

- Membuat Aplikasi Sederhana menggunakan pandas, dan membuat data dummy sebanyak 500 baris dan melakukan data load ke data frame panda. Jelaskan arti perbaris!

```

%% 1
import pandas as pd
bl = pd.read_csv('balancee.csv', sep=',')
df = pd.DataFrame(bl, columns = ['class-name','left-weight','left-distance','right-weight','right-distance'])

%% Data Dummy
dummy = pd.get_dummies (df['class-name'])
dummy.head()
#dummy=mengubah categorical menjadi integer
#dummy.head menampilkan 5 data teratas

```

Figure 4.23: Nomor 1 Roza

- Baris 1: Memanggil library pandas sebagai pd
- Baris 2: Membaca dataset Balance Scale
- Baris 3: Mengambil data frame dari library pandas
- Baris 4: Data Dummy digunakan untuk mengubah Categorical menjadi Integer
- Baris 5: Menampilkan 5 data teratas

(b) GAMBAR HASIL No1:

bl	DataFrame (501, 5)	Column names: class-name, left-weight, left-distance, right-weight, ri ...
----	--------------------	--

Figure 4.24: Hasil 1 Roza

dummy	DataFrame (501, 3)	Column names: B, L, R
-------	--------------------	-----------------------

Figure 4.25: Hasil 1 Roza

- (c) Membuat Aplikasi Sederhana menggunakan pandas, dan membuat data dummy sebanyak 500 baris dan melakukan data load ke data frame panda. Jelaskan arti perbaris.

```

%% 2
d_train= bl[:450]
d_test= bl[450:]

```

Figure 4.26: Nomor 2 Roza

- Baris 1: Membagi data set balance scale dari 500 data menjadi data train menjadi 450 data.
- Baris 2: Membagi data set balance scale dari sisa pembagian data train menjadi data test menjadi 50 data.

```
In [50]: d_train= bl[:450]
...: d_test= bl[450:]
```

Figure 4.27: Hasil 2 Roza

```
In [41]: import pandas as pd
...: bl = pd.read_csv('Youtube03-LMFAO.csv', sep=',')
```

Figure 4.28: Nomor 3 Roza

```
In [11]: from sklearn import tree
...: clf = tree.DecisionTreeClassifier(criterion="entropy", max_depth=5)
...: clf = clf.fit(d_train_att, d_train_label)
...
...: #dari library sklearn import RandomForestClassifier
...: #variabel clf membaca RandomClassi

In [12]: clf.predict(d_test_att)
Out[12]:
array([0, 0, 1, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 1, 1, 1, 0, 1, 1, 1,
       0, 1, 0, 1, 0, 1, 1, 1, 0, 0, 1, 1, 1, 0, 0, 0, 0, 0, 1, 1, 0,
       1, 1, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 1, 0,
       1, 1, 0, 1, 1, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0,
       1, 0, 1, 1, 1, 0, 1, 0, 1, 1, 0, 1, 0, 1, 1, 0, 0, 0, 0, 0, 1, 1, 0,
       1, 0, 1, 1, 1, 0, 1, 0, 0, 0, 1, 0, 1, 1, 0, 1, 0, 1, 0, 0, 0, 0, 1, 1, 0,
       0, 1, 1, 1, 0, 0, 1, 1, 0, 0, 0, 1, 0, 1, 1, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0,
```

Figure 4.29: Hasil 3 Roza

(d) Vektorisasi dan Klasifikasi Data Dengan Decission Tree

- Penjelasan:
 - Dalam in 11 impor tree dari sklearn. Dan mendefinisikan variabel clf untuk memanggil Decission Tree Classifier dan melakukan fit atau pengujian.
 - Dalam in 12 menggunakan prediksi untk clf dengan function predict untuk memprediksi test. Dan hasilnya muncul dalam bentuk array.
 - clf score memunculkan akurasi prediksi yang dilakukan terhadap clf

(e) Klasifikasi SVM

```
In [14]: from sklearn import svm
...: clfsvm = svm.SVC()
...: clfsvm.fit(d_train_att, d_train_label)
...:
...: clfsvm.score(d_test_att, d_test_label)
C:\Users\ROZA\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[14]: 0.6884057971014492
```

Figure 4.30: Nomor 4 Roza

- Import SVM dari sklearn
- Melakukan fit dari d_train_att dan d_train_label atau disebut dengan pengujian
- Mendefinisikan variabel df untuk melakukan prediksi dataset Youtube LMFAO dengan SVM. Dan akan muncul hasil prediksinya

(f) Klasifikasi Decision Tree

```
In [17]: from sklearn import tree
...: clf = tree.DecisionTreeClassifier()
...: clf = clf.fit(d_train_att, d_train_label)
...: clf.predict(d_test_att)
...: clf.score(d_test_att, d_test_label)
Out[17]: 0.9492753623188406
```

Figure 4.31: Nomor 5 Roza

- Penjelasan:
- Dalam in 17 impor tree dari sklearn. Dan mendefinisikan variabel clf untuk memanggil Decision Tree Classifier dan melakukan fit atau pengujian.
- menggunakan prediksi untuk clf dengan function predict untuk memprediksi test. Dan hasilnya muncul dalam bentuk array.
- clf score memunculkan akurasi prediksi yang dilakukan terhadap clf

(g) Matplotlib

```
In [48]: 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.
```

Figure 4.32: Nomor 6 Roza

- Penjelasan:

Fungsi ini mencetak dan memplot Confussion Matrix. Normalisasi dapat diterapkan dengan mengatur ‘normalize = True’. Ada Confusion Matrik menggunakan normalisasi dan ada confussion matrik tidak menggunakan normalisasi.

(h) Menjelaskan Program Cross Validation

```
In [37]: from sklearn.model_selection import cross_val_score
...: scores = cross_val_score(clf,d_train_att,d_train_label, cv=5)
...:
...: skorrata2=scores.mean()
...: skoresd=scores.std()
```

Figure 4.33: Nomor 7 Roza

- Penjelasan: Variabel score akan melakukan cross validation pada variabel clf, d train att, dan train label. Variabel skorrata2 akan menghitung nilai rata-rata dari variabel scores tadi menggunakan function mean. Dan scoresd menghitung standar deviasi dari data yang diberikan.
- GAMBAR HASIL No7:

```
In [53]: from sklearn.model_selection import cross_val_score
...: scores = cross_val_score(clf,d_train_att,d_train_label, cv=5)
...:
...: skorrata2=scores.mean()
...: skoresd=scores.std()
```

Figure 4.34: Hasil 7 Roza

skoresd	float64	1	0.010208097193137975
skorrata2	float64	1	0.9500490877095491

Figure 4.35: Hasil 7 Roza

(i) Program Pengamatan Komponen

- Penjelasan: Pada gambar pertama merupakan kodingan untuk mencetak data dari maxfeatures, num estimators dan accuracy. Sedangkan pada gambar yang kedua itu merupakan gambarnya. Atau hasil gambarnya.

4.2.3 Penanganan Error

- i. Skrinsut Error
- ii. Kode Error dan Jenis Errornya

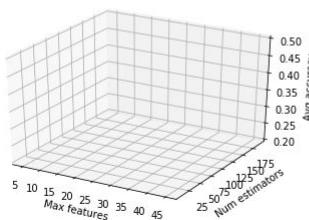
Kode Error: ”FileNotFoundException” dan ”File b ‘balancee.csv’ does not exist”.

Jenis Error: Import Dataset

```
In [56]: max_features_opts = range(5, 50, 5)
...: n_estimators_opts = range(10, 200, 20)
...: rf_params = np.empty((len(max_features_opts)*len(n_estimators_opts),4), float)
...: i = 0
...: for max_features in max_features_opts:
...:     for n_estimators in n_estimators_opts:
...:         clf = RandomForestClassifier(max_features=max_features, n_estimators=n_estimators)
...:         scores = cross_val_score(clf, d_train_att, d_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: %.2f (+/- %.2f)" %
...: (max_features, n_estimators, scores.mean(), scores.std() * 2))
Max features: 5, num estimators: 10, accuracy: 0.91 (+/- 0.11)
Max features: 5, num estimators: 30, accuracy: 0.91 (+/- 0.07)
Max features: 5, num estimators: 50, accuracy: 0.93 (+/- 0.08)
Max features: 5, num estimators: 70, accuracy: 0.93 (+/- 0.07)
Max features: 5, num estimators: 90, accuracy: 0.93 (+/- 0.06)
Max features: 5, num estimators: 110, accuracy: 0.94 (+/- 0.06)
Max features: 5, num estimators: 130, accuracy: 0.94 (+/- 0.06)
Max features: 5, num estimators: 150, accuracy: 0.93 (+/- 0.07)
Max features: 5, num estimators: 170, accuracy: 0.93 (+/- 0.07)
Max features: 5, num estimators: 190, accuracy: 0.94 (+/- 0.07)
Max features: 10, num estimators: 10, accuracy: 0.93 (+/- 0.03)
Max features: 10, num estimators: 30, accuracy: 0.93 (+/- 0.09)
```

Figure 4.36: Nomor 8 Roza

```
...: x = rf_params[:,0]
...: y = rf_params[:,1]
...: z = rf_params[:,2]
...: ax.scatter(x, y, z)
...: ax.set_zlim(0.2, 0.5)
...: ax.set_xlabel('Max features')
...: ax.set_ylabel('Num estimators')
...: ax.set_zlabel('Avg accuracy')
...: plt.show()
```



In [58]:

Figure 4.37: Nomor 8 Roza

```

...: df = pd.DataFrame(bl, columns = ['class-name','left-weight','left-distance','right-
weight','right-distance'])
Traceback (most recent call last):
  File "<ipython-input-54-4858d7fef122>", line 2, in <module>
    bl = pd.read_csv('balancee.csv', sep=',')
  File "C:\Users\ROZA\Anaconda3\lib\site-packages\pandas\io\parsers.py", line 678, in parser_f
    return _read(filepath_or_buffer, kwds)
  File "C:\Users\ROZA\Anaconda3\lib\site-packages\pandas\io\parsers.py", line 440, in _read
    parser = TextFileReader(filepath_or_buffer, **kwds)
  File "C:\Users\ROZA\Anaconda3\lib\site-packages\pandas\io\parsers.py", line 787, in __init__
    self._make_engine(self.engine)
  File "C:\Users\ROZA\Anaconda3\lib\site-packages\pandas\io\parsers.py", line 1014, in _make_engine
    self._engine = CParserWrapper(self.f, **self.options)
  File "C:\Users\ROZA\Anaconda3\lib\site-packages\pandas\io\parsers.py", line 1708, in __init__
    self._reader = parsers.TextReader(src, **kwds)
  File "pandas\_libs\parsers.pyx", line 384, in pandas._libs.parsers.TextReader.__cinit__
  File "pandas\_libs\parsers.pyx", line 695, in pandas._libs.parsers.TextReader._setup_parser_source
FileNotFoundError: File b'balancee.csv' does not exist

```

Figure 4.38: Error Roza

iii. Penanganan

Import ulang dataset dan sesuaikan dengan letak dataset pada file explorer.

4.3 Fadila-1164072

4.3.1 Teori

Penjelasan Tugas Harian 7 (No 1-6)

1. Pengertian Klasifikasi Teks Dan Ilustrasi Gambar

- Pengertian Klasifikasi Teks

Klasifikasi teks adalah proses pemberian tag atau kategori ke teks sesuai dengan isinya. Teks dapat menjadi sumber informasi yang sangat kaya, tetapi mengekstraksi wawasan darinya bisa sulit dan memakan waktu karena sifatnya yang tidak terstruktur.

- Ilustrasi Gambar

Penjelasan : Berdasarkan pengertian diatas, ada beberapa contoh yang bisa diterapkan. Untuk salah satu contoh dari klasifikasi data sendiri dapat diliat pada gambar berikut 6.10.

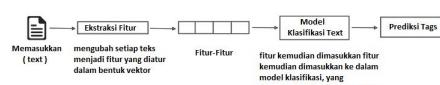


Figure 4.39: text-fadila

2. Mengapa Klasifikasi Bunga Tidak Bisa Menggunakan Machine Learning Dan Ilustrasi Gambar

- Mengapa Klasifikasi Bunga Tidak Bisa Menggunakan Machine Learning

Penjelasan : Untuk klasifikasi bunga tidak dapat menggunakan machine learning dikarenakan memiliki masalah input yang sama namun keluarannya (output) yang berbeda, biasanya output atau error ini disebut dengan istilah 'noise'. Noise sendiri merupakan output yang disimpan / ditangkap maupun direkam bukan seperti seharusnya (keluaran yang diinginkan).

Apabila diberikan contoh, maka contohnya yaitu kita berasumsi secara implisit bahwa klarifikasi bunga yang kita lakukan sudah tepat dan kita melakukannya seperti seorang ahli tanaman. Namun pada hasilnya masih saja terjadi kesalahan. Selain itu, selalu ada peluang untuk memperkenalkan kesalahan saat merekam ataupun menyimpan data, maka harus dilakukan penelitian yang lebih rinci sehingga tidak menimbulkan 'noise' itu sendiri.

- Ilustrasi Gambar

Penjelasan : Berdasarkan pengertian diatas, ada beberapa contoh yang bisa diterapkan. Untuk salah satu contoh dari klasifikasi bunga sendiri dapat diliat pada gambar berikut 4.40.

3. Train model using data with known desired output						
Feature	Flower 1		Flower 2			
	Mean	Min	Max	Mean	Min	
sepal length	5.4	3.0	7.9	6.30	5.00	7.90
sepal width	3.9	1.3	5.8	0.350	0.200	0.700
petal length	1.7	1.0	4.3	0.30	0.10	0.30
petal width	0.4	0.3	1.5	0.70	0.10	1.30
	↓	↓	↓	↓	↓	↓
Classification	iris setosa	iris versicolor		Regression	3	7

Figure 4.40: bunga-fadila

3. Teknik Pembelajaran Mesin Pada Teks Pada Kata-Kata Yang Digunakan Di Youtube Dan Ilustrasi Gambar

- Teknik Pembelajaran Mesin Pada Teks Pada Kata-Kata Yang Digunakan Youtube

Penjelasan : Kita ambil sebuah kasus yang semua orang telah ketahui dan juga pahami. Kasus tersebut yaitu perekondasian video dari pencarian menggunakan "text / kata" di Youtube. Pada saat menggunakan Youtube terdapat Machine Learning yang bekerja dan memproses perintah ataupun aktivitas tersebut, dimana akan memfilter secara otomatis

video yang disesuaikan dengan "keyword" yang kita masukkan sehingga memberikan keluaran video dengan keyword yang benar.

Adapula fitur yang di dapatkan ketika sedang menonton Youtube. Tampilan sebelah kanan terdapat pilihan 'Next' atapun 'Suggestion' yang menampilkan beberapa video serupa sesuai dengan yang dicari atau sedang ditonton. Ketika mengklik salah satu video dari baris tersebut, maka Youtube akan mengingat dan menggunakan kata yang tertera sebagai referensi kembali sehingga akan memberikan kemudahan pada pencarian yang lainnya. Dan disitulah mesin belajar sendiri dan menyimpan data secara berkala sehingga berkembang.

- Ilustrasi Gambar

Penjelasan : Berdasarkan pengertian diatas, ada beberapa contoh yang bisa diterapkan. Untuk salah satu contoh dari Mesin Teks Youtube sendiri dapat diliat pada gambar berikut 4.41.

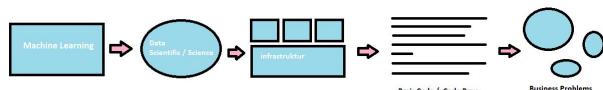


Figure 4.41: youtube-fadila

4. Vektorisasi Data

- Maksud Dari Vektorisasi Data

Penjelasan : Pembagian dan pemecahan data, kemudian dilakukan perhitungan. Vektorisasi juga dapat dimaksudkan dengan setiap data yang mungkin dipetakan ke integer tertentu. jika kita memiliki array yang cukup besar maka setiap kata / data cocok dengan slot unik dalam array (nilai pada indeks adalah nomor satu kali kata itu muncul).

Array angka floating point (Mewakili data) :

- teks
- audio
- gambar

Contoh : -[1.0, 0.0, 1.0, 0.5]

5. Pengertian Bag Of Words Dan Ilustrasi Gambar

- Pengertian Bag Of Words

bag-of-words adalah representasi penyederhanaan yang digunakan dalam pemrosesan bahasa alami dan pengambilan informasi. Model bag-of-words sederhana untuk dipahami dan diterapkan dan telah melihat kesuksesan besar dalam masalah seperti pemodelan bahasa dan klasifikasi dokumen (penyelesaian dll).

- Ilustrasi Gambar

Penjelasan : Berdasarkan pengertian diatas, ada beberapa contoh yang bisa diterapkan. Untuk salah satu contoh dari Bag Of Words sendiri dapat diliat pada gambar berikut 6.6.

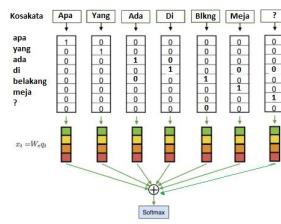


Figure 4.42: bag-fadila

6. Pengertian TF-IDF Dan Ilustrasi Gambar

- Pengertian TF-IDF

TF-IDF atau TFIDF, adalah kependekan dari istilah frekuensi dokumen terbalik, dimana merupakan statistik numerik yang dimaksudkan untuk mencerminkan betapa pentingnya sebuah kata untuk sebuah dokumen dalam kumpulan atau kumpulan.

Nilai tf-idf meningkat secara proporsional dengan berapa kali sebuah kata muncul dalam dokumen dan diimbangi dengan jumlah dokumen dalam korpus yang mengandung kata, yang membantu menyesuaikan fakta bahwa beberapa kata muncul lebih sering secara umum

- Ilustrasi Gambar

Penjelasan : Berdasarkan pengertian diatas, ada beberapa contoh yang bisa diterapkan. Untuk salah satu contoh dari TF-IDF sendiri dapat diliat pada gambar-gambar berikut 4.43.

SCORE TF-IDF (RUMUS / PERHITUNGAN)

$$TF - IDF \text{ Score} = TF_{xy} * IDF = TF_{xy} * \log \frac{N}{df} \dots \dots (1)$$

Figure 4.43: tf2-fadila

4.3.2 Praktek Fadila

Pengerjaanya Tugas Harian 8 (No 1-8)

1. Pembuatan Aplikasi Pandas Sederhana Dengan Menggunakan Dataset berisi 500 baris data.

- Penjelasan Code Pandas Sederhana

```
import pandas as pd
fadila_car = pd.read_csv('1.csv', sep=';')
len(fadila_car)

fadila = pd.DataFrame(fadila_car, columns = [ 'buying', 'maint',
dummy = pd.get_dummies ( fadila [ 'class' ] )
dummy. head()
```

- (a) Baris 1 : Mengimport Library Pandas sebagai pd
- (b) Baris 2 : Membuat variabel fadila_car dimana membaca dataset berupa format csv "1.csv" dengan separator.
- (c) Baris 3 : Menampilkan hasil dari variabel fadila_car (berupa jumlah/angka karena len)
- (d) Baris 4 : Membuat variabel fadila dimana mengambil/ memanggil DataFrame dari library pd (pandas) dengan parameter variabel.
- (e) Baris 5 : Mendefinisikan variabel dummy dimana memanggil get_dummies dari pd dengan parameter variabel fadila dan column class
- (f) Baris 6 : Menampilkan 5 data teratas pada kolom class dari variabel fadila

- Hasil Dari Aplikasi Sederhana Code Pandas

Penjelasan : Hasilnya dapat diliat pada gambar berikut 4.44, 4.45, 4.46 dan 4.47.

2. Pemecahan dataframe menjadi dua dataframe yaitu 450 row pertama dan 50 data sisa.

```

In [35]: # Load dataset (German Portuguese scores)
import pandas as pd
fadila_car = pd.read_csv('1.csv', sep=';')
len(fadila_car)
Out[35]: 5001
fadila = pd.DataFrame(fadila_car, columns = ['buying', 'maint', 'doors', 'persons', 'lug-boot', 'safety', 'class'])
Out[36]: 7
dummy = pd.get_dummies (fadila['class'])
dummy.head()

```

Figure 4.44: 1-fadila

Name	Type	Size	Value
dummy	DataFrame	(5001, 2)	Column names: acc, unacc
fadila	DataFrame	(5001, 7)	Column names: buying, maint, doors, persons, lug-boot, safety, class

Figure 4.45: lanjutan1-fadila

```

In [40]: dummy = pd.get_dummies (fadila['class'])
...: dummy.head()
Out[40]:
acc  unacc
0  0  1
1  0  1
2  0  1
3  0  1
4  0  1

```

Figure 4.46: lanjutan1-1-fadila

fadila (DataFrame)									
Index	buying	maint	doors	persons	lug-boot	safety	class	0	1
0	vhigh	vhigh	2	2	small	low	unacc	0	1
1	vhigh	vhigh	2	2	small	med	unacc	1	1
2	vhigh	vhigh	2	2	small	high	unacc	2	1
3	vhigh	vhigh	2	2	med	low	unacc	3	1
4	vhigh	vhigh	2	2	med	med	unacc	4	1
5	vhigh	vhigh	2	2	med	high	unacc	5	1
6	vhigh	vhigh	2	2	big	low	unacc	6	1
7	vhigh	vhigh	2	2	big	med	unacc	7	1
8	vhigh	vhigh	2	2	big	high	unacc	8	1
9	vhigh	vhigh	2	3	small	low	unacc	9	1
10	vhigh	vhigh	2	3	small	med	unacc	10	1
11	vhigh	vhigh	2	3	small	high	unacc	11	1
12	vhigh	vhigh	2	3	med	low	unacc	12	1
13	vhigh	vhigh	2	3	med	med	unacc	13	1
14	vhigh	vhigh	2	3	med	high	unacc	14	1
15	vhigh	vhigh	2	4	big	low	unacc	15	1
16	vhigh	vhigh	2	4	big	med	unacc	16	1
17	vhigh	vhigh	2	4	big	high	unacc	17	1
18	vhigh	vhigh	3	2	small	low	unacc	18	1
19	vhigh	vhigh	3	2	small	med	unacc	19	1
20	vhigh	vhigh	3	2	small	high	unacc	20	1

Figure 4.47: lanjutan1-2-fadila

- Dataframe 1 : 450 row data pertama

- Dataframe 2 : 50 row data sisanya.

- Codingan Dan Penjelasannya :

```
fadila_car_train = fadila_car [:450]
fadila_car_test = fadila_car [451:]
```

- (a) Baris 1 : Membuat variabel fadila_car_train dari variabel fadila_car dengan 450 data awal untuk dijadikan data training
- (b) Baris 2 : Membuat variabel fadila_car_test dari variabel fadila_car dengan 50 data sisanya untuk data testing (dimulai dari data ke 451 sampai akhir data)

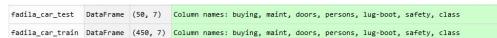
- Hasil Dari Pemecahan Dataframe

Penjelasan : Hasilnya dapat diliat pada gambar berikut 4.48 dan 4.49.



```
# split training and testing data
fadila_car_train = fadila_car[:450]
fadila_car_test = fadila_car[451:]
```

Figure 4.48: 2-fadila



```
fadila_car_test DataFrame (50, 7) Column names: buying, maint, doors, persons, lug-boot, safety, class
fadila_car_train DataFrame (450, 7) Column names: buying, maint, doors, persons, lug-boot, safety, class
```

Figure 4.49: lanjutan2-fadila

3. Vektorisasi Klasifikasi Data NPM mod 4 (Katy Perry) Dengan Decision Tree

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

Penjelasan : Sesuai Hasil.

- (a) Dalam [in 21] merupakan hasil dari impor tree dari sklearn. Dan mendefinisikan variabel clf untuk memanggil Decission Tree Classifier dan melakukan fit atau pengujian.
- (b) Dalam [in 22] menjelaskan bahwa digunakan prediksi untk clf dengan function predict untuk memprediksi test. Dan hasilnya muncul dalam bentuk array.
- (c) Kemudian Dalam [in 23] clf score memunculkan akurasi prediksi yang dilakukan terhadap clf

- Hasil Program Menggunakan Decision Tree

Hasilnya dapat diliat pada gambar berikut 4.50.

```
In [21]: from sklearn import tree
...: clf = tree.DecisionTreeClassifier(criterion="entropy", max_depth=5)
...: clf = clf.fit(d_train_att, d_train_label)

In [22]: clf.predict(d_test_att)
Out[22]:
array([1, 0, 0, 1, 1, 0, 0, 1, 1, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 1,
       0, 1, 0, 0, 1, 1, 0, 1, 0, 0, 1, 1, 0, 1, 1, 0, 1, 0, 1, 0, 1, 0,
       0, 1, 0, 0, 0, 0, 0], dtype=int64)

In [23]: clf.score(d_test_att, d_test_label)
Out[23]: 0.92
```

Figure 4.50: 3-fadila

4. Vektorisasi Dengan Klasifikasi SVM

```
from sklearn import svm
clfsvm = svm.SVC()
clfsvm.fit(d_train_att, d_train_label)

clfsvm.score(d_test_att, d_test_label)
```

Penjelasan :

- Dilakukan pengimportan module SVM dari sklearn
- Selanjutnya melakukan fit dari d train att dan d train label atau disebut dengan pengujian
- Mendefinisikan variabel df untuk melakukan prediksi dataset Youtube Katy Perry dengan SVM. Dan akan muncul hasil prediksinya seperti pada contoh gambarnya

- Hasil Klasifikasi SVM

Hasilnya dapat diliat pada gambar berikut 4.51.

```
In [27]: 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\envs\py36\lib\site-packages\sklearn\svm\base.py:195: FutureWarning: The default value of gamma will change from 'scale' to 'scale' in version 0.22 to account
better for unscaled features. Set gamma explicitly to 'auto' or 'scale' to avoid this
warning.
  "Set gamma explicitly to 'auto' or 'scale' to avoid this
warning.", FutureWarning)
Out[27]: 0.96
```

Figure 4.51: 4-fadila

5. Vektorisasi Menggunakan Klasifikasi Decision Tree

```
from sklearn import tree
clf = tree.DecisionTreeClassifier()
clf = clf.fit(d_train_att, d_train_label)
clf.predict(d_test_att)
clf.score(d_test_att, d_test_label)
```

Penjelasan :

- (a) Dalam [in 31] merupakan hasil dari impor tree dari sklearn. Dan mendefinisikan variabel clf untuk memanggil Decission Tree Classifier dan melakukan fit atau pengujian.
- (b) Di dalamnya juga menjelaskan bahwa digunakan prediksi untuk clf dengan function predict untuk memprediksi test. Dan hasilnya muncul dalam bentuk array.
- (c) clf score memunculkan akurasi prediksi yang dilakukan terhadap clf
- (d) Lalu muncul lah hasilnya pada [out 31] seperti pada gambar

- Hasil Program Cross Validation

Hasilnya dapat diliat pada gambar berikut 4.52.

```
In [31]: from sklearn import tree
... clf = tree.DecisionTreeClassifier()
... clf = clf.fit(d_train_att, d_train_label)
... clf.predict(d_test_att)
... clf.score(d_test_att, d_test_label)
Out[31]: 1.0
```

Figure 4.52: 5-fadila

6. Plot Confusion Matrix

- Code Plot Confusion Matrix

```
import matplotlib.pyplot as plt
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=45)
    plt.yticks(tick_marks, classes)

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

Figure 4.53: cod6-fadila

Penjelasan : Untuk gambar 4.53 yang ditampilkan / diperlihatkan telah dieksekusi dimana fungsi code tersebut yaitu untuk mencetak dan memplot Confussion Matrix. Normalisasi dapat diterapkan dengan mengatur ‘normalize = True’. Ada Confussion Matrik menggunakan normalisasi dan ada confussion matrik tidak menggunakan normalisasi.

- Hasil Plot Confusion Matrix

Hasilnya dapat diliat pada gambar berikut 4.54 dan 4.55 .

7. Program Cross Validation

Figure 4.54: 6-fadila

```
In [49]: import numpy as np
      ... np.set_printoptions(precision=2)
      ... plot_confusion_matrix(cm, classes=fedila, normalize=True)
      ... plt.show()
Normalized confusion matrix
[[ 1.  0. ]
 [ 0.01  0.92]]
```

Figure 4.55: lanjutan6-fadila

```
from sklearn.model_selection import cross_val_score
scores = cross_val_score(clf, d_train_att, d_train_label, cv=5)

skorrrata2=scores.mean()
skoresd=scores.std()
```

Penjelasan : Pada codingan maupun hasil dari cross validation berikut, menjelaskan dimana variabel score akan melakukan cross validation pada variabel clf, d train att, dan train label. Variabel skorrata2 akan menghitung nilai rata-rata dari variabel scores tadi menggunakan function mean. Dan scoresd menghitung standar deviasi dari data yang diberikan.

- Hasil Program Cross Validation

Hasilnya dapat diliat pada gambar berikut 4.56 dan 4.57 .

```
In [54]: from sklearn.model_selection import cross_val_score
...: scores = cross_val_score(clf,d_train_att,d_train_label, cv=5)
...:
```

Figure 4.56: 7-fadila

skoresd float64 1 0.010910084509667118

Figure 4.57: lanjutan7-fadila

8. Program Pengamatan Komponen Informasi

• Hasil Dan Code Pengamatan Komponen Informasi :

Hasilnya dapat diliat pada gambar berikut 4.58 dan 4.59

Penjelasan :

```
In [13]: max_features_opts = range(1, 10, 1)
...: n_estimators_opts = range(2, 49, 4)
...: rf_params = np.empty((len(max_features_opts)*len(n_estimators_opts),4), float)
...:
...: for max_features in max_features_opts:
...:     for n_estimators in n_estimators_opts:
...:         for rf_params_idx in range(len(max_features_opts)*len(n_estimators_opts)):
...:             rf_params[rf_params_idx][0] = max_features
...:             rf_params[rf_params_idx][1] = n_estimators
...:             rf_params[rf_params_idx][2] = scores.mean()
...:             rf_params[rf_params_idx][3] = scores.std() * 2
...:
...:             print("Max features: %d, num estimators: %d, accuracy: %0.2F (%/- %0.2F)" % (max_features,
...: n_estimators, scores.mean(), scores.std() * 2))
```

Figure 4.58: 8-fadila

```
In [14]: import matplotlib.pyplot as plt
...: from mpl_toolkits.mplot3d import Axes3D
...: from matplotlib import cm
...: fig = plt.figure()
...: ax = fig.gca(projection='3d')
...: x = rf_params[:,0]
...: y = rf_params[:,1]
...: z = rf_params[:,2]
...: ax.set_zlim(0.9, 1)
...: ax.set_xlabel('Max features')
...: ax.set_ylabel('Num estimators')
...: ax.set_zlabel('Avg accuracy')
...: plt.show()
```

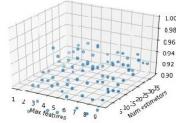


Figure 4.59: lanjutan8-fadila

- Max featuresnya dari range 1 sampai 10
- Untuk n estimators dengan range 2 sampai 40
- Kemudian pada variabel rf params berisikan function np empty dimana akan membuat array baru berisikan tipe yang didefinisikan dengan random value
- Mendefinisikan nilai i dimulai dari angka 0 dimana max features dan n estimators menggunakan klasifikasi randomforestclassifier menggunakan data prediksi.
- Mendefinisikan rfparams untuk max features , n estimators, nilai rata dan std
- Hasil (komponen informasi) juga ditampilkan dalam bentuk/module/ matplotlib
- Secara keseluruhan untuk variabel x, y dan z dilakukan penyettingan dengan parameternya masing-masing
- Label pun di definisikan untuk setiap variabelnya
- Dan hasil keseluruhan akan nampak seperti pada gambar yang mengam-barkan komponen informasi secara lengkap

4.3.3 Penanganan Error Fadila

Menangani dan Mengatasi Error Pada Praktek

1. Error 1 :

- Code Yang Error :

```
import pandas as pd
d = pd.read_csv("dataset/Youtube02-KatyPerry.csv")
```

- Peringatan Error :

```
FileNotFoundException: File b'dataset/Youtube02-KatyPerry.csv' does
```

- Gambar Error 1 : (Lebih jelas)

```
File "C:\ProgramData\Anaconda\lib\site-packages\pandas\io\parsers.py", line 707, in
    _init_
    self._make_engine(self.engine)
File "C:\ProgramData\Anaconda\lib\site-packages\pandas\io\parsers.py", line 1814, in
    _make_engine
    self._engine = CParserWrapper(self.f, **self.options)
File "C:\ProgramData\Anaconda\lib\site-packages\pandas\io\parsers.py", line 1788, in
    _read
    self._reader = parsers.TextReader(src, **kwargs)
File "pandas\_libs\parsers.pyx", line 384, in
pandas\_libs.parsers._read
File "pandas\_libs\parsers.pyx", line 695, in
pandas\_libs.parsers.TextReader._setup_parser_source
FileNotFoundException: File b'dataset/Youtube02-KatyPerry.csv' does not exist
```

Figure 4.60: error1-fadila

– Cara Penanganan :

- Pada Codingan yang dieksekusi sebenarnya untuk membaca dataset dari Youtube02-KatyPerry.csv
- Namun, terdapat error dan hal tersebut disebabkan karena file codingan yang dieksekusi tidak berada pada folder yang sama dengan dataset Katy Perry.
- Silahkan tuliskan codingan berikut untuk mengganti codingan yang bermasalah

```
import pandas as pd
d = pd.read_csv("Youtube02-KatyPerry.csv")
```

- Dengan menganti codingan tersebut, maka tidak akan terjadi error lagi.

4.3.4 Praktek

Chapter 5

Vektorisasi Kata dan Dokumen

brief of conclusion

5.1 Lusia Violita Aprilian-1164080

5.1.1 Teori

1. Jelaskan kenapa kata-kata harus di lakukan vektorisasi. dilengkapi dengan ilustrasi atau gambar.

- Jawaban :

Kata-kata harus dilakukan vektorisasi karena kita dapat mengetahui kemiripan kata satu sama lain untuk mendapatkan kinerja yang lebih baik di mesin learning.

- Ilustrasi :

Maksud dari kalimat di atas adalah apabila sebelumnya menggunakan bag of word atau kantong kata yang dimana model tersebut hanya menghitung setiap kata muncul disetiap dokumen, maka disini akan menggunakan vektorisasi kata. Misal kita memiliki 2 bag of word vektor yang bisa dikatakan mirip, lalu kita dapat membandingngkan dokumen untuk melihat kemiripannya atau membandingkan kata-kata yang digunakan di kedua dokumen tersebut. Dengan kata lain, jika kedua dokumen memiliki banyak kata yang mirip dan muncul beberapa kali, mereka akan dianggap serupa. Oleh karena itu pentingnya vektorisasi didasarkan untuk mengetahui seberapa mirip kata satu sama lain. Gambar 5.1 merupakan gambaran ilustrasi.

2. Jelaskan mengapa dimensi dari vektor dataset google bisa sampai 300, dilengkapi dengan ilustrasi atau gambar.

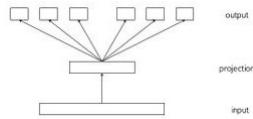


Figure 5.1: Lusia-Vektorisasi

- Jawaban :

Karena Google menggunakan model Word2Vec (model vektorisasi) dan setiap kata merupakan vektor yang bernilai 300 dimensi. Model yang digunakan google telah dilatih memeriksa jutaan atau milyaran halaman teks. Dengan begitu google dapat menyediakan informasi yang dibutuhkan berdasarkan inputan kata kunci yang divektorisasi.

- Ilustrasi :

Misalnya saja kita menggunakan kata kucing, anjing, dan spatula :

- Kucing = [0.012, 0.204, ..., -0.275, 0.056] (300 dimensi)
- Anjing = [0.051, -0.022, ..., -0.355, 0.227]
- Spatula = [-0.191, -0.043, ..., -0.348, 0.398]

Persamaan terebut menjelaskan nilai dimensi dari setiap kata. Lalu dari dimensi kata tersebut kemudian dicari kesamaannya atau jaraknya, seperti berikut :

- Kesamaan (jarak) antara kucing dan anjing = 0,761. Perhitungan pada gambar 5.2

```
In [7]: gmodel.similarity('cat', 'dog')
Out[7]: 0.76094570897822089
```

Figure 5.2: Lusia-Kesamaan Kucing Anjing

- Kesamaan antara kucing dan spatula = 0,124. Perhitungan pada gambar 5.3

```
In [8]: gmodel.similarity('cat', 'spatula')
Out[8]: 0.12412612600429632
```

Figure 5.3: Lusia-Kesamaan Kucing Spatula

Karena hasil dari kucing dan anjing lebih tinggi maka bisa dikatakan mirip. Maka dari itu, ketika kita mencari dengan kata kunci kucing di google, google kadang juga menyuguhkan gambar anjing karena dinilai sama.

3. Jelaskan konsep vektorisasi untuk kata, dilengkapi dengan ilustrasi atau gambar.

- Jawaban :

Konsep vektorisasi untuk kata adalah dengan mengetahui nilai dimensi dari setiap satuan kata. Lalu menghitung nilai kesamaan dimensi tersebut, setalah itu dibandingkan (seperti ilustrasi soal no.2). Vektorisasi kata (Word2Vec) menggunakan neural networks untuk mempelajari kata. Pada tingkatan kesulitan yang tinggi, neural networks mirip dengan random forest atau decision tree dan teknik machine learning lainnya karena mereka diberikan banyak input dan banyak output, sehingga dapat belajar bagaimana memprediksi output dari input. Untuk Word2Vec, input adalah satu kata dan hasilnya adalah kata-kata terdekat dari teks. Dengan demikian, Word2Vec belajar vektor kata dengan mengingat kata konteksnya.

- Ilustrasi :

Jadi dari pemaparan soal no.2, anjing dan kucing akan memiliki vektor kata yang sama karena kedua kata ini digunakan dengan cara yang sama. Seperti contoh pada kalimat 'dia memelihara anjing' dan 'dia memelihara kucing'. Apabila 2 kalimat tersebut menggunakan model kantong kata kontinu (salah satu teknik yang digunakan untuk Word2Vec) dengan neural networking yang mempelajari 300-dimensi perkata maka akan diperoleh hasil output yang sama. Gambar 5.4 merupakan gambar ilustrasinya.

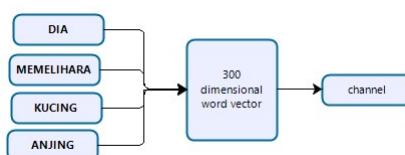


Figure 5.4: Lusia-Konsep Vektorisasi Kata

4. Jelaskan konsep vektorisasi untuk dokumen, dilengkapi dengan ilustrasi atau gambar.

- Jawaban :

Konsep vektorisasi dokumen mirip dengan konsep vektorisasi kata, hanya saja yang membedakan dalam hal ini inputnya adalah nama dokumen

(seperti nama file) dan outputnya adalah kata-kata dari dokumen. Dan bentuk atau model yang digunakan adalah Doc2Vec.

Dalam hal ini, sebagai vektor yang membantu kita memprediksi kata-kata, dari mengetahui nama file. Sebenarnya, inputnya tidak terlalu penting, yang terpenting adalah nama dari file dokumen tersebut. Kita hanya perlu melacak kata-kata yang semuanya berasal dari dokumen yang sama. Sehingga semua kata-kata akan terhubung pada nama file tersebut. Sebab itu kita dapat memprediksi kata-kata dokumen berdasarkan nama filenya, kita dapat secara efektif memiliki model yang tahu kata-kata mana yang cocok dalam sebuah dokumen.

- Ilustrasi :

Misalnya saja pada sebuah dokumen yang berisi ulasan atau komentar. Sebuah komentar tentunya memiliki 2 sisi yang berlawanan yaitu positif dan negatif. Dengan menggunakan vektorisasi dokumen, kita dapat mengetahui banyak kata positif yang berbeda digunakan dalam ulasan positif dan banyak kata negatif digunakan dalam ulasan negatif. Gambar 5.5 merupakan gambar ilustrasinya.

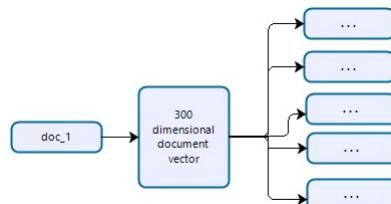


Figure 5.5: Lusia-Vektorisasi Dokumen

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

- Mean

- Penjelasan :

Mean atau nilai rata-rata adalah teknik penjelasan kelompok yang berdasarkan nilai rata-rat dari kelompok data.

- Ilustrasi :

Misal kita ingin mengetahui nilai rata-rata dari suatu kelompok, maka kita bisa menjumlahkan data seluruh individu dalam kelompok tersebut. Kemudian dibagi dengan jumlah individu yang ada pada kelompok tersebut. Gambar 5.6 merupakan gambar ilustrasinya.

$$\bar{x} = \frac{x_1 + x_2 + \dots + x_n}{n}$$

Ket:
 \bar{x} = Mean
 x_n = Nilai data
 n = Banyaknya data

Figure 5.6: Lusia-Mean

- Standar Deviasi

- Penjelasan :

Standar deviasi merupakan akar dari varians (ingat, karena pada varians kita mengkuadratkan selisih data dari rata-ratanya, maka dengan mengakarkannya, kita mendapatkan kembali nilai asalnya). Atau umumnya standar deviasi merupakan nilai statistik yang digunakan untuk menentukan bagaimana sebaran dari data sample dan seberapa dekat titik data individu ke mean atau nilai rata-rata sample.

- Ilustrasi :

Misal kita ingin mengetahui besar nilai sampel terhadap nilai rata-rata, maka kita dapat menggunakan standar deviasi. Gambar 5.7 merupakan gambar ilustrasinya.

$$SD = \sqrt{\frac{\sum x^2}{N}}$$

$$SD = \sqrt{\frac{\sum f x^2}{N}}$$

Keterangan:
 SD = Standar Deviasi
 $\sum x^2$ = Jumlah semua deviasi setelah dikuadratkan
(a) Rumus untuk frekuensi tunggal atau satu
(b) Rumus untuk frekuensi lebih dari satu

Figure 5.7: Lusia-Standar Deviasi

6. Jelaskan apa itu skip-gram, dilengkapi dengan ilustrasi atau gambar.

- Jawaban :

Skip-gram adalah teknik pada vektorisasi kata. Skrip-gram merupakan kebalikan dari teknik kantung kata kontinus. Dimana pada teknik ini, kata tengah adalah input dan kata-kata konteks adalah output. Dalam teknik ini, vektor kata tengah digunakan untuk memprediksi kata konteks yang diberikan.

- Ilustrasi :

Misalkan saja kita ingin memprediksi kata, pada teknik ini vektor kata tengah digunakan untuk memprediksi kata konteks yang diberikan kata tengah. Seperti gambar 5.8 :

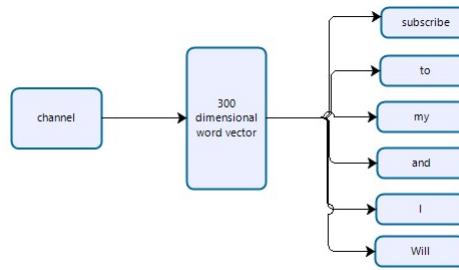


Figure 5.8: Lusia-Skip gram

5.1.2 Praktek Program

1. Cobalah dataset google, dan jelaskan vektor dari kata love, faith, fall, sick, clear, shine, bag, car, wash, motor, cycle dan cobalah untuk melakukan perbandingan similitudo dari masing-masing kata tersebut. jelaskan arti dari outputan similaritas dan setiap baris kode yang dibuat.

Jawaban :

- (a) Mencoba dataset

```

In [2]: import gensim, logging
C:\Users\lsvapr\Anaconda3\lib\site-packages\gensim\utils.py:1197:
UserWarning: detected Windows; aliasing chunkize to chunkize_serial
  warnings.warn("detected Windows; aliasing chunkize to
  chunkize_serial")
In [3]: logging.basicConfig(format='%(asctime)s : %(levelname)s : %
(message)s', level=logging.INFO)
In [4]: gmodel =
gensim.models.KeyedVectors.load_word2vec_format('GoogleNews-vectors-
negative300.bin', binary=True)
2019-03-28 17:31:59,535 : INFO : loading projection weights from
GoogleNews-vectors-negative300.bin
  
```

Figure 5.9: Lusia-Mencoba dataset

Penjelasan gambar 5.9 :

- Pada In [2] berfungsi untuk mengimport gensim untuk doc2vec dan tagging untuk opsi informasi
- Pada In [3] berfungsi untuk men-setting setiap aktifitas keluar logging nya yang berisi informasi waktu, level, dan pesan.
- Pada In [4] berfungsi untuk instalasi word2vec dari google model

- (b) Menjelaskan vektor

- Love

Pada gambar 5.10, gmodel['love'] berfungsi untuk testing model love dari google model. Dan outputnya adalah berupa array.

```

In [5]: gmodel['love']
Out[5]:
array([ 0.10302734, -0.15234375,  0.02587891,  0.16503906,
-0.16593906,
         0.066689453,  0.29296875, -0.26367188, -0.148625 ,
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,  0.22949219, -0.14355469,
0.16796875,
         -0.03515625,  0.05517578,  0.10693359,  0.11181641,
-0.16308594,
         -0.11181641,  0.13964844,  0.01556396,  0.12792969,
0.15429688,

```

Figure 5.10: Lusia-Menjelaskan Vektor Love

```

In [6]: gmodel['faith']
Out[6]:
array([ 0.26367188, -0.04150391,  0.1953125 ,  0.13476562,
-0.14648438,
         0.11962891,  0.04345703,  0.10351562,  0.12207031,
0.13476562,
         0.06640625,  0.18945312, -0.16601562,  0.21679688,
-0.27148438,
         0.3203125 ,  0.10449219,  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,

```

Figure 5.11: Lusia-Menjelaskan Vektor Faith

- Faith

Pada gambar 5.11, gmodel['faith'] berfungsi untuk testing faith dari google model. Dan outputnya adalah berupa array.

- Fall

```

In [7]: gmodel['fall']
Out[7]:
array([-0.042472461,  0.10742188, -0.09277344,  0.16894531,
-0.1328125 ,
         -0.10693359,  0.04321289,  0.01904297,  0.14648438,
0.15039606,
         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 ,

```

Figure 5.12: Lusia-Menjelaskan Vektor Fall

Pada gambar 5.12, gmodel['fall'] berfungsi untuk testing fall dari google model. Dan outputnya adalah berupa array.

- Sick

```

In [8]: gmodel['sick']
Out[8]:
array([ 1.82617188e-01,  1.49414062e-01, -4.05273438e-02,
1.64062500e-01,
         -2.59705625e-01,  3.22265625e-01,  1.73828125e-01,
-1.47468056e-01,
         1.01074219e-01,  5.46875000e-02,  1.66992188e-01,
-1.68945312e-01,
         2.24384199e-03,  9.66796875e-02, -1.66015625e-01,
-1.12304688e-01,
         1.66015625e-01,  1.79687500e-01,  5.92041016e-03,
2.45117188e-01,
         8.74023438e-02, -2.56347656e-02,  3.41796875e-01,
4.98046875e-02,

```

Figure 5.13: Lusia-Menjelaskan Vektor Sick

Pada gambar 5.13, gmodel['sick'] berfungsi untuk testing sick dari google model. Dan outputnya adalah berupa array.

```
In [9]: gmodel['clear']
Out[9]:
array([-2.44140625e-04, -1.02050781e-01, -1.49414062e-01,
-4.24804688e-02,
-1.67968750e-01, -1.46484375e-01, 1.76757812e-01,
1.46484375e-01,
2.26562500e-01, 9.76562500e-02, -2.67578125e-01,
-1.29882812e-01,
1.2404719e-01, 2.23632812e-01, -2.13867188e-01,
3.10058962e-02,
2.00195312e-01, -4.76874219e-02, -6.83593750e-02,
-1.21093750e-01,
```

Figure 5.14: Lusia-Menjelaskan Vektor Clear

- Clear

Pada gambar 5.14, gmodel['clear'] berfungsi untuk testing clear dari google model. Dan outputnya adalah berupa array.

- Shine

```
In [10]: gmodel['shine']
Out[10]:
array([-0.12402344, 0.25976562, -0.15917969, -0.27734375,
0.30273438,
0.09960938, 0.39257812, -0.22949219, -0.18359375,
0.3671875,
-0.10302734, 0.13671875, 0.25390625, 0.07128906,
0.02539062,
0.21777344, 0.24023438, 0.5234375, 0.12304688,
-0.19335938,
```

Figure 5.15: Lusia-Menjelaskan Vektor Shine

Pada gambar 5.15, gmodel['shine'] berfungsi untuk testing shine dari google model. Dan outputnya adalah berupa array.

- Bag

```
In [11]: gmodel['bag']
Out[11]:
array([-0.03515625, 0.15234375, -0.12402344, 0.13378906,
-0.11328125,
-0.0133667, -0.16113281, 0.14648438, -0.06835938, 0.140625,
-0.06005859, -0.3046875, 0.20996094, -0.04345703,
-0.2109375,
-0.05957031, -0.05053711, 0.10253906, 0.19042969,
-0.09423628,
0.18847656, -0.07958984, -0.11035156, -0.07910156,
```

Figure 5.16: Lusia-Menjelaskan Vektor Bag

Pada gambar 5.16, gmodel['bag'] berfungsi untuk testing bag dari google model. Dan outputnya adalah berupa array.

- Car

```
In [12]: gmodel['car']
Out[12]:
array([ 0.13085938, 0.00642285, 0.03344727, -0.05883789,
0.04003906,
-0.14257812, 0.04931641, -0.16894531, 0.20898438,
0.11962891,
0.18066406, -0.25, -0.10400391, -0.10742188,
-0.01879883,
0.05200195, -0.00216675, 0.06445312, 0.14453125,
-0.04541016,
0.16113281, -0.01611328, -0.03088379, 0.08447266,
```

Figure 5.17: Lusia-Menjelaskan Vektor Car

Pada gambar 5.17, gmodel['car'] berfungsi untuk testing car dari google model. Dan outputnya adalah berupa array.

```
In [13]: gmodel['wash']
Out[13]:
array([-0.46044922e-03,  1.41601562e-01, -5.46875000e-02,
       1.34765625e-01, -2.08081250e-01,  3.24218750e-01, -8.44726562e-02,
       -1.29880312e-01,  1.67910156e-01,  2.53906250e-01,  1.13525391e-02,
       -1.66992188e-01, -2.7954316e-02,  2.08007812e-01, -4.27246094e-02,
       1.05468750e-01, -7.42187500e-02,  3.04687500e-01,  2.11914062e-01,
```

Figure 5.18: Lusia-Menjelaskan Vektor Wash

- Wash

Pada gambar 5.18, gmodel['wash'] berfungsi untuk testing wash dari google model. Dan outputnya adalah berupa array.

- Motor

```
In [14]: gmodel['motor']
Out[14]:
array([ 5.73738469e-02,  1.50390625e-01, -4.61425781e-02,
       -1.32812500e-01, -2.59765625e-01, -1.77734375e-01,  3.68652344e-02,
       -4.37500000e-01,  2.34375000e-02,  2.57812500e-01,  1.74804688e-01,
       2.44140625e-02, -2.51953125e-01, -5.76171875e-02,  8.15429688e-02,
       1.86767578e-02,
```

Figure 5.19: Lusia-Menjelaskan Vektor Motor

Pada gambar 5.19, gmodel['motor'] berfungsi untuk testing motor dari google model. Dan outputnya adalah berupa array.

- Cycle

```
In [15]: gmodel['cycle']
Out[15]:
array([ 0.04541016,  0.21679688, -0.02709961,  0.12353516,
       -0.20703125,  -0.1328125,  0.26367188, -0.12890625, -0.125
       , 0.15332031, -0.18261719, -0.15820312, -0.06176758,  0.21972656,
       -0.15820312,  0.02563477, -0.07568359, -0.0625
       , 0.04614258, -0.31054688,
```

Figure 5.20: Lusia-Menjelaskan Vektor Cycle

Pada gambar 5.20, gmodel['cycle'] berfungsi untuk testing cycle dari google model. Dan outputnya adalah berupa array.

(c) Perbandingan similitudi

Dari percobaan model google, selanjutnya dibandingkan similitudinya seperti gambar 5.21 berikut :

Penjelasan gambar 5.21 'Lusia-Perbandingan' :

- gmodel.similarity('sick','love') untuk mengecek similaritas sick dengan love dan hasilnya 0,266
- gmodel.similarity('sick','faith') untuk mengecek similaritas sick dengan faith dan hasilnya 0,123

```

In [16]: gmodel.similarity('sick','love')
Out[16]: 0.2665636147352395

In [17]: gmodel.similarity('sick','faith')
Out[17]: 0.12307321986354715

In [18]: gmodel.similarity('sick','fall')
Out[18]: 0.0896575428172724

In [19]: gmodel.similarity('sick','clear')
Out[19]: 0.1590854201720585

In [20]: gmodel.similarity('sick','shine')
Out[20]: 0.04658270568098772

In [21]: gmodel.similarity('sick','bag')
Out[21]: 0.1177503527595472

In [22]: gmodel.similarity('sick','car')
Out[22]: 0.11852219525661713

In [23]: gmodel.similarity('sick','wash')
Out[23]: 0.25871449587899464

In [24]: gmodel.similarity('sick','motor')
Out[24]: 0.08594041275038308

In [25]: gmodel.similarity('sick','cycle')
Out[25]: 0.014689879077015326

```

Figure 5.21: Lusia-Perbandingan

- gmodel.similarity('sick','fall') untuk mengecek similaritas sick dengan fall dan hasilnya 0,089
- gmodel.similarity('sick','clear') untuk mengecek similaritas sick dengan clear dan hasilnya 0,159
- gmodel.similarity('sick','shine') untuk mengecek similaritas sick dengan shine dan hasilnya 0,046
- gmodel.similarity('sick','bag') untuk mengecek similaritas sick dengan bag dan hasilnya 0,117
- gmodel.similarity('sick','car') untuk mengecek similaritas sick dengan car dan hasilnya 0,118
- gmodel.similarity('sick','wash') untuk mengecek similaritas sick dengan wash dan hasilnya 0,258
- gmodel.similarity('sick','motor') untuk mengecek similaritas sick dengan motor dan hasilnya 0,085
- gmodel.similarity('sick','cycle') untuk mengecek similaritas sick dengan cycle dan hasilnya 0,014

Dari hasil percobaan tersebut diperoleh nilai tertinggi 0,266. Maka yang dianggap dalam kategori kata yang sama atau mirip dengan kata sick adalah kata love.

2. Jelaskan dengan kata dan ilustrasi fungsi dari extract words dan PermuteSentences.

- extract words

```

In [64]: import re, string
...
...: def extract_words(line):
...:     out = (x.lower() for x in line.split())
...:     return (x for x in out if len(x) > 0)
...
...: # Test the function
...: list(extract_words("This isn't really a sentence"))
Out[64]: ['this', "isn't", 'really', 'a', 'sentence']

```

Figure 5.22: Lusia-extract words

Penjelasan gambar 5.22 :

– Penjelasan 5.22 :

Gambar 'Lusia-extract words' menjelaskan fungsi extract words. Fungsi extract words berfungsi untuk mengambil suatu argumen dengan tipe data string. Atau lebih jelasnya mengekstrak kata dari string, menghapus tanda baca, dan memisahkan kata-kata kedalam daftar.

– Ilustrasi 5.22 :

Gambar 'Lusia-extract words' mengilustrasikan sebuah kalimat 'This isn't really a sentence' yang akan dipisahkan perkata. Dimana library re (regex module) dan library string di import terlebih dahulu. Lalu variable out mendefinisikan X untuk mengembalikan string pada objek line yang telah di split (dibagi atau dipisahkan). Kemudian, X dikembalikan (return) berdasarkan jumlah kata. Dimana jumlah kata harus lebih dari 0. Sehingga didapat hasil output seperti pada gambar.

• Permute Sentences

```

In [75]: import random
...
...: class PermuteSentences(object):
...:     def __init__(self, unit_cost=0):
...:
...:         self.unit_cost = unit_cost
...:
...:     def __call__(self, sentence):
...:         random.setstate()

```

Figure 5.23: Lusia-Permute Sentences

Penjelasan gambar 5.23 :

– Penjelasan 5.23 :

Permute sentences berfungsi untuk melakukan random teks atau pengocokan pada sebuah teks.

– Ilustrasi 5.23 :

Gambar 'Lusia-Permute Sentences' mengilustrasikan atribut unit cost yang akan dirandom pada sebuah class. Kemudian atribut diinisialisasi menggunakan method def init, yang dimana atribut bernilai samadengan

gan 0. Setelah itu atribut dipresentasikan oleh 'self'. Dan yang terakhir perintah random akan mengeksekusi, sehingga menghasilkan hasil seperti gambar 5.24 :

```
721690759,  
4083310136,  
3287611347,  
243048837,  
1918243295,  
4144150636,  
1834516913,  
949502382,  
624),  
None)
```

Figure 5.24: Lusia-Hasil Permute Sentences

3. Jelaskan fungsi dari librari gensim TaggedDocument dan Doc2Vec disertai praktek pemakaiannya. Tunjukkan keluarannya dari komputer sendiri dan artikan maksud setiap luaran yang didapatkan.

```
In [26]: from gensim.models.doc2vec import  
TaggedDocument  
...: from gensim.models import Doc2Vec
```

Figure 5.25: Lusia-TaggedDocument dan Doc2Vec

- (a) Penjelasan fungsi :

Doc2Vec telah disediakan oleh library gensim dan library tersebut memiliki sebuah class yang disebut TaggedDocument. Jadi fungsi TaggedDocument berfungsi untuk mempresentasikan kata pada dokumen dan Doc2Vec adalah model yang digunakan.

- (b) Praktek pemakaian :

Perhatikan gambar 5.25. Gambar tersebut merupakan output dari praktek pemakaian TaggedDocument dan Doc2Vec.

- Baris 1 = mengimport TaggedDocument dari library gensim
- Baris 2 = mengimport Doc2Vec dari library gensim

4. Jelaskan dengan kata dan praktek cara menambahkan data training dari file yang dimasukkan kedalam variabel.

Perhatikan gambar 5.26.

Berikut adalah praktek cara menambahkan data training dari file yang dimasukkan kedalam variabel beserta penjelasan katanya, berdasarkan gambar 5.26 :

```

In [33]: import os
...: unsup_sentences = []
...:
...: for dirname in ["/train/pos", "/train/neg", "/train/unsup", "/test/pos", "/test/neg"]:
...:     for fname in sorted(os.listdir("aclImdb" + dirname)):
...:         if fname[-4:] == '.txt':
...:             with open("aclImdb/" + dirname + "/" + fname, encoding='UTF-8') as f:
...:                 sent = f.read()
...:                 words = extract_words(sent)
...:                 unsup_sentences.append(TaggedDocument(words, [dirname + "/" + fname]))
...:
...: for dirname in ["txt_sentoken/pos", "txt_sentoken/neg"]:
...:     for fname in sorted(os.listdir(dirname)):
...:         if fname[-4:] == '.txt':
...:             with open(dirname + "/" + fname, encoding='UTF-8') as f:
...:                 for i, sent in enumerate(f):
...:                     words = extract_words(sent)
...:                     unsup_sentences.append(TaggedDocument(words, ["%s/%s-%d" % (dirname, fname, i)]))
...:
...: with open("stanfordSentimentTreebank/original_rt_snippets.txt", encoding='UTF-8') as f:
...:     for i, line in enumerate(f):
...:         words = extract_words(line)
...:         unsup_sentences.append(TaggedDocument(words, ["rt-%d" % i]))

```

Figure 5.26: Lusia-menambahkan data training

- (a) Import library os
 - (b) membuat variable unsup sentences
 - (c) Berikutnya memamnggil direktori yang akan digunakan.
 - (d) Lalu fname memberikan nama pada direktori yang telah dipanggil.
 - (e) Direktori yang telah dinamai kemudian memanggil data yang bertipe data .txt.
 - (f) Lalu data tersebut dibuka dan di urutkan sesuai dengan list dir dengan parameter dirnamanya.
 - (g) Setiap data training direalisasikan pada sebuah inputan variabel words dengan extract word.
 - (h) Yang selanjutnya dihubungkan dengan unsup sentences yang mengeksekusi class tagged document.
 - (i) Dan data selesai ditambahkan.
5. Jelaskan dengan kata dan praktek kenapa harus dilakukan pengocokan dan pembersihan data. Tunjukkan keluarannya dari komputer sendiri dan artikan maksud setiap luaran yang didapatkan.

Data harus dilakukan pengocokan supaya hasil keluaran acak tidak berurutan. Begitu juga dilakukan pembersihan data bertujuan untuk pengecekan serta pemulihan data. Perhatikan gambar 5.27.

Gambar 5.27 merupakan gambar praktik dari pengocokan dan pembersihan data. Pembersihan data dimulai dari import library re. Baris kedua penggunaan extract word dengan object sent. Baris ketiga variable sent menggunakan

```

In [41]: import re
...: def extract_words(sent):
...:     sent = sent.lower()
...:     sent = re.sub('<[^>]+>', ' ', sent) #hapus tag html
...:     sent = re.sub('(\w)\'(\w)', ' ', sent) #hapus petik satu
...:     sent = re.sub('\'\W', ' ', sent) #hapus tanda baca
...:     sent = re.sub('^\s+', ' ', sent) #hapus spasi yang berurutan
...:     return sent.split()
...:
...:
...: import random
...: class PermuteSentences(object):
...:     def __init__(self, sents):
...:         self.sents=sents
...:
...:     def __iter__(self):
...:         shuffled = list(self.sents)
...:         random.shuffle(shuffled)
...:         for sent in shuffled:
...:             yield sent

```

Figure 5.27: Lusia-Pengocokan dan Pembersihan Data

method lower untuk mengembalikan salinan string. Baris keempat untuk menghapus tag html, baris kelima untuk menghapus tanda baca petik satu, baris ke enam untuk menghapus tanda baca, baris ke tujuh untuk menghapus spasi yang berurutan, Dan yang terakhir untuk mengembalikan variable sent yang telah dibagi.

Dan untuk pengocokan dimulai dari import library random. Lalu membuat class permute sentences yang berisi object yang menfinisikan metode pengocokan.

6. Jelaskan dengan kata dan praktik kenapa model harus di save dan kenapa temporari training harus dihapus. Tunjukkan keluarannya dari komputer sendiri dan artikan maksud setiap luaran yang didapatkan.

- Penjelasan :

- Model harus di save karena berfungsi menyimpan data inferensi untuk mengikat vektor dokumen baru.
- Temporari training harus dihapus karena untuk mengosongkan sebagian memory yang telah digunakan saat melakukan training.

- Praktek :

- Save

```

In [51]: model.save('sudahiperihini.d2v')
2019-03-29 21:32:09,001 : INFO : saving Doc2Vec object under sudahiperihini.d2v, separately None
2019-03-29 21:32:12,778 : INFO : saved sudahiperihini.d2v

```

Figure 5.28: Lusia-Save

Penjelasan Gambar 5.28 :

model.save berfungsi untuk menyimpan model yang telah dibuat. Dan dua baris dibawahnya memberikan informasi mengenai penyimpanan model tersebut.

- Delete Temporary

```
In [50]: model.delete_temporary_training_data(keep_inference=True)
```

Figure 5.29: Lusia-Delete Temporary

Penjelasan gambar 5.29 :

model.delete temporary training data berfungsi untuk menghapus beberapa data.

7. jalankan dengan kata dan praktek maksud dari infer code. Tunjukkan keluarannya dari komputer sendiri dan artikan maksud setiap luaran yang didapatkan. Infer code berfungsi untuk menyimpulkan vektor yang berkaitan dengan vektor dokumen baru.

Perhatikan gambar 5.30

```
In [52]: model.infer_vector(extract_words("Aku sayang emak bapak"))
Out[52]:
array([ 0.02258755,  0.04762995, -0.00579314,  0.03751514,  0.04823327,
       0.07713442,  0.07269505, -0.17505808,  0.05247376, -0.04423291,
       0.04624939, -0.05419887,  0.19810857,  0.10814948,  0.09219473,
      -0.02153289,  0.12561473, -0.24373437,  0.06651334, -0.02546891,
      0.07725087,  0.05040888, -0.1479178,  0.02037754, -0.1078615,
      0.09396095, -0.10534833, -0.07729118, -0.05097554, -0.03977873,
      -0.05071803,  0.1458353, -0.04884738, -0.06973811, -0.11166104,
      0.02934828, -0.04872385, -0.04674706,  0.12112793,  0.11732782,
      -0.18216516,  0.00590295,  0.02064698, -0.04830755,  0.10375717,
      -0.00994448,  0.04728816, -0.054299,  0.0393485,  0.03294549,
      -0.08531133, -0.02412128], dtype=float32)
```

Figure 5.30: Lusia-Infer code

Gambar tersebut merupakan praktek dari infer code. Dimana infer vektor menyimpulkan vektor dari kata 'aku sayang emak bapak' yang sudah di ekstrak. Sehingga menghasilkan output berupa array dengan tipe data float32.

8. Jelaskan dengan praktek dan kata maksud dari cosine similarity. Tunjukkan keluarannya dari komputer sendiri dan artikan maksud setiap luaran yang didapatkan.

Cosine similarity digunakan untuk mengecek kemiripan dari 2 kalimat berbeda. Perhatikan gambar 5.31 untuk praktiknya.

```
In [82]: from sklearn.metrics.pairwise import cosine_similarity
...: cosine_similarity(
...:     [model.infer_vector(extract_words("Aku sayang emak bapak"))],
...:     [model.infer_vector(extract_words("Aduh aku ngantuk, bagaimana ini"))])
Out[82]: array([[0.8350432]], dtype=float32)
```

Figure 5.31: Lusia-cosine similarity

Mula-mula import cosine similarity dari library sklearn. Lalu cek kemiripan menggunakan cosine similarity yang didalamnya sudah terdapat 2 kalimat yang akan dibandingkan. Praktek ini menggunakan kalimat 'Aku sayang emak bapak' dan kalimat 'Aduh aku ngantuk, bagaimana ini' yang menghasilkan output berupa array dengan tipe data float32.

9. Jelaskan dengan praktek score dari cross validation masing-masing metode. Tunjukkan keluarannya dari komputer sendiri dan artikan maksud setiap luaran yang didapatkan.

Score dari cross validation berfungsi untuk memprediksi model yang telah dibuat dan memperkirakan keakuratannya. Perhatikan gambar 5.32

```
In [62]: scores = cross_val_score(clf, sentvecs, sentiments, cv=5)
...: np.mean(scores), np.std(scores)
Out[62]: (0.757, 0.007845734863959857)
```

Figure 5.32: Lusia-score

Gambar 5.32 merupakan praktik dari score. Gambar tersebut menjelaskan variable score yang akan menghitung keakurasi menggunakan teknik cross validation. Baris kedua merupakan perintah untuk menghitung variable score. Sehingga menghasilkan output 0.757 (akurasi = 76 persen)

5.1.3 Penanganan Error

1. Skrinsut Error

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

Figure 5.33: Lusia-Error

2. Kode error dan jenis error

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

3. Solusi Pemecahan

Solusi dari permasalahan error pada skrinsut error adalah dengan mengintal modul gensim di anaconda dengan cara berikut :

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

Figure 5.34: Lusia-Solusi

5.2 Rahmi Roza-1164085

5.2.1 Teori

Praktek Tugas Harian

1. Mengapa Kata-Kata Harus di Vektorisasi

Kata harus di vektorisasi dikarenakan mesin hanya mampu membaca data dalam bentuk angka. Oleh karena itu diperlukan vektorisasi kata agar mesin mampu membaca data yang telah di vektorisasi.

- Gambar :

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

Word Vector Representations

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

	<i>I</i>	<i>like</i>	<i>enjoy</i>	<i>deep</i>	<i>learning</i>	<i>NLP</i>	<i>flying</i>	.
1. I enjoy flying.	<i>I</i>	0 2	1 0	0 0	0 0	0 0	0 0	0 0
	<i>like</i>	2 0	0 1	0 0	0 0	1 0	0 0	0 0
	<i>enjoy</i>	1 0	0 0	0 0	0 0	0 1	0 0	0 0
2. I like NLP.	<i>deep</i>	0 1	0 0	0 0	1 0	0 0	0 0	0 0
	<i>learning</i>	0 0	0 1	0 0	0 0	0 0	0 0	1 0
3. I like deep learning.	<i>NLP</i>	0 1	0 0	0 0	0 0	0 0	0 0	1 0
	<i>flying</i>	0 0	1 0	0 0	0 0	0 0	0 0	1 0
	.	0 0	0 0	0 1	1 1	1 1	0 0	0 0

Figure 5.35: Vektorisasi Kata Roza

2. Mengapa Dimensi Dari Vektor Dataset Google Bisa Sampai 300

Masing-masing nilai dalam vektor 300 dimensi yang terkait dalam sebua kata "dioptimalkan" dalam beberapa hal untuk menangkap aspek yang berbeda dari

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

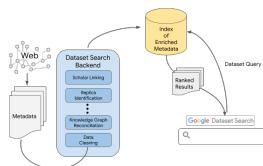


Figure 5.36: Google Dataset Rozza

3. Konsep Vektorisasi Kata

Konsep vektorisasi data merupakan kata-kata yang di inputkan pada mesin learning. Dan outputnya berupa kara-kata atau keyword dari pencarian yang tekah di lakukan sebelumnya. Contoh nya pada saat kita melakukan pencarian di channel youtube kita. Maka akan muncul hasil dari pencarian dari kata-kata yang telah kita cari.

- Gambar :

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

4. Konsep Vektorisasi Dokumen

Konsep vektorisasi dokumen yaitu mesin akan membaca terlebih dahulu semua kalimat yang ada di dalam dokumen da nanti kalimat yang ada di dalam diku-men tersebut akan di oecah menjadi kata-kata.

Word Vector Representations

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

	<i>I</i>	<i>like</i>	<i>enjoy</i>	<i>deep</i>	<i>learning</i>	<i>NLP</i>	<i>flying</i>	<i>.</i>
1. I enjoy flying.	0	2	1	0	0	1	0	0
	<i>like</i>	2	0	1	0	1	0	0
	<i>enjoy</i>	1	0	0	0	0	1	0
2. I like NLP.	<i>deep</i>	0	1	0	1	0	0	0
	<i>learning</i>	0	0	0	1	0	0	1
	<i>NLP</i>	0	1	0	0	0	0	1
3. I like deep learning.	<i>flying</i>	0	0	1	0	0	0	1
	<i>.</i>	0	0	0	1	1	1	0

Figure 5.37: Vektorisasi Kata Roza

- Gambar :

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

Documents	Vector-space representation							
1. I like deep learning.	0	0	0	0	1	0	0	0
2. I like NLP.	0	0	0	1	0	0	0	1
3. I like deep learning.	0	1	0	0	1	0	0	1
4. I like flying.	0	0	1	0	0	0	1	0
5. I like NLP.	0	1	0	0	0	0	1	0
6. I like deep learning.	0	0	1	0	1	0	0	1
7. I like NLP.	0	1	0	0	0	0	1	0
8. I like flying.	0	0	1	0	0	1	0	0
9. I like NLP.	0	1	0	0	0	0	1	0
10. I like deep learning.	0	0	1	0	1	0	0	1
11. I like NLP.	0	1	0	0	0	0	1	0
12. I like flying.	0	0	1	0	0	1	0	0
13. I like NLP.	0	1	0	0	0	0	1	0
14. I like deep learning.	0	0	1	0	1	0	0	1
15. I like NLP.	0	1	0	0	0	0	1	0
16. I like flying.	0	0	1	0	0	1	0	0
17. I like NLP.	0	1	0	0	0	0	1	0
18. I like deep learning.	0	0	1	0	1	0	0	1
19. I like NLP.	0	1	0	0	0	0	1	0
20. I like flying.	0	0	1	0	0	1	0	0
21. I like NLP.	0	1	0	0	0	0	1	0
22. I like deep learning.	0	0	1	0	1	0	0	1
23. I like NLP.	0	1	0	0	0	0	1	0
24. I like flying.	0	0	1	0	0	1	0	0
25. I like NLP.	0	1	0	0	0	0	1	0
26. I like deep learning.	0	0	1	0	1	0	0	1
27. I like NLP.	0	1	0	0	0	0	1	0
28. I like flying.	0	0	1	0	0	1	0	0
29. I like NLP.	0	1	0	0	0	0	1	0
30. I like deep learning.	0	0	1	0	1	0	0	1
31. I like NLP.	0	1	0	0	0	0	1	0
32. I like flying.	0	0	1	0	0	1	0	0
33. I like NLP.	0	1	0	0	0	0	1	0
34. I like deep learning.	0	0	1	0	1	0	0	1
35. I like NLP.	0	1	0	0	0	0	1	0
36. I like flying.	0	0	1	0	0	1	0	0
37. I like NLP.	0	1	0	0	0	0	1	0
38. I like deep learning.	0	0	1	0	1	0	0	1
39. I like NLP.	0	1	0	0	0	0	1	0
40. I like flying.	0	0	1	0	0	1	0	0
41. I like NLP.	0	1	0	0	0	0	1	0
42. I like deep learning.	0	0	1	0	1	0	0	1
43. I like NLP.	0	1	0	0	0	0	1	0
44. I like flying.	0	0	1	0	0	1	0	0
45. I like NLP.	0	1	0	0	0	0	1	0
46. I like deep learning.	0	0	1	0	1	0	0	1
47. I like NLP.	0	1	0	0	0	0	1	0
48. I like flying.	0	0	1	0	0	1	0	0
49. I like NLP.	0	1	0	0	0	0	1	0
50. I like deep learning.	0	0	1	0	1	0	0	1
51. I like NLP.	0	1	0	0	0	0	1	0
52. I like flying.	0	0	1	0	0	1	0	0
53. I like NLP.	0	1	0	0	0	0	1	0
54. I like deep learning.	0	0	1	0	1	0	0	1
55. I like NLP.	0	1	0	0	0	0	1	0
56. I like flying.	0	0	1	0	0	1	0	0
57. I like NLP.	0	1	0	0	0	0	1	0
58. I like deep learning.	0	0	1	0	1	0	0	1
59. I like NLP.	0	1	0	0	0	0	1	0
60. I like flying.	0	0	1	0	0	1	0	0
61. I like NLP.	0	1	0	0	0	0	1	0
62. I like deep learning.	0	0	1	0	1	0	0	1
63. I like NLP.	0	1	0	0	0	0	1	0
64. I like flying.	0	0	1	0	0	1	0	0
65. I like NLP.	0	1	0	0	0	0	1	0
66. I like deep learning.	0	0	1	0	1	0	0	1
67. I like NLP.	0	1	0	0	0	0	1	0
68. I like flying.	0	0	1	0	0	1	0	0
69. I like NLP.	0	1	0	0	0	0	1	0
70. I like deep learning.	0	0	1	0	1	0	0	1
71. I like NLP.	0	1	0	0	0	0	1	0
72. I like flying.	0	0	1	0	0	1	0	0
73. I like NLP.	0	1	0	0	0	0	1	0
74. I like deep learning.	0	0	1	0	1	0	0	1
75. I like NLP.	0	1	0	0	0	0	1	0
76. I like flying.	0	0	1	0	0	1	0	0
77. I like NLP.	0	1	0	0	0	0	1	0
78. I like deep learning.	0	0	1	0	1	0	0	1
79. I like NLP.	0	1	0	0	0	0	1	0
80. I like flying.	0	0	1	0	0	1	0	0
81. I like NLP.	0	1	0	0	0	0	1	0
82. I like deep learning.	0	0	1	0	1	0	0	1
83. I like NLP.	0	1	0	0	0	0	1	0
84. I like flying.	0	0	1	0	0	1	0	0
85. I like NLP.	0	1	0	0	0	0	1	0
86. I like deep learning.	0	0	1	0	1	0	0	1
87. I like NLP.	0	1	0	0	0	0	1	0
88. I like flying.	0	0	1	0	0	1	0	0
89. I like NLP.	0	1	0	0	0	0	1	0
90. I like deep learning.	0	0	1	0	1	0	0	1
91. I like NLP.	0	1	0	0	0	0	1	0
92. I like flying.	0	0	1	0	0	1	0	0
93. I like NLP.	0	1	0	0	0	0	1	0
94. I like deep learning.	0	0	1	0	1	0	0	1
95. I like NLP.	0	1	0	0	0	0	1	0
96. I like flying.	0	0	1	0	0	1	0	0
97. I like NLP.	0	1	0	0	0	0	1	0
98. I like deep learning.	0	0	1	0	1	0	0	1
99. I like NLP.	0	1	0	0	0	0	1	0
100. I like flying.	0	0	1	0	0	1	0	0
101. I like NLP.	0	1	0	0	0	0	1	0
102. I like deep learning.	0	0	1	0	1	0	0	1
103. I like NLP.	0	1	0	0	0	0	1	0
104. I like flying.	0	0	1	0	0	1	0	0
105. I like NLP.	0	1	0	0	0	0	1	0
106. I like deep learning.	0	0	1	0	1	0	0	1
107. I like NLP.	0	1	0	0	0	0	1	0
108. I like flying.	0	0	1	0	0	1	0	0
109. I like NLP.	0	1	0	0	0	0	1	0
110. I like deep learning.	0	0	1	0	1	0	0	1
111. I like NLP.	0	1	0	0	0	0	1	0
112. I like flying.	0	0	1	0	0	1	0	0
113. I like NLP.	0	1	0	0	0	0	1	0
114. I like deep learning.	0	0	1	0	1	0	0	1
115. I like NLP.	0	1	0	0	0	0	1	0
116. I like flying.	0	0	1	0	0	1	0	0
117. I like NLP.	0	1	0	0	0	0	1	0
118. I like deep learning.	0	0	1	0	1	0	0	1
119. I like NLP.	0	1	0	0	0	0	1	0
120. I like flying.	0	0	1	0	0	1	0	0
121. I like NLP.	0	1	0	0	0	0	1	0
122. I like deep learning.	0	0	1	0	1	0	0	1
123. I like NLP.	0	1	0	0	0	0	1	0
124. I like flying.	0	0	1	0	0	1	0	0
125. I like NLP.	0	1	0	0	0	0	1	0
126. I like deep learning.	0	0	1	0	1	0	0	1
127. I like NLP.	0	1	0	0	0	0	1	0
128. I like flying.	0	0	1	0	0	1	0	0
129. I like NLP.	0	1	0	0	0	0	1	0
130. I like deep learning.	0	0	1	0	1	0	0	1
131. I like NLP.	0	1	0	0	0	0	1	0
132. I like flying.	0	0	1	0	0	1	0	0
133. I like NLP.	0	1	0	0	0	0	1	0
134. I like deep learning.	0	0	1	0	1	0	0	1
135. I like NLP.	0	1	0	0	0	0	1	0
136. I like flying.	0	0	1	0	0	1	0	0
137. I like NLP.	0	1	0	0	0	0	1	0
138. I like deep learning.	0	0	1	0	1	0	0	1
139. I like NLP.	0	1	0	0	0	0	1	0
140. I like flying.	0	0	1	0	0	1	0	0
141. I like NLP.	0	1	0	0	0	0	1	0
142. I like deep learning.	0	0	1	0	1	0	0	1
143. I like NLP.	0	1	0	0	0	0	1	0
144. I like flying.	0	0	1	0	0	1	0	0
145. I like NLP.	0	1	0	0	0	0	1	0
146. I like deep learning.	0	0	1	0	1	0	0	1
147. I like NLP.	0	1	0	0	0	0	1	0
148. I like flying.	0	0	1	0	0	1	0	0
149. I like NLP.	0	1	0	0	0	0	1	0
150. I like deep learning.	0	0	1	0	1	0	0	1
151. I like NLP.	0	1	0	0	0	0	1	0
152. I like flying.	0							

$$\begin{aligned}
 \text{Mean} &= \frac{\text{Sum of all data values}}{\text{Number of data values}} \\
 &= \frac{15+13+18+16+14+17+12}{7} \\
 &= \frac{105}{7} \\
 &= 15
 \end{aligned}$$

Figure 5.39: Mean Roza



Figure 5.40: Standar Deviasi Roza

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 sehingga skip-gram cenderung lebih baik ketika ukuran corpus sangat besar.

- Gambar :

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

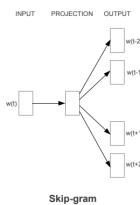


Figure 5.41: Skip-Gram Roza

- Plagiarisme Roza



Figure 5.42: Plagiarisme Roza

5.2.2 Praktek

1. Contoh Dataset Google dan Menjelaskan Vektor Dari Kata-Kata Yang Telah Ditentukan

- Love

```
Out[8]:
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,  0.22949219, -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, -0.20996094,  0.07080078,  0.02600098, -0.10644531,
       -0.10253906,  0.12304688,  0.04711914,  0.02209473,  0.05834961,
       -0.10986328,  0.14941406, -0.10693359,  0.01556396,  0.08984375,
       0.11230469, -0.04370117, -0.11376953, -0.0037384, -0.01818848,
       0.24316406,  0.08447266, -0.07080078,  0.18066406,  0.03515625,
       -0.09667969, -0.21972656, -0.00328064, -0.03198242,  0.18457031,
       0.28515625, -0.0859375, -0.11181641,  0.0213623, -0.30664062,
       -0.09228516, -0.18945312,  0.01513672,  0.18554688,  0.34375,
       -0.31054688,  0.22558594,  0.08740234, -0.2265625, -0.29492188,
       0.08251953, -0.38476562,  0.25390625,  0.26953125,  0.06298828,
       -0.00958252,  0.23632812, -0.17871094, -0.12451172, -0.17285156,
       -0.11767578,  0.19726562, -0.03466797, -0.10400391, -0.1640625,
       -0.19726562,  0.19824219,  0.09521484,  0.00561523,  0.12597656,
       0.00073624, -0.0402832, -0.03063965,  0.01623555, -0.1640625,
       -0.22167969,  0.171875,  0.12811719, -0.01965332,  0.4453125,
       0.06494141,  0.05932617, -0.1640625, -0.01367188,  0.18945312,
```

Figure 5.43: Love Roza

Penjelasan Hasil: Dapat dilihat pada hasil gambar, bahwa nilai vektor pada baris pertamanya adalah 0.10302734

- Faith

```
In [10]: gmodel['faith']
Out[10]:
array([ 0.26367188, -0.04150391,  0.1953125,  0.13476562, -0.14648438,
       0.11962891,  0.04345703,  0.10351562,  0.12207031,  0.13476562,
       0.06640625,  0.18945312, -0.16601562,  0.21679688, -0.27148438,
       0.3203125,  0.10449219,  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, -0.01708984, -0.1640625,
       -0.02819824,  0.01257324, -0.09521484, -0.18866406, -0.140625,
       -0.02258301,  0.16308594, -0.13183594, -0.08007812,  0.13085938,
       0.27539062, -0.20605469,  0.10351562, -0.20214844, -0.1875,
       0.16992188,  0.13574219,  0.13769531,  0.16308594, -0.03881836,
       -0.11132812, -0.05688477,  0.12255859,  0.09814453, -0.04956055,
       -0.02331543, -0.04248847, -0.08203125,  0.16015625,  0.04150391,
       -0.16601562,  0.13671875,  0.09619141,  0.32617188,  0.08251953,
       -0.20800781,  0.04199219,  0.05834961, -0.27734375,  0.09130859,
       -0.17382812, -0.22460938,  0.03466797,  0.19824219, -0.08837891,
       0.18359375,  0.07324219,  0.1171875, -0.33984375,  0.16796875,
       -0.13574219, -0.30078125, -0.00469971,  0.06005859, -0.29296875,
       0.15234375,  0.02966309,  0.33203125,  0.28320312,  0.09375,
       -0.20605469, -0.09082031,  0.0534668,  0.05834961, -0.03222656,
       -0.29296875,  0.25585938,  0.00430298,  0.140625,  0.05810547,
       0.21582031,  0.0291748,  0.02929688,  0.20019531,  0.34960938,
       0.10449219, -0.01940918,  0.04077148,  0.32226562, -0.1953125,
```

Figure 5.44: Faith Roza

Penjelasan Hasil: Pada hasil gambar faith dapat dilihat bahwa nilai pada vektor baris pertamanya adalah 0.26367188. Jika dibandingkan dengan gambar love dapat dikatakan bahwa kedua gambar tersebut dapat dimasukkan pada kategori yang sama.

- Fall

```
In [11]: gmodel['fall']
Out[11]:
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, -0.0098877 , -0.07763672, -0.15527344, -0.03393555,
       0.04199219, -0.29882812, -0.18554688,  0.08496694, -0.02087402,
       0.13574219, -0.22558594,  0.33789062, -0.03564453, -0.10839844,
       -0.19335938,  0.0546875 , -0.049556055,  0.3671875 , -0.03295898,
       0.10205078, -0.15136719, -0.00445557,  0.04003906,  0.27539062,
       -0.06933594,  0.05834961,  0.01422119, -0.01397705, -0.05395508,
       -0.0255127 ,  0.06298828,  0.07080078, -0.07617188,  0.06542969,
       -0.01672363, -0.04711914,  0.19628906, -0.08984375,  0.078125 ,
       0.2109375 ,  0.0612793 ,  0.08789062,  0.19628906,  0.11376953,
       0.06542969,  0.03125 ,  0.12988281,  0.02270508,  0.14550781,
       -0.06225586, -0.37695312, -0.05737305, -0.06396484,  0.08984375,
       0.00448608, -0.14160156, -0.04541016, -0.0703125 ,  0.06005859,
       0.26757812,  0.02001953, -0.12695312, -0.04882812,  0.18945312,
       -0.03466797,  0.04638672,  0.1484375 ,  0.01708984, -0.08789062,
       -0.14941406, -0.02331543, -0.03955078, -0.10400391, -0.14160156,
       -0.18261719, -0.03076172,  0.04589844, -0.2890625 , -0.03540039,
       0.12890625, -0.10595703,  0.17578125,  0.06689453,  0.34960938,
       0.04296875,  0.09863281, -0.08056641, -0.06298828,  0.12255859,
       0.0234375 , -0.06494141,  0.09667969, -0.04589844,  0.04956055,
       0.08007812, -0.00482178, -0.1640625 , -0.03271484,  0.0703125 ,
```

Figure 5.45: Fall Roza

Penjelasan Hasil: Pada hasil gambar fall dapat dilihat bahwa nilai pada vektor baris pertamanya adalah -0.04. Jika dibandingkan dengan gambar faith dapat dikatakan bahwa kedua gambar tersebut tidak dapat dimasukkan pada kategori yang sama.

- Sick

```
In [12]: gmodel['sick']
Out[12]:
array([ 1.82617188e-01,  1.49414062e-01, -4.05273438e-02,  1.64062500e-01,
       -2.59765625e-01,  3.22265625e-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,  5.92041016e-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, -3.12500000e-02,  1.98974609e-02, -6.39648438e-02,
       1.18652344e-01,  1.23046875e-01, -6.03027344e-02,  4.68750000e-01,
       9.13085938e-02, -3.12500000e-01,  1.84570312e-01, -1.51367188e-01,
       5.78613281e-02, -1.04980469e-01, -1.68945312e-01, -8.00781250e-02,
       -2.01171875e-01,  1.06281172e-02, -1.29882812e-01, -1.25976562e-01,
       -5.56640625e-02,  3.14453125e-01,  5.61523438e-02, -1.20117188e-01,
       7.12890625e-02,  4.37011719e-02,  2.05078125e-01,  5.71289062e-02,
       8.44726562e-02,  2.15820312e-01, -1.26953125e-01,  8.78906250e-02,
       2.48046875e-01, -6.54296875e-02, -0.202636719e-02,  1.52343750e-01,
       -3.57421875e-01,  3.02124023e-03, -2.0007812e-01, -5.05371094e-02,
       2.81982422e-02,  1.73828125e-01, -2.08007812e-01, -5.93261719e-02,
       6.49414062e-02,  3.63769531e-02,  1.91406250e-01,  2.77343750e-01,
       3.54008590e-02,  1.56250000e-01, -2.03857422e-02,  2.26562500e-01,
       -4.66308594e-02, -5.17578125e-02, -1.63085938e-01,  4.17480469e-02,
       2.01171875e-01, -2.01171875e-01, -1.50756836e-02,  2.61718750e-01,
       -1.10839844e-01, -4.21875000e-01,  2.22167969e-02,  1.46484375e-01,
       4.19921875e-01, -6.88476562e-02,  9.42382812e-02, -1.96289062e-01,
       -9.42382812e-02, -3.12500000e-02,  6.34765625e-02,  2.47802734e-02,
```

Figure 5.46: Sick Roza

Penjelasan Hasil: Pada hasil gambar sick dapat dilihat bahwa nilai pada vektor baris pertamanya adalah 1.82. Jika dibandingkan dengan gambar fall dapat dikatakan bahwa kedua gambar tersebut tidak dapat dimasukkan pada kategori yang sama.

- Clear

```
In [13]: gmodel['clear']
Out[13]:
array([-2.44140625e-04, -1.02050781e-01, -1.49414062e-01, -4.24804688e-02,
-1.67968750e-01, -1.46484375e-01, 1.76757812e-01, 1.46484375e-01,
2.26562500e-01, 9.76562500e-02, -2.67578125e-01, -1.29882812e-01,
1.24511719e-01, 2.23632812e-01, -2.13867188e-01, 3.10058594e-02,
2.00195312e-01, -4.76074219e-02, -6.83593750e-02, -1.21093750e-01,
3.22265625e-02, 3.14453125e-01, -1.11816406e-01, 8.00781250e-02,
-2.75878906e-02, -6.04248047e-03, -7.37304688e-02, -1.72851562e-01,
9.66796875e-02, -4.91333008e-03, -1.78710938e-01, -1.40380859e-03,
7.91015625e-02, 1.07910156e-01, -1.10351562e-01, -8.34960938e-02,
1.98974609e-02, -3.14331055e-03, 1.30615234e-02, 3.34472656e-02,
2.55859375e-01, -7.12890625e-02, 2.83203125e-01, -2.75390625e-01,
-8.49609375e-02, -3.73535156e-02, -9.17968750e-02, -1.30859375e-01,
3.49121094e-02, 7.32421875e-02, 2.17773438e-01, 5.00488281e-02,
7.47070312e-02, -1.25000000e-01, -5.73730469e-02, -2.74658203e-02,
-1.37329102e-02, -2.13867188e-01, -1.12792969e-01, -4.98046875e-02,
-1.92382812e-01, 1.32812500e-01, -5.00488281e-02, -1.66015625e-01,
-1.57470703e-02, -1.37695312e-01, 3.88183594e-02, 1.57226562e-01,
-1.52343750e-01, -1.64794922e-02, -2.27539062e-01, 3.34472656e-02,
1.55273438e-01, 1.65039062e-01, -1.66992188e-01, 3.14941406e-02,
2.85156250e-01, 2.69531250e-01, 3.32031250e-02, 2.41210938e-01,
1.39160156e-02, 5.12695312e-02, 9.76562500e-02, 3.14941406e-02,
2.51464844e-02, -2.85156250e-01, -1.24023438e-01, -6.88476562e-02,
-5.29785156e-02, 2.06654688e-01, -2.07031250e-01, -1.60156250e-01,
-2.61230469e-02, -3.01513672e-02, 8.66699219e-03, -1.30859375e-01,
3.88183594e-02, 8.60595703e-03, 5.31005859e-03, -6.05468750e-02,
1.03759766e-02, 1.33789062e-01, -1.89971924e-03, -4.27246094e-02,
```

Figure 5.47: Clear Roza

Penjelasan Hasil: Pada hasil gambar clear dapat dilihat bahwa nilai pada vektor baris pertamanya adalah -2.44. Jika dibandingkan dengan gambar sick dapat dikatakan bahwa kedua gambar tersebut tidak dapat dimasukkan pada kategori yang sama.

- Shine

```
In [14]: gmodel['shine']
Out[14]:
array([-0.12402344, 0.25976562, -0.15917969, -0.27734375, 0.30273438,
0.09960938, 0.39257812, -0.22949219, -0.18359375, 0.3671875 ,
-0.10302734, 0.13671875, 0.25390625, 0.07128906, 0.02539062,
0.21777344, 0.24023438, 0.5234375 , 0.12304688, -0.19335938,
-0.05883789, 0.0612793 , -0.01940918, 0.07617188, 0.05102539,
0.20019531, 0.38085938, 0.01625006, -0.05029297, 0.14648438,
-0.34765625, 0.02563477, -0.23925781, -0.04516602, -0.00479126,
-0.24121094, -0.18945312, -0.15234375, -0.05493164, 0.01434326,
0.390625 , -0.2109375 , 0.1484375 , -0.13183594, 0.24511719,
-0.24023438, -0.36132812, -0.12792969, 0.10595703, 0.09912109,
-0.0246582 , 0.32226562, 0.11376953, 0.18164062, 0.04931641,
-0.10253906, -0.00283813, -0.29882812, -0.171875 , -0.18945312,
-0.01367188, -0.20898438, -0.07861328, -0.0859375 , 0.05395508,
-0.14257812, -0.140625 , 0.03027344, -0.14453125, 0.359375 ,
0.16113281, 0.22265625, 0.265625 , -0.06347656, -0.02807617,
0.04760742, 0.08837891, -0.04272461, 0.05908203, 0.07128906,
0.01519775, -0.11621094, 0.07128906, 0.01403809, -0.10644531,
0.08886719, 0.11523438, 0.09667969, -0.11083984, 0.16015625,
0.3359375 , -0.1875 , 0.14550781, 0.00463867, 0.07617188,
-0.09521484, 0.08447266, 0.20117188, 0.11230469, -0.33984375,
-0.25390625, 0.05200195, 0.27539062, -0.08398438, -0.31054688,
-0.22949219, 0.14941406, -0.1953125 , 0.08496094, -0.00753784,
0.078125 , 0.05906203, 0.02355957, 0.06347656, 0.32617188,
-0.08740234, 0.10058594, -0.11474609, -0.18164062, 0.13378906,
0.11230469, -0.00080109, 0.08691406, 0.03808594, 0.0300293 ,
-0.03417969, -0.00830078, 0.14160156, -0.09619141, -0.11328125,
```

Figure 5.48: Shine Roza

Penjelasan Hasil: Pada hasil gambar shine dapat dilihat bahwa nilai pada vektor baris pertamanya adalah -0.12. Jika dibandingkan dengan gambar

clear dapat dikatakan bahwa kedua gambar tersebut tidak dapat dimasukkan pada kategori yang sama.

- Bag

```
In [15]: gmodel['bag']
Out[15]:
array([-0.03515625,  0.15234375, -0.12402344,  0.13378906, -0.11328125,
       -0.0133667,  -0.16113281,  0.14648438, -0.06835938,  0.140625 ,
       -0.06005859,  -0.3046875,  0.20996094, -0.04345703, -0.2109375 ,
       -0.05957031, -0.05053711,  0.10253906,  0.19042969, -0.09423828,
       0.18847656,  -0.07958984, -0.11035156, -0.07910156,  0.06347656,
       -0.15527344,  -0.18945312,  0.11132812,  0.27539062, -0.06787109,
       0.01806641,  0.06689453,  0.2578125,  0.0324707,  -0.24609375,
       -0.05541992,  0.01013184,  0.24121094, -0.21875,  , 0.07568359,
       -0.09814453,  -0.16113281,  0.16503906, -0.09521484, -0.16601562,
       -0.41796875,  0.0300293,  0.19433594,  0.2890625,  0.12695312,
       -0.19824219,  -0.05517578,  0.04296875, -0.10107422,  0.07324219,
       -0.13378906,  0.265625,  -0.00466919,  0.19628906, -0.10839844,
       0.14941406,  0.1484375,  0.09619141,  0.21777344, -0.08544922,
       -0.02819824,  0.02539062, -0.03759766,  0.23242188,  0.19628906,
       0.27539062,  0.09130859,  0.23730469,  0.09033203, -0.28515625,
       0.05932617,  0.06591797, -0.01794434, -0.00055313, -0.1796875 ,
       0.05615234,  -0.12207031,  -0.09863281, -0.05786133, -0.09375 ,
       -0.30273438,  -0.06396484, -0.00744629, -0.17871094,  0.08544922,
       -0.28410156,  0.33789062,  0.00228882, -0.39453125, -0.14453125,
       -0.328125,  , -0.12695312, -0.08544922,  0.15234375,  0.03662109,
       -0.1484375,  , 0.05566406,  0.02844238,  0.07519531, -0.21484375,
       -0.15722656,  0.3359375,  -0.04736328, -0.00405884, -0.19726562,
       0.27929688,  0.05566406,  -0.10058594, -0.00811768, -0.20703125,
       0.03295898,  -0.14550781,  -0.15917969,  0.16503906,  0.234375 ,
       0.03588867,  0.04296875, -0.25,  , 0.1171875, -0.07714844,
```

Figure 5.49: Bag Roza

Penjelasan Hasil: Pada hasil gambar bag dapat dilihat bahwa nilai pada vektor baris pertamanya adalah -0.035. Jika dibandingkan dengan gambar shine dapat dikatakan bahwa kedua gambar tersebut dapat dimasukkan pada kategori yang sama.

- Car

```
In [16]: gmodel['car']
Out[16]:
array([ 0.13085938,  0.00842285,  0.03344727, -0.05883789,  0.04003906,
       -0.14257812,  0.04931641, -0.16894531,  0.20898438,  0.11962891,
       0.18066406, -0.25,  , -0.10400391, -0.10742188, -0.01879883,
       0.05200195, -0.00216675,  0.06445312,  0.14453125, -0.04541016,
       0.16113281,  -0.01611328, -0.03088379,  0.08447266,  0.16210938,
       0.04467773,  -0.15527344,  0.25390625,  0.33984375,  0.00756836,
       -0.25585938,  -0.01733398, -0.03295898,  0.16308594, -0.12597656,
       -0.09912109,  0.16503906,  0.06884766, -0.18945312,  0.02832031,
       -0.05344668,  -0.03063965,  0.11083984,  0.24121094, -0.234375 ,
       0.12353516,  -0.00294495,  0.1484375,  0.33203125,  0.05249023,
       -0.20019531,  0.37695312,  0.12255859,  0.11425781, -0.17675781,
       0.10009766,  0.0030365,  0.26757812,  0.20117188,  0.03710938,
       0.11083984,  -0.09814453, -0.3125,  , 0.03515625,  0.02832031,
       0.26171875,  -0.08642578, -0.02258301, -0.05834961, -0.00787354,
       0.11767578,  -0.04296875, -0.17285156,  0.04394531, -0.23046875,
       0.1640625,  -0.11474609, -0.06030273,  0.01196289, -0.24707031,
       0.32617188,  -0.04492188, -0.11425781,  0.22851562, -0.01647949,
       -0.15039062,  -0.13183594,  0.12597656, -0.17480469,  0.02209473,
       -0.1015625,  0.00817871,  0.10791016, -0.24609375, -0.109375 ,
       -0.09375,  , -0.01623535, -0.20214844,  0.23144531, -0.05444336,
       -0.05541992,  -0.20898438,  0.26757812,  0.27929688,  0.17089844,
       -0.17578125,  -0.02770996, -0.20410156,  0.02392578,  0.03125 ,
       -0.25390625, -0.125,  , -0.05493164, -0.17382812,  0.28515625,
       -0.23242188,  0.0234375,  -0.20117188, -0.13476562,  0.26367188,
       0.00769043,  0.20507812, -0.01708984, -0.12988281,  0.04711914,
```

Figure 5.50: Car Roza

Penjelasan Hasil: Pada hasil gambar car dapat dilihat bahwa nilai pada

vektor baris pertamanya adalah -0.13. Jika dibandingkan dengan gambar bag dapat dikatakan bahwa kedua gambar tersebut dapat dimasukkan pada kategori yang sama

- Wash

```
In [17]: gmodel['wash']
Out[17]:
array([ 9.46044922e-03,  1.41601562e-01, -5.46875000e-02,  1.34765625e-01,
       -2.38281250e-01,  3.24218750e-01, -8.44726562e-02, -1.29882812e-01,
       1.07910156e-01,  2.53906250e-01,  1.13525391e-02, -1.66992188e-01,
       -2.79541016e-02,  2.08007812e-01, -4.27246094e-02,  1.05468750e-01,
       -7.42187500e-02,  3.04687500e-01,  2.11914062e-01, -8.88671875e-02,
       2.67578125e-01,  2.12890625e-01,  1.74560547e-02,  2.02941895e-03,
       6.29882812e-02,  1.62109375e-01,  1.93359375e-01,  2.17285156e-02,
       -2.67028809e-03, -9.13085938e-02, -2.38281250e-01,  2.23632812e-01,
       -8.00781250e-02, -3.80859375e-02, -1.0097656e-01, -1.39648438e-01,
       1.74884688e-01,  6.78718938e-02,  1.11328125e-01,  1.65039962e-01,
       -1.05468750e-01,  2.30712891e-02,  2.00195312e-01, -6.03027344e-02,
       -3.43750000e-01, -1.02050781e-01, -3.80859375e-01, -5.05371994e-02,
       5.07812500e-02,  1.45507812e-01,  2.81250000e-01,  7.03125000e-02,
       2.84423828e-02, -2.29492188e-01, -5.81054688e-02,  4.51660156e-02,
       -3.56445312e-02,  1.77734375e-01,  1.22070312e-01,  3.71093750e-02,
       -1.10839844e-01,  6.83593750e-02, -2.52685547e-02, -1.27929688e-01,
       4.21875000e-01,  5.32226562e-02, -3.92578125e-01,  1.74804688e-01,
       1.77001953e-02, -2.05078125e-02,  2.21679688e-01,  3.18359375e-01,
       1.08398438e-01, -4.30297812e-03, -2.45117188e-01, -2.08984375e-01,
       3.58886719e-02,  8.30078125e-02,  1.68945312e-01,  2.79541016e-02,
       1.04980469e-01, -3.47656250e-01, -5.20019531e-02,  2.24609375e-01,
       1.69677734e-02,  1.69921875e-01, -1.46484375e-01,  2.65625000e-01,
       2.17285156e-02,  1.12304688e-02, -1.14257812e-01,  7.22656250e-02,
       4.32128906e-02,  1.116943336e-02,  5.07354736e-04, -7.91015625e-02,
       -5.98144531e-02, -5.44433594e-02,  3.73046875e-01,  5.62500000e-01,
       -2.26562500e-01, -5.39550781e-02,  1.13769531e-01, -5.83496094e-02,
```

Figure 5.51: Wash Roza

Penjelasan Hasil: Pada hasil gambar wash dapat dilihat bahwa nilai pada vektor baris pertamanya adalah 9.460. Jika dibandingkan dengan gambar car dapat dikatakan bahwa kedua gambar tersebut tidak dapat dimasukkan pada kategori yang sama

- Motor

```
In [18]: gmodel['motor']
Out[18]:
array([ 5.737304469e-02,  1.50390625e-01, -4.61425781e-02, -1.32812500e-01,
       -2.59765625e-01, -1.77734375e-01,  3.68652344e-02, -4.37500000e-01,
       2.34375000e-02,  2.57812500e-01,  1.74804688e-01,  2.44140625e-02,
       -2.51953125e-01, -5.76171875e-02,  8.15429688e-02,  1.86765758e-02,
       -3.83300781e-02,  1.58203125e-01, -5.85937500e-02,  1.12304688e-01,
       1.56250000e-01, -4.24804688e-02, -1.32812500e-01,  2.11914062e-01,
       1.23046875e-01,  1.69921875e-01, -1.55273438e-01,  4.59864375e-01,
       3.02734375e-01,  1.53320312e-01, -1.69921875e-01, -1.01074219e-01,
       -3.26538086e-03,  2.28515625e-01,  8.98437500e-02, -7.12890625e-02,
       1.54296875e-01, -8.88671875e-02, -2.36328125e-01,  5.61523438e-03,
       -4.4677344e-02, -3.06640625e-01,  7.142187500e-02,  5.58593750e-01,
       -1.30859375e-01,  1.00585938e-01, -3.34472656e-02,  2.10937500e-01,
       3.10058594e-02, -6.50024414e-03,  6.34765625e-02,  4.02832031e-02,
       -2.78320312e-01,  1.07421875e-02,  1.78460938e-01,  2.80761719e-02,
       -1.50390625e-01, -1.37695312e-01,  9.96093750e-02,  1.28906250e-01,
       -3.34472656e-02, -1.08032227e-02, -2.14843750e-01, -9.52148438e-02,
       -6.39648438e-02,  7.51953125e-02, -3.06640625e-01,  2.17773438e-01,
       -2.21679688e-01,  2.33398438e-01,  5.05371094e-02, -3.37890625e-01,
       1.53320312e-01, -7.12890625e-02, -3.68652344e-02,  7.66601562e-02,
       -8.00781250e-02, -1.14257812e-01, -9.71679688e-02, -2.61718750e-01,
       3.84765625e-01, -1.87500000e-01, -1.10351562e-01,  1.00585938e-01,
       1.08398438e-01,  9.57031250e-02, -8.20312500e-02,  1.54296875e-01,
       -2.40234375e-01,  8.34960938e-02,  4.19921875e-02, -1.91650391e-02,
       9.71679688e-02,  2.52685547e-02, -5.46875000e-02, -5.88378906e-02,
       8.20312500e-02, -3.32031250e-01,  3.27148438e-02,  5.71289062e-02,
```

Figure 5.52: Motor Roza

Penjelasan Hasil: Pada hasil gambar motor dapat dilihat bahwa nilai pada vektor baris pertamanya adalah 5.737. Jika dibandingkan dengan gambar wash dapat dikatakan bahwa kedua gambar tersebut tidak dapat dimasukkan pada kategori yang sama.

- Cycle

```
In [19]: gmodel['cycle']
Out[19]:
array([ 0.04541016,  0.21679688, -0.02709961,  0.12353516, -0.20703125,
       -0.1328125,  0.26367188, -0.12890625, -0.125,      0.15332031,
       -0.18261719, -0.15820312, -0.06176758,  0.21972656, -0.15820312,
       0.02563477, -0.07568359, -0.0625,      0.04614258, -0.31054688,
       -0.13378906, -0.11669922, -0.3359375,  0.078125,      0.08447266,
       0.07226562, -0.06445312,  0.05517578,  0.14941406,  0.13671875,
       0.10302734,  0.02172852, -0.10693359,  0.02490234, -0.10644531,
       -0.05541992, -0.29492188, -0.40039062,  0.06347656, -0.08447266,
       0.17871094,  0.01165771, -0.01696777,  0.13671875, -0.1640625,
       0.11425781,  0.20800781, -0.06079102, -0.07275391,  0.15039062,
       0.18066406, -0.28515625, -0.04052734,  0.01806641,  0.00331116,
       0.00872803,  0.03564453, -0.29882812,  0.09960938, -0.1484375,
       -0.06787109,  0.05957031, -0.05517578, -0.19628906,  0.2265625,
       0.03173828, -0.07080078,  0.1484375, -0.20214844, -0.03393555,
       0.09863281, -0.02038574, -0.08789062, -0.07226562, -0.09423828,
       -0.17089844,  0.1484375,  0.10546875,  0.06445312,  0.01031494,
       -0.02636719, -0.03686523, -0.125,      -0.06787109,  0.14257812,
       0.37109375, -0.15722656,  0.09326172,  0.34960938, -0.00091553,
       -0.03613281,  0.16894531, -0.02856445,  0.10791016, -0.32421875,
       -0.14355469,  0.03173828, -0.07421875,  0.34179688,  0.140625,
       0.0043335, -0.12890625, -0.34960938, -0.02929688, -0.19628906,
       -0.2578125, -0.3671875,  0.01483154,  0.20703125,  0.09667969,
       -0.10351562, -0.31054688,  0.02844238,  0.10400391,  0.17773438,
       0.06689453, -0.1796875,  0.02783203,  0.15625,      0.02026367,
       0.0324707, -0.13476562,  0.15527344,  0.11132812, -0.01055908,
       0.07958984,  0.01989746,  0.25585938,  0.13378906,  0.02539062,
```

Figure 5.53: Cycle Roza

Penjelasan: Pada hasil gambar cycle dapat dilihat bahwa nilai pada vektor baris pertamanya adalah 0.045. Jika dibandingkan dengan gambar motor dapat dikatakan bahwa kedua gambar tersebut tidak dapat dimasukkan pada kategori yang sama.

- Perbandingan Similirati

```
In [30]: gmodel.similarity('car','faith')
Out[30]: 0.08886280140425973

In [31]: gmodel.similarity('car','fall')
Out[31]: 0.11321347547615473

In [32]: gmodel.similarity('car','sick')
Out[32]: 0.11852219525661713

In [33]: gmodel.similarity('car','clear')
Out[33]: 0.028753966364984635

In [34]: gmodel.similarity('car','shine')
Out[34]: 0.015017907091992005
```

Figure 5.54: Perbandingan Similariti Roza

Penjelasan Hasil: Pada gambar tersebut dapat dilihat bahwa perbandingan antara car dan faith tidak dapat dimasukkan pada kaegori yang sama karena memounyai nilai yang kecil. Begitu juga dengan car,clear dan car

shine. Sedangkan pada car, fall dan car, sick bisa dikatakan mempunyai kategori yang sama.

2. Fungsi Dari Extract word Dan PermuteSentences

- Extract Word

Penjelasan: Extract word bergungsi untuk mengambil suatu argumen dengan type data string.

```
In [67]: import re, string
...
...: def extract_words(line):
...:     out = (x.lower() for x in line.split())
...:     out = (re.sub('[%s]' % re.escape(string.punctuation), '',
...: x) for x in out)
...:
...:     return (x for x in out if len(x) > 0)
...:
...:
...: # Test the function
...: list(extract_words("Mohon maaf mata saya sudah mengantuk"))
Out[67]: ['mohon', 'maaf', 'mata', 'saya', 'sudah', 'mengantuk']
```

Figure 5.55: Extract Word Roza

Penjelasan Hasil: mengilustrasikan sebuah kalimat 'Mohon maaf saya sudah mengantuk' yang akan dipisahkan perkata. Dimana library re (regex module) dan library string di import terlebih dahulu. Lalu variable out mendefinisikan X untuk mengembalikan string pada objek line yang telah di split (dibagi atau dipisahkan). Re.sub digunakan untuk mengembalikan string yang diperoleh dengan mengganti kemunculan pola dalam string yang paling tidak tumpang tindih dengan repl pengganti. Kemudian, X dikembalikan berdasarkan jumlah kata. Sehingga didapat hasil output seperti pada gambar.

- PermuteSentence

Penjelasan: PermuteSentences berfungsi untuk melakukan pengocokan atau random pada teks.

```
In [3]: import string
...: string.ascii_letters
...: 'abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ'
...: import random
...: random.choice(string.ascii_letters)
...: 'j'
Out[3]: 'j'
```

Figure 5.56: PermueSentences Roza

Penjelasan: Pada gambar di atas dapat dikatakan bahwa dari huruf abjad tersebut dikocok atau di random secara ascenden. Dan hasilnya adalah huruf j.

3. Fungsi dari Librari Gensim TaggedDocument dan Doc2Vec

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

Figure 5.57: No 3 Roza

- Penjelasan Hasil: Pada baris pertama yaitu mengimport modul Tagged Document dari model gensim. Sedangkan pada baris kedua import Doc2vec dari model gensim.

4. Cara Menambahkan Data Training

Penskalaan yaitu dataset ditempatkan sambil mempertimbangkan kumpulan data linier seperti data bank.

Disintegrasi dan Komposisi: Langkah ini melibatkan pemecahan fitur tertentu untuk membangun data pelatihan yang lebih baik untuk model yang Anda pahami lebih komprehensif.

Komposisi: Ini adalah proses terakhir yang melibatkan penggabungan berbagai fitur menjadi satu fitur untuk mendapatkan data yang lebih akurat atau bermakna.

```
In [5]: for dirname in ["train/pos", "train/neg", "train/unsup", "test/
pos", "test/neg"]:
...:     for fname in sorted(os.listdir("aclImdb/" + dirname)):
...:         if fname[-4:] == '.txt':
...:             with open("aclImdb/" + dirname + "/" + fname,
encoding='UTF-8') as f:
...:                 sent = f.read()
...:                 words = extract_words(sent)
...:                 unsup_sentences.append(TaggedDocument(words,
[dirname + "/" + fname])))

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

In [7]: with open("stanfordSentimentTreebank/original_rt_snippets.txt",
encoding='UTF-8') as f:
...:     for i, line in enumerate(f):
...:         words = extract_words(sent)
...:         unsup_sentences.append(TaggedDocument(words, ["rt-%d" %
i]))
```

Figure 5.58: No 4 Roza

- Penjelasan Hasil: Membaca direktori name dari data yang ada di dalam kurung. Pada kodingan di atas ada 3 data. Data pertama yaitu: train/-pos, train/neg, train/unsuo, test/pos dan test/neg. Data kedua yaitu: txtsentoken/pos dan txt/sentokenneg. Sedangkan data ke 3 yaitu: standfordSentimentTree-Bank/Originalrtsnippes.txt

5. Kenapa Harus Dilakukan Pengocokan dan Pembersih Data

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

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

Figure 5.59: No 5 Roza

- Penjelasan Hasil: Pada gambar pertama yaitu melakukan pengacakan model terhadap data unsupervised learning. Dan kemudian baru membuat modelnya setelah dilakukan pengacakan terhadap yang pertama tadi.

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

Figure 5.60: No 5 Roza

- Penjelasan Hasil: Mengimport Library Re. Kemudian membuat fungsi untuk menghapus tag html dan perkocakan. Dimana di dalam variabel ini ada kodingan untuk menghapus tag html yaitu petik satu, tanda baca dan spasi yang berurutan.

6. Model harus di Save dan Kenapa Temporari Training Harus Dihapus

Kenapa model harus di save berfungsi untuk mencegah ram agar tidak lemot. Sedangkan kenapa temporari training harus dihapus mengosongkan memori agar sedikit lega atau tidak lemot.

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

Figure 5.61: No 6 Roza

- Penjelasan Hasil: Menghapus Temporary Training Data

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

Figure 5.62: No 6 Roza

- Penjelasan Hasil: Menyimpan model rozroza.d2v

7. InferCode

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

```
In [76]: model.infer_vector(extract_words("Roza LDR an jauh parah"))
Out[76]:
array([-0.01481314, -0.00188257,  0.01908973,  0.00728912,  0.0266825 ,
       0.00738126, -0.01835221, -0.0488576 ,  0.00070386,  0.0105413 ,
       0.00830346, -0.02532332,  0.02238536, -0.00789794,  0.00118123,
       0.00567044,  0.01542506, -0.03314974, -0.0027141 , -0.00166321,
      -0.0061044 , -0.00256514, -0.01496234,  0.04006485, -0.02149059,
      -0.0164069 , -0.00824546, -0.00944306, -0.02430394,  0.01084839,
      0.01695666,  0.04544538, -0.01500919,  0.01889228, -0.02264787,
      0.01594709, -0.01795742, -0.01759925,  0.04112145, -0.00495921,
      0.00145809, -0.02969404,  0.04033814,  0.01220725,  0.03532154,
     -0.00558988, -0.00160033,  0.01057541,  0.01741943,  0.0217849 ,
      0.01274981,  0.01111757], dtype=float32)
```

Figure 5.63: No 7 Roza

- Penjelasan Hasil: Pada hasil gambar tersebut dapat disimpulkan bahwa kalimat "Roza LDR an Jauh Parah" menghasilkan outputan berupa array seperti angka-angka di bawah.

8. Cosine Similarity.

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

Figure 5.64: No 8 Roza

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

9. Score Dari Cross Validation

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

Figure 5.65: No 9 Roza

- Penjelasan Hasil: Menghitung model clrf, sentvecs, sentiments serta juga menghitung keakuratannya.

5.2.3 Penanganan Error

(a) Skrinsut Error

```
In [3]: len(unsup_sentences)
Traceback (most recent call last):
File "<ipython-input-3-bbbab3688fee>", line 1, in <module>
len(unsup_sentences)
NameError: name 'unsup_sentences' is not defined
```

Figure 5.66: Error Roza

- (b) Kode Error dan Jenis Errornya

Kode Error: "NameError: name 'unsup sentences is not defined".

Jenis Error: Unsup Sentence Not Defined

- (c) Penanganan

Running ulang dari awal, sehingga unsupervide baru bisa ditemukan.

5.3 Fadila-1164072

5.3.1 Teori

Penjelasan Tugas Harian 9 (No 1-6). (No. 7 Bukti Plagiarisme)

1. Mengapa Kata-Kata Harus Di Lakukan Vektorisasi Dan Ilustrasi Gambar

- Penjelasan :

Alasan mengapa kata-kata harus dilakukan vektorisasi terlebih dahulu yaitu dikarenakan mesin hanya mampu membaca data dengan bentuk angka. Berdasarkan hal tersebut maka tentunya diperlukan vektorisasi kata atau bisa disebut dengan mengubah kata menjadi bentuk vektor agar mesin seolah-olah paham apa yang kita maksudkan dan dapat memproses aktifitas/perintah dengan benar. Kata juga harus di vektorisas iuntuk mengetahui presentase kata yang sering muncul dalam setiap kalimatnya, yang berguna untuk menetukan kata kunci.

- Ilustrasi Gambar

Penjelasan : Berdasarkan penjelasan diatas, ada beberapa contoh yang bisa diterapkan. Untuk salah satu contoh dari vektorisasi kata dapat diliat pada gambar berikut 5.67.

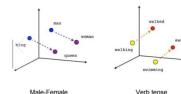


Figure 5.67: Vektorisasi Kata-fadila

2. Mengapa Dimensi Dari Vektor Dataset Google Bisa Mencapai 300 Dan Ilustrasi Serta Contoh Gambar

- Penjelasan :

Dimensi dari Vektor Dataset Google Bisa Mencapai 300 itu dikarenakan pada masing-masing objek yang terdapat pada dataset akan memiliki identitasnya tersendiri, selain itu juga untuk nilai dalam vektor 300 dimensi yang terkait dalam sebuah kata "dioptimalkan" dalam berbagai hal untuk menangkap aspek yang berbeda dari makna dan penggunaan kata itu. secara singkatnya terdapat ada lebih dari 3 miliar kata-kata dan kalimat yang tidak mungkin disimpan dalam 1 dimensi vektor, lalu disimpan menjadi 300 dimensi vektor untuk mengatasi yang namanya kegagalan memori

- Ilustrasi :(berdasarkan identitas tersendiri) Apabila dicontohkan dengan penjelasan yang lebih rinci maka dilakukan perumpamaan sederhana. Misalnya untuk sebuah dataset google yang memiliki 3 buah objek yaitu berat, lebar, dan tinggi. Kemudian dari masing-masing objek tersebut dilakukan perbandingan antara berat dan lebar beserta berat dan tinggi. Hasil yang didapatkan akan memiliki presentasi yang berbeda sehingga dapat diartikan bahwa mesin dapat membedakan objek yang hampir serupa namun tak sama.

- Contoh Gambar

Penjelasan : Berdasarkan penjelasan diatas, ada beberapa contoh yang lain bisa diterapkan. Untuk salah satu contoh dari vektor dataset google dapat diliat pada gambar berikut 5.68 dan 5.69.

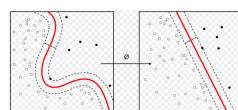


Figure 5.68: Dimensi Vektor Dataset Wikipedia-fadila

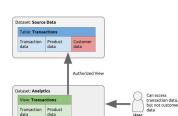


Figure 5.69: Dimensi Vektor Dataset -fadila

3. Konsep Vektorisasi Untuk Kata Dan Ilustrasi Gambar

- Penjelasan :

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

- **Ilustrasi Gambar**

Penjelasan : Berdasarkan konsep diatas, ada beberapa contoh yang bisa diterapkan. Untuk salah satu contoh dari vektorisasi kata dapat diliat pada gambar berikut 5.70.

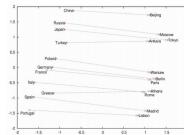


Figure 5.70: Vektorisasi Untuk Kata-fadila

4. Konsep Vektorisasi Untuk Dokumen Dan Ilustrasi Gambar

- **Penjelasan :**

Untuk vektorisasi dokumen sebenarnya terbilang sama dengan konsep vektorisasi kata, yang membedakan hanya pada proses awalnya (pada eksekusi awal). Untuk vektorisasi dokumen ini, mesin akan membaca semua kalimat yang terdapat pada dokumen tersebut, kemudian kalimat yang terdapat pada dokumen tersebut akan dipecah menjadi kata-kata. Seperti itulah konsep vektorisasi dokumen.

- **Ilustrasi Gambar**

Penjelasan : Berdasarkan konsep diatas, ada beberapa contoh yang bisa diterapkan. Untuk salah satu contoh dari vektorisasi dokumen dapat diliat pada gambar berikut 5.71.

5. Pengertian Mean Dan Standar Deviasi Beserta Ilustrasi Gambar

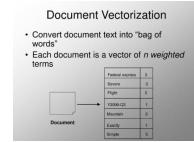


Figure 5.71: Vektorisasi Untuk Dokumen-fadila

- Pengertian Mean :

Mean merupakan nilai rata-rata dari suatu data. Mean sendiri dapat dicari dengan cara membagi jumlah data dengan banyak data sehingga diperolehlah nilai rata-rata dari suatu data yang dicari / tersebut.

- Pengertian Standar Deviasi :

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

- Ilustrasi Gambar

Penjelasan : Berdasarkan penjelasan diatas, ada beberapa contoh yang bisa diterapkan. Untuk salah satu contoh dari mean dan standar deviasi sendiri dapat diliat pada gambar berikut 5.72 , 5.73 dan 5.74.

Population Mean	Sample Mean
$\mu = \frac{\sum x_i}{N}$	$\bar{X} = \frac{\sum x_i}{n}$

Figure 5.72: Mean-fadila



Figure 5.73: Mean Lanjutan-fadila

Rumus Deviasi Standar

$$SD = \sqrt{\frac{\sum x^2}{N}}$$

Keterangan:
 SD = Standar Deviasi
 $\sum x^2$ = Jumlah semua deviasi sempit.
 (6) Rumus untuk frekuensi simpel atau

$$SD = \sqrt{\frac{\sum f x^2}{N}}$$

Figure 5.74: Standar Devisiasi-fadila

6. Penjelasan Skip-gram Dan Ilustrasi Gambar

- Penjelasan :

Sebuah teknik yang digunakan di area speech processing, dimana n-gram yang dibentuk kemudian ditambahkan juga dengan tindakan “skip” pada token-tokennya.

Untuk membentuk k-skip-n-grams, ada dua nilai yang harus didefinisikan, dimana kedua nilai tersebut yaitu k (jumlah kata yang di-skip) dan n (banyak kata dalam n-gram, e.g. bigram (2-gram), trigram (3-gram), dll.).

- Ilustrasi Gambar

Penjelasan : Berdasarkan penjelasan diatas, ada beberapa contoh yang bisa diterapkan. Untuk salah satu contoh dari skip-gram sendiri dapat diliat pada gambar-gambar berikut 5.76 .

Continuous Skip-gram Model

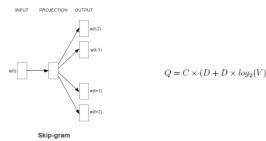


Figure 5.75: Skip Gram - fadila

7. Plagiarisme Fadila :

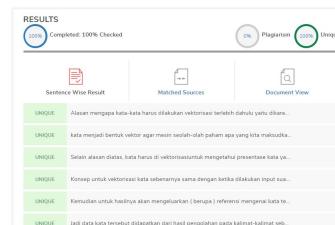


Figure 5.76: Plagiarisme - fadila

5.3.2 Praktek

Penjelasan Tugas Harian 10 (No 1-9). (Dan Penanganan Error)

1. Percobaan Google Dataset (Perbandingan Dan Similarity) Untuk Beberapa Data Berikut :

Note : Penjelasan setiap datasetnya saya berikan perbandingan langsung untuk contoh yang lebih jelas sebelum dilakukan similarity (dengan dataset tertentu).

(a) Love : Berpatokan pada gambar 5.77.

Penjelasan : Pada hasil gambar love dapat dilihat bahwa nilai pada vektor baris pertamanya yaitu sebesar -0.10.

```
In [64]: gmodel['love']
Out[64]:
array([-0.18902754, -0.15234375,  0.02587891,  0.18593986, -0.16893986,
       0.06684531,  0.2526875,  0.140925,  0.20121708,
       0.02624512, -0.08283125, -0.26371788, -0.140925,  0.23535156,
       0.140925,  0.140925,  0.140925,  0.140925,  0.140925,
       -0.03857422,  0.07958984,  0.22949219,  0.14354609,  0.16798875,
       -0.03515625,  0.05517578,  0.16093359,  0.11161641, -0.16308594,
       0.11161641,  0.11161641,  0.11161641,  0.11161641,
       0.07714844,  0.26121875,  0.08642578, -0.02554464,  0.13396438,
       0.18652344, -0.2096064,  0.07808879,  0.02606459, -0.16644511,
       0.13034641,  0.13034641,  0.13034641,  0.13034641,
       -0.19986328,  0.14941406, -0.16903359,  0.01553396,  0.08984375,
       0.11230469, -0.0470117, -0.11376953, -0.005784, -0.01828846,
       0.24316486,  0.00447264, -0.07988875,  0.10965466,  0.05325625,
```

Figure 5.77: Google Dataset - Love - fadila

(b) Faith : Berpatokan pada gambar 5.78.

Penjelasan : Pada hasil gambar faith dapat dilihat bahwa nilai pada vektor baris pertamanya yaitu sebesar -0.26. Jika dibandingkan dengan gambar love dapat dikatakan bahwa kedua gambar tersebut dapat dimasukkan pada kategori yang sama dikarenakan nilainya mendekati/dekat/hampir sama.

```
In [65]: gmodel['faith']
Out[65]:
array([ 0.26367185, -0.40153691,  0.195115,  0.1576565,  0.14464845,
       0.06648625,  0.04515793,  0.13031562,  0.12207831,  0.13475951,
       0.06648625,  0.18945312,  0.16881562,  0.1679886, -0.04354511, -0.23535156,
       0.1285125,  0.18449219,  0.36181281,  0.1953125, -0.18164482,
       0.15332125,  0.15332125,  0.15332125,  0.15332125,  0.15332125,
       -0.05957831,  0.22949219, -0.08684248,  0.26171875,  0.16987734,
       -0.1326125,  0.21464375,  0.01135254,  0.06171816,  0.18556734,
       0.044377,  0.044377,  0.044377,  0.044377,  0.044377,
       0.3125,  0.06648625, -0.17675781, -0.01789884, -0.1546825,
       -0.02328386,  0.02328386,  0.02328386,  0.02328386,  0.02328386,
       -0.25253862,  0.28605469,  0.03951562, -0.20214484, -0.1875,
       0.16995105,  0.16995105,  0.16995105,  0.16995105,  0.16995105,
       0.11232125,  0.09566271,  0.12255952,  0.09014452, -0.04959895,
       -0.02323154, -0.04240847, -0.02823125,  0.10815625,  0.04193951,
       -0.16601562, -0.13671875,  0.09819141,  0.26217185,  0.0251955,
```

Figure 5.78: Google Dataset - Faith- fadila

(c) Fall : Berpatokan pada gambar 5.79.

Penjelasan : Pada hasil gambar fall dapat dilihat bahwa nilai pada vektor baris pertamanya yaitu sebesar -0.04. Jika dibandingkan dengan gambar faith dapat dikatakan bahwa kedua gambar tersebut tidak dapat dimasukkan pada kategori yang sama karena nilainya berbeda (tidak dalam range yang dekat).

```
In [66]: gmodel['fall']
Out[66]:
array([-0.94272461,  0.18742185, -0.09277344,  0.16894531, -0.1328125,
       0.08912466,  0.04462188,  0.0145874,  0.08901496, -0.10824219,
       -0.11805156,  0.01092529, -0.0380781, -0.019209, -0.1953125,
       0.14032125,  0.14032125,  0.14032125,  0.14032125,  0.14032125,
       0.03437969,  0.2180375,  0.01977539,  0.125,  0.0154180,
       -0.26953125, -0.0986377, -0.07763672, -0.15527344, -0.03393555,
       0.0434125,  0.0434125,  0.0434125,  0.0434125,  0.0434125,
       0.13574219, -0.22558954,  0.33799862, -0.03564453, -0.10838844,
       -0.1935958,  0.0546375, -0.0495695,  0.3571875, -0.13293886,
       0.10203125,  0.10203125,  0.10203125,  0.10203125,  0.10203125,
       -0.06935954,  0.05824961,  0.01422113, -0.01397795, -0.05395986,
       -0.8255127,  0.0625125,  0.07808869, -0.07617185,  0.05542969,
       -0.2035125,  0.01312125,  0.03803996,  0.01312125,  0.01312125,
       0.2189375,  0.04161793,  0.08789862,  0.19628986,  0.11376955,
       0.09542969,  0.03125,  0.12988281,  0.02278988,  0.14556781,
```

Figure 5.79: Google Dataset - Fall - fadila

- (d) Sick : Berpatokan pada gambar 5.80.

Penjelasan : Pada hasil gambar sick dapat dilihat bahwa nilai pada vektor baris pertamanya yaitu sebesar 1,8. Jika dibandingkan dengan gambar fall dapat dikatakan bahwa kedua gambar tersebut tidak dapat dimasukkan pada kategori yang sama karena nilainya berbeda (tidak dalam range yang dekat).

```
In [67]: gmodel['sick']
Out[67]:
array([-1.8207186e-01, 1.4941486e-01, -4.6527545e-02, 1.6460250e-01,
       3.2236623e-01, 1.7382312e-01, -1.4764893e-01,
       1.01874213e-01, 5.4687500e-02, 1.6999218e-01, -1.6894531e-01,
      2.24304139e-03, 9.66796875e-02, -1.6601562e-01, -1.1230460e-01,
      1.4661250e-01, 1.1230460e-01, 1.1230460e-01, -1.4661250e-01,
      8.74823439e-02, -2.19547956e-02, 3.41796875e-01, 4.39886875e-02,
      1.79718936e-01, -9.18221289e-02, 8.86671875e-02, -1.9312590e-01,
      1.03921289e-01, 1.1230460e-01, 1.1230460e-01, -1.4661250e-01,
      9.42382312e-02, -3.12580000e-02, 1.8997440e-02, -6.3964843e-02,
      1.18694531e-01, -1.33846875e-01, -6.0927440e-02, 4.06250000e-01,
      3.2889339e-01, -1.33846875e-01, -6.0927440e-02, 4.06250000e-01,
      5.78613281e-02, -1.04980446e-01, -1.69945312e-01, -8.98701250e-02,
      -2.01017188e-01, 1.1230460e-01, 1.1230460e-01, -1.4661250e-01,
      5.56640825e-02, 3.14451320e-01, 9.61523430e-02, -1.20117188e-01,
      7.12996250e-02, 4.37911719e-02, 2.69878125e-01, 5.71299625e-02,
      8.44726562e-02, 2.15821590e-01, 1.69953250e-01, 8.799863250e-02,
```

Figure 5.80: Google Dataset - Sick - fadila

- (e) Clear : Berpatokan pada gambar 5.81.

Penjelasan : Pada hasil gambar clear dapat dilihat bahwa nilai pada vektor baris pertamanya yaitu sebesar -2.4. Jika dibandingkan dengan gambar sick dapat dikatakan bahwa kedua gambar tersebut tidak dapat dimasukkan pada kategori yang sama karena nilainya berbeda (tidak dalam range yang dekat)..

```
In [68]: gmodel['clear']
Out[68]:
array([2.4414625e-04, -1.02959781e-01, -1.09414062e-01, -4.2480468e-02,
       -1.67968750e-01, -1.46484378e-01, 1.76797812e-01, 1.46464375e-01,
       1.01874213e-01, 2.23632812e-01, -2.13867189e-01, 3.10805854e-02,
      1.24511719e-01, 2.23632812e-01, -2.13867189e-01, 3.10805854e-02,
      1.00195312e-01, -4.66974210e-02, -6.35953750e-02, -1.21803750e-01,
      1.23231250e-01, -1.23231250e-01, -1.23231250e-01, 1.23231250e-01,
      -2.75879906e-02, -6.04240847e-03, -7.37784468e-02, -1.72815152e-01,
      9.66796875e-02, -4.91333980e-03, -1.78710938e-01, -1.40308059e-03,
      1.00195312e-01, -4.66974210e-02, -6.35953750e-02, -1.21803750e-01,
      1.99794669e-02, -3.14351053e-03, 1.39015254e-02, 3.54472556e-02,
      3.55859375e-01, -7.12089625e-02, 2.83283125e-01, 2.75308025e-01,
      1.00195312e-01, -4.66974210e-02, 2.83283125e-01, 2.75308025e-01,
      3.4912121894e-02, 7.32421375e-02, 2.17773435e-01, 5.00848232e-02,
      1.47095182e-02, 1.25900000e-01, -5.72789500e-02, -2.74658250e-02,
      -1.37251825e-02, 1.25900000e-01, -6.27923962e-01, -1.46612500e-01,
      -1.93828125e-01, 1.32812590e-01, -5.00848232e-02, -1.66015625e-01,
```

Figure 5.81: Google Dataset - Clear - fadila

- (f) Shine : Berpatokan pada gambar 5.82.

Penjelasan : Pada hasil gambar shine dapat dilihat bahwa nilai pada vektor baris pertamanya yaitu sebesar -0.12. Jika dibandingkan dengan gambar clear dapat dikatakan bahwa kedua gambar tersebut tidak dapat dimasukkan pada kategori yang sama karena nilainya berbeda (tidak dalam range yang dekat).

- (g) Bag : Berpatokan pada gambar 5.83.

Penjelasan : Pada hasil gambar bag dapat dilihat bahwa nilai pada vektor baris pertamanya yaitu sebesar -0.03. Jika dibandingkan dengan gambar shine dapat dikatakan bahwa kedua gambar tersebut dapat dimasukkan

```
In [69]: gmodel['shine']
Out[69]:
array([-0.11402344,  0.2597662, -0.15917969,  0.27734375,  0.36273485,
       0.09946938,  0.95270721, -0.12949319,  0.18539975,  0.2671679,
      -0.10302734,  0.13671875,  0.25398625,  0.07238696,  0.02530862,
      0.2177344,  0.2482348,  0.25234375,  0.21238468, -0.19335998,
      0.05892345,  0.11421381,  0.22234375,  0.22234375,  0.22234375,
      0.20849531,  0.38865933,  0.0401256,  0.05802397,  0.14646438,
      0.54756525,  0.03526517, -0.13925781, -0.05491364, -0.14343292,
      0.24214242,  0.10435313,  0.10435313,  0.10435313,  0.10435313,
      0.390625, -0.2108375,  0.1484375, -0.13185394,  0.24511719,
      0.2484125,  0.10435313,  0.10435313,  0.10435313,  0.10435313,
      0.02465852,  0.12225562,  0.11376953,  0.21816482,  0.04931164,
      0.18233966, -0.08023813, -0.29882812, -0.1718175, -0.18945312,
      0.03332181,  0.10435313,  0.10435313,  0.10435313,  0.10435313,
      0.14257812, -0.148625,  0.03907344, -0.14453125,  0.559375,
      0.16113261,  0.2225625,  0.265625, -0.05347656, -0.26287617,
      0.04704162,  0.11621894,  0.11621894,  0.11621894,  0.11621894,
      0.01519775, -0.11621894,  0.47128986,  0.01405389, -0.18644531,
```

Figure 5.82: Google Dataset - Shine - fadila

pada kategori yang sama dikarenakan nilainya mendekati/dekat/hampir sama.

```
In [70]: gmodel['bag']
Out[70]:
array([-0.03515625,  0.16113261,  0.14646438,  0.06865593,  0.148625,
       0.08668519,  0.2597662, -0.16949319,  0.20896438,  0.11962091,
      -0.14257812,  0.03907344,  0.02530862,  0.02530862,  0.02530862,
      0.02530862,  0.02530862,  0.02530862,  0.02530862,  0.02530862,
      0.18847656, -0.07958984, -0.11035156,  0.07910156,  0.06347656,
      0.15332181,  0.10435313,  0.10435313,  0.10435313,  0.10435313,
      0.21186654,  0.06860453,  0.2578125,  0.0524797, -0.24689375,
      0.05541992,  0.01813184,  0.24121094,  0.01875,  0.07563359,
      0.09002167,  0.10435313,  0.10435313,  0.10435313,  0.10435313,
      -0.41798875,  0.0398295,  0.19433594,  0.23896425,  0.12693312,
      -0.19824219, -0.05517578,  0.04266875, -0.01807422,  0.20732425,
      -0.13377395,  0.03907344,  0.02530862,  0.02530862,  0.02530862,
      0.14941466,  0.1484375,  0.09019314,  0.2177344, -0.0854922,
      -0.02819624,  0.02530862, -0.0375976,  0.21241285,  0.16262986,
      0.17332181,  0.10435313,  0.10435313,  0.10435313,  0.10435313,
      0.05952027,  0.05517579, -0.01794454, -0.00053113, -0.0798075,
      0.05615224, -0.12287031, -0.09863281, -0.0787813, -0.09375,
```

Figure 5.83: Google Dataset - Bag - fadila

(h) Car : Berpatokan pada gambar 5.84.

Penjelasan : Pada hasil gambar car dapat dilihat bahwa nilai pada vektor baris pertamanya yaitu sebesar 0.13. Jika dibandingkan dengan gambar bag dapat dikatakan bahwa kedua gambar tersebut tidak dapat dimasukkan pada kategori yang sama karena nilainya berbeda (tidak dalam range yang dekat).

```
In [71]: gmodel['car']
Out[71]:
array([ 0.13085938,  0.00842285, -0.0334727, -0.05883789,  0.04003986,
       -0.14257812,  0.0931641,  0.14646438,  0.06865593,  0.11962091,
      -0.14257812,  0.03907344,  0.02530862,  0.02530862,  0.02530862,
      0.02530862,  0.02530862,  0.02530862,  0.02530862,  0.02530862,
      0.18847656, -0.09216675,  0.06445312, -0.14531125, -0.045454916,
      0.14111321, -0.01611328, -0.03988379,  0.08447266,  0.16210938,
      0.04265852,  0.10435313,  0.10435313,  0.10435313,  0.10435313,
      -0.25589398, -0.07373398, -0.03295089,  0.16308594, -0.12597566,
      -0.06991289,  0.15681596,  0.06884765, -0.18045312,  0.02530861,
      0.04265852,  0.10435313,  0.10435313,  0.10435313,  0.10435313,
      -0.12355161, -0.06294495,  0.1484375,  0.33203125,  0.0524797,
      0.12355161, -0.06294495,  0.1484375,  0.33203125,  0.0524797,
      0.11083984, -0.09814453, -0.3125,  0.03515625,  0.02832031,
      0.21173875, -0.00545376, -0.02530861, -0.0534921, -0.08787554,
      0.17021181,  0.10435313,  0.10435313,  0.10435313,  0.10435313,
      0.1640625, -0.11474699, -0.06936273,  0.01196209, -0.24797031,
      0.22617181, -0.04040238, -0.11425781,  0.22051562, -0.01547949,
      -0.19897961, -0.11185394,  0.13297056, -0.17488469,  0.02309475,
```

Figure 5.84: Google Dataset - Car - fadila

(i) Wash : Berpatokan pada gambar 5.85.

Penjelasan : Pada hasil gambar wash dapat dilihat bahwa nilai pada vektor baris pertamanya yaitu sebesar 9,4. Jika dibandingkan dengan gambar car dapat dikatakan bahwa kedua gambar tersebut tidak dapat dimasukkan pada kategori yang sama karena nilainya berbeda (tidak dalam range yang dekat).

(j) Motor : Berpatokan pada gambar 5.86.

```
[23]: gmodel['wash'] = 
Out[23]: array([ 0.46064228e-02,  1.4606502e-02,  5.46673980e-02,  1.3076532e-01,
   3.2823504e-01,  2.3207510e-01,  44.7472350e-02,  1.2982181e-01,  1.0791556e-01,
   1.7936025e-01,  1.5253591e-01,  0.6961228e-01,  1.0962188e-01,  1.0791556e-01,
   2.4172806e-02,  1.2609750e-01,  1.1934670e-01,  1.0962188e-01,  1.0791556e-01,
   6.2988120e-02,  1.6209375e-01,  1.1934670e-01,  1.0962188e-01,  1.0791556e-01,
   6.7629880e-01,  1.3988505e-01,  3.2682120e-01,  2.2531263e-01,  1.0791556e-01,
   1.7484688e-01,  1.7679193e-01,  1.1321252e-01,  1.6593862e-01,  1.0791556e-01,
   1.7484688e-01,  1.7679193e-01,  1.1321252e-01,  1.6593862e-01,  1.0791556e-01,
   3.4579800e-01,  1.4595781e-01,  1.8893575e-01,  1.05371594e-01,  1.0791556e-01,
   8.7815008e-02,  1.4595781e-01,  1.2805800e-01,  2.0120500e-01,  1.0791556e-01,
   1.0791556e-01,  1.0791556e-01,  1.0791556e-01,  1.0791556e-01,  1.0791556e-01,
   3.56443812e-01,  1.6593795e-02,  1.25685174e-01,  1.37198379e-01,  1.0791556e-01,
   -1.16833124e-01,  1.6593795e-02,  1.25685174e-01,  1.37198379e-01,  1.0791556e-01])
```

Figure 5.85: Google Dataset - Wash - fadila

Penjelasan : Pada hasil gambar motor dapat dilihat bahwa nilai pada vektor baris pertamanya yaitu sebesar 5,7. Jika dibandingkan dengan gambar wash dapat dikatakan bahwa kedua gambar tersebut tidak dapat dimasukkan pada kategori yang sama karena nilainya berbeda (tidak dalam range yang dekat).

```
[17]: gpd1[0].get('rect')
Out[17]:
array([ 3.7370048e-01,  1.3099063e-01, -0.4102573e-01, -1.3381259e-01,
       -0.9795625e-01, -1.7774375e-01,  3.8605234e-02, -1.7950880e-02,
       2.5170000e-02,  2.5170000e-02,  1.7480000e-02,  2.4416000e-02,
       2.5170000e-02,  2.5170000e-02,  1.7480000e-02,  2.4416000e-02,
       1.3380781e-01,  1.8823125e-01,  5.9379596e-02,  1.2984476e-01,
       1.3380781e-01,  1.8823125e-01,  5.9379596e-02,  1.2984476e-01,
       3.6273475e-01,  1.5532012e-01,  1.6912781e-01,  1.0174310e-01,
       3.6273475e-01,  1.5532012e-01,  1.6912781e-01,  1.0174310e-01,
       1.3485675e-01,  1.8260125e-01,  2.3621252e-01,  1.5623438e-01,
       1.3485675e-01,  1.8260125e-01,  2.3621252e-01,  1.5623438e-01,
       1.30859375e-01,  1.0659832e-01,  3.3472655e-02,  4.2893725e-02,
       1.30859375e-01,  1.0659832e-01,  3.3472655e-02,  4.2893725e-02,
       3.10858945e-02,  1.0659832e-01,  3.3472655e-02,  4.2893725e-02,
       3.10858945e-02,  1.0659832e-01,  3.3472655e-02,  4.2893725e-02,
       -3.3472655e-02,  1.76802227e-01,  9.24867597e-02,  1.29516843e-01])
```

Figure 5.86: Google Dataset - Motor - fadila

(k) Cycle : Berpatokan pada gambar 5.87.

Penjelasan : Pada hasil gambar cycle dapat dilihat bahwa nilai pada vektor baris pertamanya yaitu sebesar 0,04. Jika dibandingkan dengan gambar motor dapat dikatakan bahwa kedua gambar tersebut tidak dapat dimasukkan pada kategori yang sama karena nilainya berbeda (tidak dalam range yang dekat).

Figure 5.87: Google Dataset - Cycle - fadila

(l) Similarity : (Berbeda Dengan Teman) : Berpatokan pada gambar 5.88.

Penjelasan : (Berdasarkan yang saya pahami dan kerjakan) Pada gambar diatas terdapat 5 perbandingan jelas untuk setiap datasetnya. Untuk nilai yang mendekati (yang paling besar) maka bisa dikategorikan dengan jenis/kelompok yang sama dengan satu sama lain. Sebenarnya nilai 0, itu kecil namun karena ada perbandingan perbesaran yang signifikan (jelas)

) maka bisa dikelompokkan dengan lebih mudah. Untuk kategori yang sama berdasarkan contoh gambar yaitu ada (bag,cash), (faith,love) dan (shine,clear).

```
In [75]: gmodel.similarity('love', 'sick')
Out[75]: 0.2665361473523943

In [76]: gmodel.similarity('fall', 'faith')
Out[76]: 0.05692647791944066

In [77]: gmodel.similarity('shine', 'clear')
Out[77]: 0.1165274575383886

In [78]: gmodel.similarity('car', 'bag')
Out[78]: 0.2531557716352525

In [79]: gmodel.similarity('wash', 'motor')
Out[79]: 0.10288077965687967
```

Figure 5.88: Google Dataset - Similarity - fadila

2. Penjelasan Dan Ilustrasi Extract_Words Dan PermuteSentences

- Extract_Words : Berpatokan pada gambar 5.89 dan 5.90.

Untuk Extract_Words berfungsi untuk memecahkan kata menjadi beberapa komponen atau lebih mudahnya kata yang dieksekusi dipecah kemudian dikelompokkan sesuai dengan keinginan (tujuan pemecahan).

- Penjelasan Pertama (Keseluruhan Maksud Gambar): 5.89

Pada hasil yang didapatkan untuk contoh ini yaitu terdapat satu kalimat untuk yang terbentuk dari 5 kata. Kata tersebut kemudian dipisahkan menggunakan perintah (`test_string.split`) kemudian hasil keluarannya akan di print dengan parameter tambahan kata sehingga memberikan penjelasan ataupun tanda yang lebih jelas terhadap perbedaan eksekusi kata ketika masih terangkai dan ketika telah terpecahkan.

```
In [1]: test_string = "Padilla Berasal Dari Sulawesi Selatan"
.....
.... # printing original string
.... print ("String Originalnya adalah : " + test_string)
.....
.... # using split()
.... # to extract words from string
.... res = test_string.split()
.....
.... # printing result
.... print ("Splitnya adalah : " + str(res))
String Originalnya adalah : Padilla Berasal Dari Sulawesi Selatan
Splitnya adalah : ['Padilla', 'Berasal', 'Dari', 'Sulawesi', 'Selatan']
```

Figure 5.89: 1- Extract-Word - fadila

- Penjelasan Kedua (Keseluruhan Maksud Gambar): 5.90

Gambar tersebut didalmnya telah mengilustrasikan sebuah kalimat 'Fadila Bestfriendnya Cahyoni Forever' yang akan dipisahkan perkata. Dimana library re (regex module) dan library string di import terlebih dahulu. Kemudian untuk variable out mendefinisikan X untuk mengembalikan string pada objek line yang telah di split yang eksekusi

perintah split itu diartikan sebagai (dibagi atau dipisahkan). Kemudian, X dikembalikan berdasarkan jumlah kata yang ada, sehingga muncullah hasil seperti pada gambar yang dijadikan sebagai keluaran.

Figure 5.90: 2- Extract-Word - fadila

- `PermuteSentences` : Berpatokan pada gambar 5.91

PermuteSentences digunakan untuk melakukan pengocokan (shuffle ataupun random) pada text yang diinginkan.

Penjelasan (Keseluruhan Maksud Gambar) :

Pada gambar tersebut mengeluarkan sebuah hasil (keluaran) yang telah berupa pengacakan/pengocokan kata/huruf/kalimat yang telah didefinisikan pada perintah untuk dieksekusi. Kata Dila di random dengan beberapa kali sehingga memberikan hasil yang beragam dimana semuanya berbentuk string dan dieksekusi berupa len. Untuk return dari inputan yang dieksekusi tersebut di definisikan dengan pemanggilan variabel next_list.

Figure 5.91: PermuteSentences - fadila

3. Library Gensim TaggedDocument Dan Doc2Vec

- Gensim TaggedDocument : Tagged Document merupakan class yang digunakan dalam library gensim yang dicontohkan dimana tagged dokument yang 'memisahkan informasi dan struktur dari presentasi' dengan menggunakan tag .
 - Doc2Vec : Untuk membuat representasi numerik dari suatu dokumen, terlepas dari panjangnya. Model doc2vec dapat digunakan dengan cara berikut: untuk pelatihan, diperlukan seperangkat dokumen.

```
In [2]: from gensim.models.doc2vec import TaggedDocument
...: from gensim.models import Doc2Vec
```

Figure 5.92: 3-tagged - fadila

- Contoh / Ilustrasi Gambar : 5.92

Penjelasan (Maksud Dari Gambar Dan Code Secara Keseluruhan) :

PPada gambar tersebut direalisasikan Perintah ” import ” yaitu digunakan untuk pemanggilan class tagged document dan model doc2vec yang masing-masing berada pada bagian model library gensim. Tidak terjadi error maka pemanggilan kedua perintah tersebut berhasil dan terbukti pemraktekannya.

4. Menambahkan Data Training (Melatih model doc2vac)

- Penjelasan Untuk Penambahan Data Training : Ada beberapa point penting yang bisa diperhatikan dan dipertimbangkan pada penambahan data yaitu

- Disintegrasi dan Komposisi: Langkah ini melibatkan pemecahan fitur tertentu untuk membangun data pelatihan yang lebih baik untuk model yang Anda pahami lebih komprehensif sehingga menghasilkan hasil yang lebih tepat dan efisien.
- Penskalaan dimana dataset ditempatkan sambil mempertimbangkan kumpulan data linier seperti data bank ataupun data lainnya.
- Kemudian untuk Komposisi: merupakan proses terakhir yang melibatkan penggabungan berbagai fitur menjadi satu fitur untuk mendapatkan data yang lebih akurat atau bermakna.

Cara diatas dapat membantu dalam penambahan data training yang akan dilakukan.

- Contoh / Ilustrasi Gambar : 5.93, 5.94 dan 5.95.

```
In [5]: for dirname in ["train/pos", "train/neg", "train/unsup", "test/pos", "test/neg"]:
...:     for fname in sorted(os.listdir("aclImdb/" + dirname)):
...:         if fname.endswith('.txt'):
...:             with open('aclImdb/' + dirname + '/' + fname, encoding='UTF-8') as f1:
...:                 sent = f1.read()
...:                 words = extract.words(sent)
...:                 unsup_sentences.append(TaggedDocument(words, [dirname + '/' + fname]))
```

Figure 5.93: 4-model-1- fadila

Penjelasan (Maksud Dari Gambar Dan Code Secara Keseluruhan) :

Berdasarkan gambar diatas, direalisasikan pemanggilan directory name (dirname) dari file yang akan dieksekusi . Kemudian didefinisikan fname

```
In [4]: for dirname in ["text_sentiment/pos", "text_sentiment/neg"]:
    ...:
        for fname in sorted(os.listdir(dirname)):
    ...:
        if fname[-4:] == '.txt':
    ...:
            with open(dirname + '/' + fname, encoding='utf-8') as f:
    ...:
                for l, sent in enumerate(f):
    ...:
                    words = extract_words(sent)
    ...:
                    unsup_sentences.append(TaggedDocument(words, [
    ...:
                    ["text-3d"] * (dirname, fname, l)]))
```

Figure 5.94: 4-model-2- fadila

```
In [5]: with open("europarlSentiment/freebank/original_en_3d_snippets.txt",
    encoding='utf-8') as f1:
    ...:
        for l, line in enumerate(f1):
    ...:
            words = extract_words(line)
    ...:
            unsup_sentences.append(TaggedDocument(words, [
    ...:
                    ["text-3d"] * l]))
```

Figure 5.95: 4-model-3- fadila

untuk pemberian nama terhadap file tersebut yang akan di urutkan sesuai dengan list dir dengan parameter dirnamanya. Setiap contoh yang diberikan semuanya akan direalisasikan dalam inputan variabel words dengan extract_word yang dihubungkan dengan unsup_sentences yang mengeksekusi class tagged document. Ketiga gambar tersebut ketika dijalankan tidak terjadi error, maka pengujian atau praktekpun berhasil dilakukan.

5. Pengocokan Dan Pembersihan Data

- Pengocokan Data : Melakukan pengacakan terhadap data kemudian nantinya bisa dieksekusi. Kemudian pada setiap eksekusinya akan menghasilkan hasil yang berbeda berdasarkan/berkaitan dengan penerapan mode (shuffle atau random) dalam pengeksekusian data.
- Pembersihan Data : Melakukan pengecekan dan pemulihan data.
- Contoh / Ilustrasi Gambar : 5.96,

```
In [68]: import re
...:
...: def extract_words(sent):
...:     sent = sent.lower()
...:     sent = re.sub(r'<[^>]+>', ' ', sent) #hapus tag html
...:     sent = re.sub(r'(\w)\.(\w)', ' ', sent) #hapus petik satu
...:     sent = re.sub(r'(\w)\,(\w)', ' ', sent) #hapus tanda baca
...:     sent = re.sub(r'(\w)\.(\w)', ' ', sent) #hapus spasi yang berurutan
...:     return sent.split()
...:
...: import random
...: class PermuteSentences(object):
...:     def __init__(self, sents):
...:         self.sents = sents
...:
...:     def __iter__(self):
...:         shufflesent = list(self.sents)
...:         random.shuffle(shufflesent)
...:         for sent in shufflesent:
...:             yield sent
```

Figure 5.96: 5- Pengocokan Dan Pembersihan Data- fadila

Penjelasan (Maksud Dari Gambar Dan Code Secara Keseluruhan) :

Terjadi pengimportan Library Re. Selanjutnya dilakukan pembuatan fungsi untuk menghapus tag html dan perkocakan (pengocokan) . Berkaitan dengan penghapusan tersebut, pada variabel ini terdapat kodingan petik satu yang bisa direalisasikan, tanda baca dan spasi yang berurutan pun ada.'

Selanjutnya yaitu melakukan pengacakan model terhadap data unsupervised learning yang ada, kemudian baru dibuatkan (membuat) modelnya setelah dilakukan pengacakan data yang telah ada sebelumnya.

6. Mengapa Model Harus Di Save Dan Temporari Training Harus Dihapus

- Penjelasan : Model harus di save dikarenakan atau difungsikan untuk mencegah ram pada komputer/laptop tidak terjadi malfunction ataupun lemot (loading yang lama). Kemudian untuk mengapa temporari training harus dihapus dimana digunakan untuk mengosongkan memori sehingga terdapat ruang lebih ataupun lapang sehingga tidak terjadi malfunction pada komputer dll.
- Contoh / Ilustrasi Gambar : 5.97 dan 5.98

```
In [71]: model.delete_temporary_training_data(keep_inference=True)
```

Figure 5.97: 6- Temporari Train Hapus-fadila

```
In [73]: model.save('fadilafadiluy.d2v')
2019-03-29 22:25:42,240 : INFO : saving Doc2Vec object under fadilafadiluy.d2v, separately None
2019-03-29 22:25:59,259 : INFO : saved fadilafadiluy.d2v
```

Figure 5.98: 6- Model Disave-fadila

Penjelasan (Maksud Dari Gambar Dan Code Secara Keseluruhan) :

Untuk model gambar pertama diperlihatkan praktek untuk percobaan penghapusan temporary training data dimana apabila parameternya "keep_inference" true maka akan dilakukan penghapusan sesuai dengan perintah yang ada. Kemudian untuk gambar kedua memperlihatkan praktek dari perintah save terhadap sebuah parameter kata yaitu (fadilafadiluy) yang disimpan berupa INFO berbarengan / disertakan dengan tanggal dan waktu rincinya.

7. Maksud Dari Infer_code

- Penjelasan : Difungsikan untuk menyimpulkan vector yang mana berhubungan dengan vector dokumen baru (pada subjek pengeksekusian).
- Contoh / Ilustrasi Gambar : 5.99

Penjelasan (Maksud Dari Gambar Dan Code Secara Keseluruhan) :

Berdasarkan dari hasil tersebut, kalimat yang dieksekusi yaitu " fadila cahyoni lovers muah " dipecah kemudian disimpulkan dimana keluarannya menghasilkan array dengan dtype=float32.

```
In [75]: model1.evec_vector(extract_words['Fadila calydon loves suns'])  
Out[75]:  
array([ 0.4938709, -0.131924,  0.85260587,  0.1793808,  0.23528285,  
       0.12234241, -0.131924, -0.17877583,  0.04386685,  0.05408505,  
       0.03796692,  0.02813409,  0.06147039,  0.11558057,  0.1516677,  
       0.05212909,  0.01672004,  0.27326147,  0.03234414,  0.0659543,  
       0.06587683,  0.01830678,  0.02831406, -0.14223233,  0.0799877,  
       0.15269593,  0.0757742,  0.08705599,  0.01494568,  0.16526285,  
       0.08000001,  0.02732061,  0.05527454,  0.18512045,  0.1571387,  
       0.11222389,  0.07320621,  0.05527454,  0.18512045,  0.1571387,  
       0.02869939,  0.01678139, -0.26752719,  0.03290815,  0.20921196,  
       0.03291997,  0.03441899], dtype='float32')
```

Figure 5.99: 7-infercode-fadila

8. Maksud Dari Cosine_similarity

- Penjelasan : Cosine_Similarity ini digunakan untuk menghitung tingkat kesamaan maupun kemiripan antar dua buah objek ataupun kata yang dieksekusi sehingga menghasilkan sebuah keluaran.
 - Contoh / Ilustrasi Gambar : 5.100

```
In [78]: from sklearn.metrics.pairwise import cosine_similarity
... cosine_similarity(
...     [model.infer_vector(extract_words("Services sucks2"))],
...     [model.infer_vector(extract_words("Services sucks2.2"))])
Out[78]: 0.5111111111111111
```

Penjelasan (Maksud Dari Gambar Dan Code Secara Keseluruhan) :

Berdasarkan dari hasil gambar diatas, pengeksekusian dan juga pengujian terhadap model cosine_similarity dengan 2 objek kata yaitu "services sucks 2" memberikan keluaran berupa array sebagai tingkatan kesamaan ataupun perbandingan terhadap kedua kata yang sama tersebut. Hasilnya 0.9 sehingga membuktikan tingkat kesamaan yang signifikan diantara keduanya.

9. Praktek Score Dari Cross Validation

- Penjelasan : Untuk Praktek dari score cross validation ini akan berpatokan pada perhitungan model clrf, sentvecs, setiments kemudian akan meghitung keakuratan dari data ataupun parameter yang dieksekusi tersebut.
 - Contoh / Ilustrasi Gambar : 5.101

Figure 5.101: 9-score-fadila

Penjelasan (Maksud Dari Gambar Dan Code Secara Keseluruhan) :

Berdasarkan dari gambar tersebut, telah dilakukan pengeksekusian untuk pengujian score dari cross validation dengan perhitungan model clf

sentvecs, setiments sebagai inputan kemudian menghasilkan keluaran (output) yang berupa angka dimana akan tersebut diartikan sebagai tingkat keakuratan perhitungan yang terjadi pada inputan yang ada.

10. Penanganan Error

(a) Code Error 1:

- Code Error 1:

```
for dirname in [” review_polarity / txt_sentoken / pos ”, ” review_p
```

- Gambar Error :

```
...:           for i, sent in enumerate(f):
...:               words = extract_words(sent)
...:               untag_sentences.append(TaggedDocument(words, [”%s/%s-Nd” %
...: (dirname, fname, i)]))
...:
...: Traceback (most recent call last):
...: File “c:\python\input-5-cab1695599f”, line 2, in <module>
...:     for fname in sorted(os.listdir(dirname)):
...: FileNotFoundError: [Errno 2] The system cannot find the path specified:
...: ‘review_polarity / txt_sentoken / pos’
```

Figure 5.102: error-error-fadila

- Code Dan Cara Pemecahan Error :

- Pada error tersebut dapat dipastikan bahwa directory file yang ingin dipanggil salah ataupun sesuai (spesifik)
- Pastikan apabila ingin menggunakan perintah directory dalam pemanggilannya maka harus benar dan sesuai
- Namun disarankan untuk melakukan eksekusi dari file codingannya berada dalam satu folder sehingga lebih mudah dan efisien.
- Silahkan menyesuaikan codingan sebelumnya dengan codingan dibawah ini untuk mengatasi error

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

- Setelah menganti codingan sebelumnya dengan code baru tersebut maka dipastikan tidak akan terjadi kesalahan atau error yang sama.

(b) Code Error 2 :

- Code Error 2 :

```
NameError: name ‘os’ is not defined
```

- Gambar Error :

- Code Dan Cara Pemecahan Error :

```

In [6]: for dirname in ["train/pos", "train/neg", "train/unsup", "test/pos", "test/neg"]:
...:     for fname in sorted(os.listdir("aclImdb/" + dirname)):
...:         if fname[-4:] == '.txt':
...:             with open("aclImdb/" + dirname + "/" + fname, encoding="UTF-8") as f:
...:                 sent = f.read()
...:                 words = extract_words(sent)
...:                 unsup_sentences.append(taggedDocument(words, [dirname + "/" + fname]))
...:
NameError: name 'os' is not defined

```

Figure 5.103: error-error-2-fadila

- Pada error tersebut dapat dipastikan bahwa os tidak terdefinisikan
- Pada pemrosesan codenya harus di dahului dengan mendefinisikan os itu sendiri
- Silahkan masukkan code berikut, kemudian jalankan.

```

import re
import os
unsup_sentences = []

```

- Setelah menjalankan code tersebut silahkan jalankan kembali code yang telah dicoba sebelumnya
- Maka error tidak akan terjadi lagi.

Chapter 6

MFCC dan Neutral Network

Untuk praktikum saat ini menggunakan buku Python Artificial Intelligence Projects for Beginners. Dengan praktek menggunakan python 3 dan editor spyder dan library python keras dan librosa.

6.1 Rahmi Roza-1164085

6.1.1 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

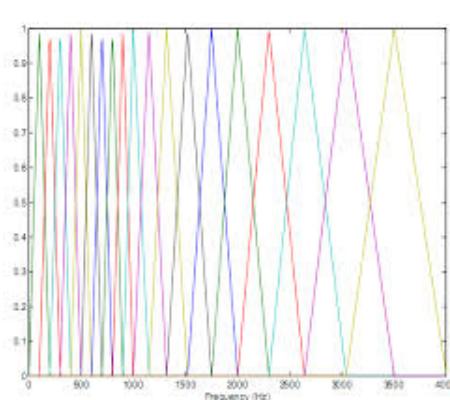


Figure 6.1: MFCC Roza

2. Konsep Dasar Neural Network

- Penjelasan:

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

- Ilustrasi Gambar

NEURAL NETWORK MAPPING

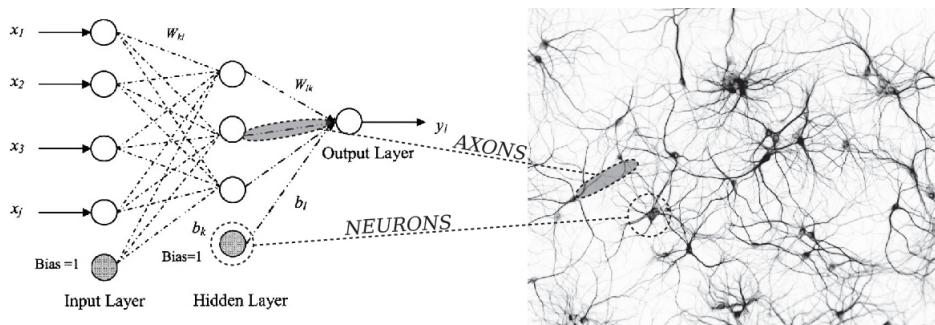


Figure 6.2: Konsep Dasar Neural Network Roza

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.

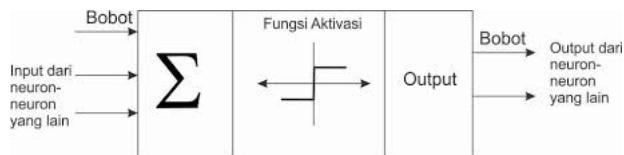


Figure 6.3: Konsep Pembobotan Roza

- Ilustrasi Gambar

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

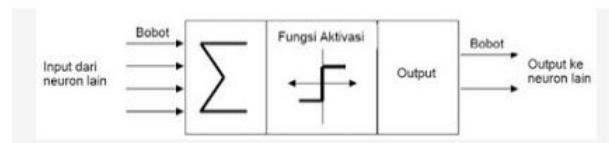


Figure 6.4: Konsep Aktivasi Roza

5. Cara Membaca Hasil Plot Dari MFCC

- Penjelasan:

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

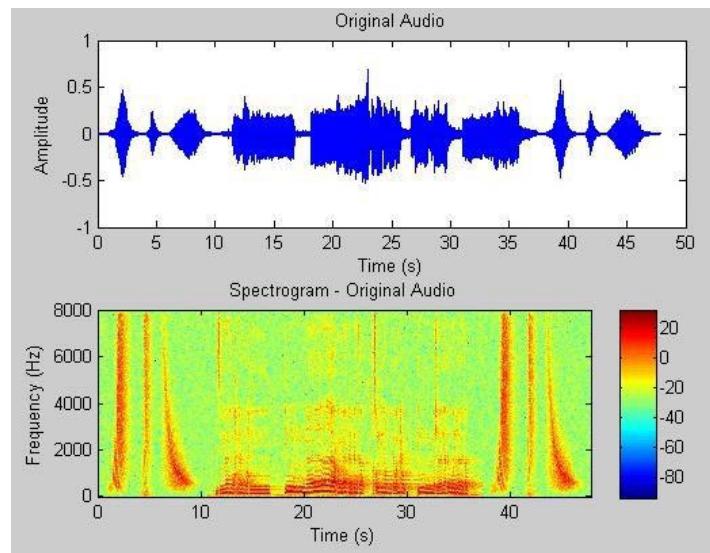


Figure 6.5: Cara Membaca Hasil Plot MLCC Roza

- Ilustrasi Gambar

6. Apa Itu One-Hot Encoding?

- Penjelasan:

One-hot encoding adalah representasi dari variabel kategori sebagai vektor biner. Yang pertama adalah mengharuskan nilai kategorikal 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	Red	Yellow	Green
Red	1	0	0
Red	1	0	0
Yellow	0	1	0
Green	0	0	1
Yellow	0	0	1

Figure 6.6: One Hot Encoding Roza

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

- Np.unique
- Penjelasan: 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])
>>> np.unique(a)
array([ 1.,  2.,  3.,  4.,  5.])
```

Figure 6.7: NP Unique Roza

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

```
import keras
from keras.models import Sequential
from keras.layers import Dense, Dropout, Activation
from keras.optimizers import SGD

# Generate dummy data
import numpy as np
x_train = data.data
y_train = keras.utils.to_categorical(data.target, num_classes=3)
x_test = data.data
y_test = keras.utils.to_categorical(data.target, num_classes=3)
```

Figure 6.8: To Categorical roza

8. Fungsi Dari Sequential Pada Kode Program

- Penjelasan:

Sequential proses membandingkan setiap elemen larik satu per satu secara beruntun, mulai dari elemen pertama, sampai dengan elemen terakhir atau elemen yang dicari sudah ditemukan. atau merupakan jenis model yang digunakan dalam perhitungan ataupun code program yang direalisasikan.

- Ilustrasi Gambar
- Plagiarisme Roza

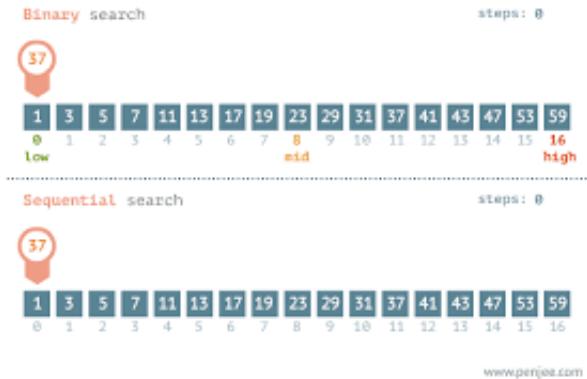


Figure 6.9: Sequential Roza Roza

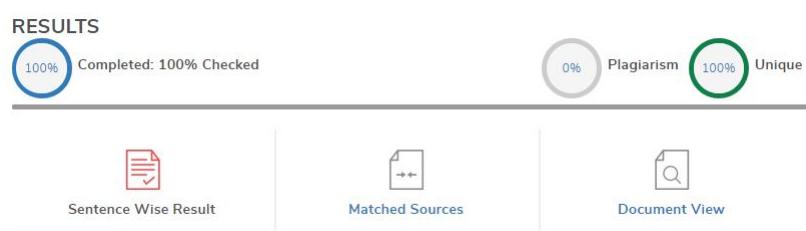


Figure 6.10: Plagiarisme Roza

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.

```
filename_jazz = 'genres/jazz/jazz.00000.au'
x_jazz, sr_jazz = librosa.load(filename_jazz, duration=10)
```

Hasil 6.11 :

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

Figure 6.11: Hasil No 1 Roza

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

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

2. Menjelaskan Perbaris Kode Fungsi Dari display mfcc()

- Kode Program

```
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()
display_mfcc('genres/classical/classical.00067.au')
```

Kode Program 26.12 :

```
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 6.12: Kode Program No 2 Roza

- Hasil Plot

Apabila Sudah Di Plot 6.13 :

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

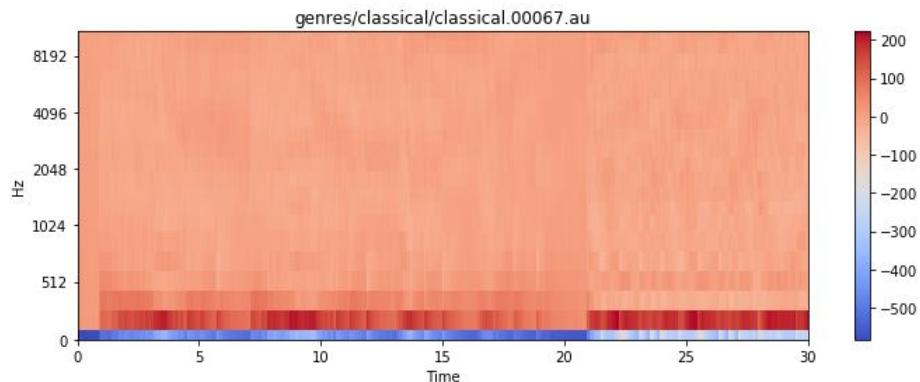


Figure 6.13: Hasil No 2 Roza

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 kodangan ini berwarna atau pada grafiknya

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

```
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]
```

Hasil 36.14 :

```
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 6.14: Hasil No 3 Roza

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

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

```
def generate_features_and_labels():
    all_features = []
    all_labels = []

    genres = ['blues', 'classical', 'country', 'disco', 'hiphop']
    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)
```

Hasil 46.15 :

```
In [11]: def generate_features_and_labels():
...:     all_features = []
...:     all_labels = []
...:
...:     genres = ['blues', 'classical', 'country', 'disco', 'hiphop', 'jazz', 'metal', 'pop',
...: 'reggae', 'rock']
...:     for genre in genres:
...:         sound_files = glob.glob('genres/' + genre + '/*.au')
...:         print('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 6.15: Hasil No 4 Roza

Baris 1: Pembuatan perintah untuk fungsi generate features 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.
`features, labels = generate_features_and_labels()`

Hasil 6.16 :

- Penjelasan Hasil:

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

```
In [9]: features, labels = generate_features_and_labels()
Processing 100 songs in blues genre...
Processing 100 songs in classical genre...
Processing 100 songs in country genre...
Processing 100 songs in disco genre...
Processing 100 songs in hiphop genre...
Processing 100 songs in jazz genre...
Processing 100 songs in metal genre...
Processing 100 songs in pop genre...
Processing 100 songs in reggae genre...
Processing 100 songs in rock genre...
```

Figure 6.16: Hasil No 5 Roza

- 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. Kenapa Harus Dilakukan Pemisahan Data Training dan Data Set 80 persen?

- Agar saat dilakukan pengocokan tidak teracak dalam urutan yang berbeda. Diperlukan Pemisahan sebesar 80% data dikarenakan untuk kemudahan dalam melakukan pengacakannya 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 pengacakannya yang dilakukan datanya akan berada pada urutan yang berbeda.

```
# In [3]: fitur ekstraksi
training_split = 0.8
# In [3]: fitur ekstraksi
# last column has genre, turn it into unique ids
alldata = np.column_stack((features, labels))
# In [3]: fitur ekstraksi
np.random.shuffle(alldata)
splitidx = int(len(alldata) * training_split)
train, test = alldata[:splitidx, :], alldata[splitidx:, :]
```

```
# In [3]: fitur ekstraksi
print(np.shape(train))
print(np.shape(test))
# In [3]: fitur ekstraksi
```

Hasil 6.17 :

```
In [7]: training_split = 0.8
In [8]: alldata = np.column_stack((features, labels))
In [9]: np.random.shuffle(alldata)
...: splitidx = int(len(alldata) * training_split)
...: train, test = alldata[:splitidx,:], alldata[splitidx:,:]
In [10]: print(np.shape(train))
...: print(np.shape(test))
(800, 25010)
(200, 25010)
In [11]: train_input = train[:, :-10]
...: train_labels = train[:, -10:]
In [12]: test_input = test[:, :-10]
...: test_labels = test[:, -10:]
In [13]: print(np.shape(train_input))
...: print(np.shape(train_labels))
(800, 25000)
(800, 10)
```

Figure 6.17: Hasil No 6 Roza

- 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

7. Menjelaskan Masing-Masing Parameter dari Fungsi Sequential.

- Kodingan:

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

Hasil 6.18 :

```

In [14]: model = Sequential([
...:     Dense(100, input_dim=np.shape(train_input)[1]),
...:     Activation('relu'),
...:     Dense(10),
...:     Activation('softmax'),
...: ])
WARNING:tensorflow:From E:\Anaconda3\lib\site-packages\tensorflow\python\framework
\op_def_library.py:263: colocate_with (from tensorflow.python.framework.ops) is deprecated and
will be removed in a future version.
Instructions for updating:
Colocations handled automatically by placer.

```

Figure 6.18: Hasil No 7 Roza

- Penjelasan Hasil:

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 keluarnya menggunakan aktivasi yaitu fungsi Softmax.

8. Menjelaskan Masing-Masing Parameter dari Fungsi Compile.

- Kodingan:

```

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

```

Hasil 6.19 :

```

In [15]: model.compile(optimizer='adam',
...:                     loss='categorical_crossentropy',
...:                     metrics=['accuracy'])
...: print(model.summary())

```

Layer (type)	Output Shape	Param #
dense_1 (Dense)	(None, 100)	2500100
activation_1 (Activation)	(None, 100)	0
dense_2 (Dense)	(None, 10)	1010
activation_2 (Activation)	(None, 10)	0

```

Total params: 2,501,110
Trainable params: 2,501,110
Non-trainable params: 0

```

Figure 6.19: Hasil No 8 Roza

- Penjelasan Hasil:

Pada gambar di atas dapat dilihat bahwa untuk model.compilenya dilakukan pemrosesan menggunakan algortima adam sebagai optimizer yang sudah didefinisikan. Kemudian adam tersebut merupakan algoritme pen-goptimalan 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.

9. Menjalankan Masing-Masing Parameter dari Fungsi Fit

- Kodingan:

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

Hasil 6.20 :

```
In [34]: model.fit(train_input, train_labels, epochs=10, batch_size=32,
...:           validation_split=0.2)
Train on 640 samples, validate on 160 samples
Epoch 1/10
640/640 [=====] - 4s 7ms/step - loss: 2.0536 - acc: 0.2750 - val_loss:
1.9664 - val_acc: 0.3125
Epoch 2/10
640/640 [=====] - 2s 3ms/step - loss: 1.4084 - acc: 0.5094 - val_loss:
1.5388 - val_acc: 0.4375
Epoch 3/10
640/640 [=====] - 2s 3ms/step - loss: 1.0708 - acc: 0.6594 - val_loss:
1.5688 - val_acc: 0.4750
Epoch 4/10
640/640 [=====] - 2s 3ms/step - loss: 0.8133 - acc: 0.7734 - val_loss:
1.4032 - val_acc: 0.4938
Epoch 5/10
640/640 [=====] - 2s 3ms/step - loss: 0.6116 - acc: 0.8687 - val_loss:
1.4091 - val_acc: 0.4813
Epoch 6/10
640/640 [=====] - 2s 3ms/step - loss: 0.4764 - acc: 0.9187 - val_loss:
1.4386 - val_acc: 0.4875
Epoch 7/10
640/640 [=====] - 2s 3ms/step - loss: 0.3743 - acc: 0.9438 - val_loss:
1.4281 - val_acc: 0.4938
Epoch 8/10
640/640 [=====] - 2s 3ms/step - loss: 0.3034 - acc: 0.9516 - val_loss:
1.5063 - val_acc: 0.4750
Epoch 9/10
640/640 [=====] - 2s 3ms/step - loss: 0.2211 - acc: 0.9828 - val_loss:
```

Figure 6.20: Hasil No 9 Roza

- Penjelasan Hasil:

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

10. Menjalaskan Masing-Masing Parameter dari Fungsi Evaluate

- Kodingan:

```
# In [3]: fitur ekstraksi
loss, acc = model.evaluate(test_input, test_labels, batch_size=32)
# In [3]: fitur ekstraksi
print("Done!")
print(" Loss: %.4f, accuracy: %.4f" % (loss, acc))
```

Hasil 6.21 :

```
In [35]: loss, acc = model.evaluate(test_input, test_labels, batch_size=32)
200/200 [=====] - 0s 1ms/step

In [36]: print("Done!")
...: print("Loss: %.4f, accuracy: %.4f" % (loss, acc))
Done!
Loss: 1.2903, accuracy: 0.5650
```

Figure 6.21: Hasil No 10 Roza

- Penjelasan Hasil:

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

11. Menjalaskan Masing-Masing Parameter dari Fungsi Predict

- Kodingan:

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

Hasil 6.22 :

```
In [37]: model.predict(test_input[:1])
Out[37]:
array([[5.1591680e-03, 8.0156066e-03, 2.5642773e-03, 8.9955945e-05,
       8.1511022e-04, 9.7009897e-01, 7.4521388e-04, 2.1298551e-04,
       1.1472276e-02, 8.2642148e-04]], dtype=float32)
```

Figure 6.22: Hasil No 11 Roza

- Penjelasan Hasil:

Kode tersebut digunakan 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.

6.1.3 Penanganan Error

NameError: name 'librosa' is not defined

(a) Skrinsut Error

```
In [3]: display_mfcc('98195_grrlrighter_whistling.wav')
Traceback (most recent call last):
  File "<ipython-input-3-00307953a0a0>", line 1, in <module>
    display_mfcc('98195_grrlrighter_whistling.wav')
  File "<ipython-input-1-59953e4547ad>", line 2, in display_mfcc
    y, _ = librosa.load(song)
NameError: name 'librosa' is not defined
```

Figure 6.23: Error Roza

(b) Kode Error dan Jenis Errornya

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

Jenis Error: Nama Librosa Tidak terdefinisi

(c) Penanganan

Mendefinisikan library librosa

6.2 Fadila-1164072

6.2.1 Teori

Penjelasan Tugas Harian 11 (No 1-8).

1. Mengapa File Suara Harus Dilakukan MFCC Dilengkapi Dengan Ilustrasi Atau Gambar :

Perlu diketahui terlebih dahulu bahwa MFCC (Mel Frequency Cepstral Coefficients) merupakan koefisien yang merepresentasikan audio (lebih simpelnya digunakan sebagai library). Ekstraksi ciri dalam proses ini ditandai dengan pengubahan data suara menjadi data citra berupa spektrum gelombang.

- Penjelasan File Suara Harus Dilakukan MFCC :

Diharuskan melakukan MFCC kepada objek suara agar suara dapat berubah / diubah ke dalam bentuk data matrix dimana telah dilakukan ekstraksi oleh MFCC (data citra berupa spektrum gelombang) kemudian direalisasikan sebagai data matrix. Suara tersebut menjadi vektor yang nantinya akan diolah jadi outputan.

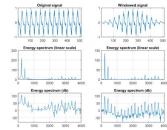


Figure 6.24: Suara Harus Dilakukan MFCC - fadila

- Ilustrasi Gambar : 6.24

Penjelasan :

Pada gambar tersebut, digambarkan sebuah bingkai dari klip suara bernyanyi untuk pengujian yang sama. Dengan menggunakan jendela Hamming, harmonik dalam respons frekuensi jauh lebih tajam. Untuk bingkai input terdiri dari 3 periode fundamental yang identik, maka respons frekuensi magnitudo akan dimasukkan 2 nol antara setiap dua titik tetangga dari respons frekuensi dari periode fundamental tunggal. Dengan kata lain, harmonik dari respons frekuensi umumnya disebabkan oleh periode fundamental berulang dalam bingkai.

2. Konsep Dasar Neural Network Dilengkapi Dengan Ilustrasi Atau Gambar :

Neural Network merupakan sistem komputasi yang efisien yang tema utamanya dipinjam dari analogi jaringan saraf biologis. JST juga disebut sebagai "sistem saraf tiruan," atau "sistem pemrosesan terdistribusi paralel," atau "sistem koneksiis."

- Penjelasan Konsep Dasar Neural Network : (Proses Kerja Saraf Pada Manusia)

Ide dasar Neural Network dimulai dari otak manusia atau terkenal yang diawali dengan karya fisiolog, dimana otak memuat sekitar 10^{11} neuron. Untuk setiap dari sel yang dihasilkan akan saling berinteraksi satu sama lain yang menghasilkan kemampuan tertentu pada kerja otak manusia.

- Ilustrasi Gambar : 6.25.

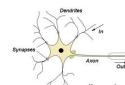


Figure 6.25: Konsep Dasar Neural Network- fadila

Penjelasan :

Gambar tersebut mendefinisikan beberapa hal berikut :

- (a) Untuk Dendrit (Dendrites) berfungsi untuk mengirimkan impuls yang diterima ke badan sel syaraf.
- (b) Untuk Akson (Axon) berfungsi untuk mengirimkan impuls dari badan sel ke jaringan lain
- (c) Untuk Sinapsis berfungsi sebagai unit fungsional di antara dua sel syaraf.

Secara keseluruhan dapat dijelaskan bahwa sebuah neuron menerima impuls dari neuron lain melalui dendrit dan mengirimkan sinyal yang dihasilkan oleh badan sel melalui akson. Akson dari sel syaraf ini bercabang-cabang dan berhubungan dengan dendrit dari sel syaraf lain dengan cara mengirimkan impuls melalui sinapsis.

3. Konsep Pembobotan Neural Network Dilengkapi Dengan Ilustrasi Atau Gambar :

- Penjelasan Konsep Pembobotan Neural Network :

Sebuah Neural Network dikonfigurasi untuk aplikasi tertentu, seperti pengenalan pola atau klasifikasi data. Terjadi penglibatan dalam penyesuaian koneksi sinaptik yang ada antara neuron ketika melakukan penyempurnaan dengan proses pembelajaran. Penyesuaian nilai bobot yang ada pada tiap koneksi baik dari input, neuron maupun output disinkronkan dengan penyesuaian koneksi sinaptik antar neuron itu sendiri.

- Ilustrasi Gambar : 6.26.

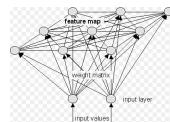


Figure 6.26: Pembobotan Pada Neural Network- fadila

4. Konsep Fungsi Aktifasi Dalam Neural Network Dilengkapi Dengan Ilustrasi Atau Gambar :

- Penjelasan Konsep Fungsi Aktifasi Dalam Neural Network :

Operasi matematik yang dikenakan pada sinyal output y . Fungsi ini akan digunakan untuk pengaktifan dan juga penonaktifan neuron. Perilaku dari JST (jaringan saraf tiruan) ditentukan oleh bobot beserta dengan masukan-keluaran fungsi aktivasi yang ditetapkan.

- 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)
 - Fungsi Linear (Nilai masukan dan keluaran sama)
 - Fungsi Saturating Linear (Keluarannya bernilai satu apabila masukkanya lebih dari 0)

- Ilustrasi Gambar : 6.27.

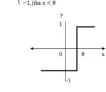


Figure 6.27: Fungsi Aktifasi - fadila

Penjelasan : (Contoh Fungsi Aktivasi Neural Network : Bipolar Threshold)

Pada gambar tersebut menjelaskan bahwa fungsi ini mempunyai keluaran atau outputan yang bernilai 1,0 ataupun -1 untuk batas nilai ambang 0 tertentu. Secara sistematis, fungsi tersebut dapat digambarkan seperti contoh yang telah diberikan.

5. Cara Membaca Hasil Plot Dari MFCC Dilengkapi Dengan Ilustrasi Atau Gambar :

- Pembacaan Hasil Plot Dari MFCC : (berdasarkan contoh)

Untuk hasil plotting dari MFCC, perhitungan pada prosesnya melibatkan atau mendefinisikan pemrosesan sinyal dimana terdapat 430 sampel yang tumpang tindih dengan 215 sampel (kesetaraan jendela 50 ms) kemudian di terapkan jendela Hamming ke sebuah segmen (Hitung FFT: $X = \text{fft}(x)$). Perhitungan energi tidak termasuk pada bagian frekuensi negatif oleh

karena hanya diambil frekuensi antara 0-4kHz. Kemudian di dapatlah bank 40 filter berbentuk segitiga dengan pusat tersebar di frekuensi mel antara 20Hz dan 4kHz.

- Ilustrasi Gambar : 6.28 dan 6.29.



Figure 6.28: Pembacaan Hasil Plot MFCC- fadila

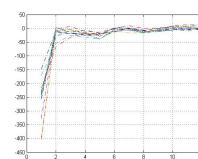


Figure 6.29: Pembacaan Hasil Plot MFCC- fadila

6. Apa itu One-Hot Encoding Dilengkapi Dengan Ilustrasi Atau Gambar :

- Penjelasan One-Hot Encoding :

Proses di mana variabel kategorikal dikonversi menjadi bentuk yang dapat disediakan untuk algoritma ML untuk melakukan pekerjaan yang lebih baik dalam prediksi.

- Ilustrasi Code Dan Gambar :

- (a) Ilustrasi 1 : 6.30 dan 6.31.



Figure 6.30: One-Hot Encoding 1- fadila

	Vid	Acura	Honda	Price
1	1	0	0	28000
2	0	1	0	18011
3	0	0	1	58000
4	0	0	1	18000

Figure 6.31: One-Hot Encoding 1- fadila

Penjelasan :

Nilai kategoris mewakili nilai numerik dari entri dalam dataset. Dicon- tohkan apabila ada perusahaan lain dalam dataset, itu akan diberi nilai

kategoris sebagai 4. Ketika jumlah entri unik meningkat, nilai kategoris juga meningkat secara proporsional. Tabel sebelumnya hanyalah representasi. Pada kenyataannya, nilai-nilai kategorikal mulai dari 0 sampai dengan kategori N-1.

Inimerupakan bentuk organisasi yang didefinisikan dengan VW & Acura & Honda berdasarkan pada nilai-nilai kategorikal. Rata-rata VW dan Honda adalah Acura, dengan menggunakan satu one-hot encoding untuk melakukan "binarisasi" kategori dan memasukkannya sebagai fitur untuk melatih model sehingga memberikan hasil yang sesuai.

(b) Ilustrasi 2 : 6.32.

```
from numpy import array
from numpy import argmax
from keras.utils import to_categorical
# define example
data = [1, 3, 2, 0, 3, 2, 2, 1, 0, 1]
data = array(data)
print(data)
# one hot encode
encoded = to_categorical(data)
print(encoded)
# invert encoding
inverted = argmax(encoded[0])
print(inverted)
```

1	3	2	0	3	2	2	1	0	1
1	0	1	0	0	1	0	0	1	0
1	0	0	1	0	0	1	0	0	1
1	0	0	0	1	0	0	1	0	0
1	0	0	0	0	1	0	0	0	1
1	0	0	0	0	0	1	0	0	0
1	0	0	0	0	0	0	1	0	0
1	0	0	0	0	0	0	0	1	0
1	0	0	0	0	0	0	0	0	1
1	0	0	0	0	0	0	0	0	0

Figure 6.32: One-Hot Encoding 2 - fadila

Penjelasan :

Pada gambar tersebut dijelaskan bahwa terdapat bilangan bulat yang kemudian dikodekan sebagai vektor biner dan dicetak. Kemudian dapat dilihat bahwa nilai integer pertama 1 dikodekan sebagai [0, 1, 0, 0]. Selanjutnya membalikkan pengkodean dengan menggunakan fungsi argumax NumPy () pada nilai pertama dalam urutan yang mengbalikkan nilai yang diharapkan 1 untuk bilangan bulat pertama.

7. Fungsi Dari Unp.Unique Beserta To.Categorical Dilengkapi Dengan Ilustrasi Atau Gambar :

(a) Penjelasan Fungsi Dari Unp.Unique :

Berfungsi untuk menemukan elemen unik array. Dimana dapat , mengembalikan elemen unik array yang diurutkan. 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

Ilustrasi Gambar : 6.33.

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

Figure 6.33: Np.Unique - fadila

Penjelasan :

Pada gambar tersebut, secara garis besarnya np.unique mengambil data yang berbeda dari variabel a yang berada dalam array modul np itu sendiri. Pada contoh terdapat nilai 1 yang memiliki pengulangan angka, maka hanya salah satu diantara angka tersebut yang ditampilkan bukan keduaanya yang diikuti dengan angka berbeda lainnya. Hasilnya yang telah digambarkan pada contoh.

(b) Penjelasan Fungsi Dari To_Categorical :

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

- Ilustrasi Gambar : 6.34.

```
> labels
array([0, 2, 1, 2, 0])
> to_categorical(labels)
array([[ 1.,  0.,  0.],
       [ 0.,  0.,  1.],
       [ 0.,  1.,  0.],
       [ 0.,  0.,  1.],
       [ 0.,  0.,  0.1]], dtype=float32)
```

Figure 6.34: To Categorical - fadila

Penjelasan :

Pada gambar berikut di contoh penerapan array dari 5 label yang telah diset menjadi 3 kelas. Kemudian `to_categorical` difungsikan untuk mengkonversi array tersebut kedalam matrix sebanyak dari kolom-kolom yang ada pada kelas tersebut. Jumlah dari barisnya akan tetap sama dengan yang telah dieksekusi sebelumnya. Dan jadilah hasil seperti contoh tersebut.

8. Fungsi Dari Sequential Dilengkapi Dengan Ilustrasi Atau Gambar :

- Penjelasan Fungsi Dari Sequential :

Sebuah jenis model yang digunakan dalam perhitungan ataupun code program yang direalisasikan. Neural Networks Sequential membangun fitur tingkat tinggi melalui lapisannya yang berurutan. Sequential juga merupakan proses dimana 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 : 6.35.



Figure 6.35: Sequential - fadila

Penjelasan :

Pada gambar diatas, mendefinisikan pencarian terhadap angka 47 dimana hasilnya tetap diurutkan sesuai dengan elemennya.

9. Plagiarism :

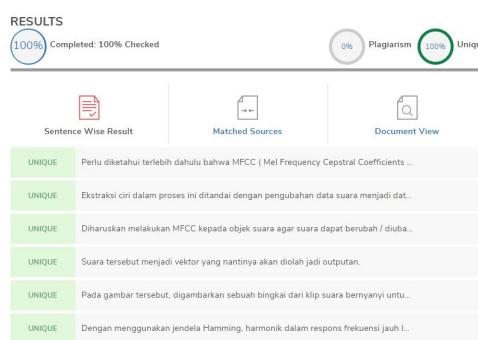


Figure 6.36: Plagiarisme- fadila

6.2.2 Praktek

1. Penjelasan Isi Data GTZAN Genre Collection Dan Data Dari Freesound.

- Penjelasan Isi Data GTZAN :

Pada GTZAN Genre Collection itu isinya berupa data musik yang sudah di folderkan berdasarkan genre lagunya (dikelompokkan).

- Code Yang Digunakan :

```
filename_reggae = 'genres/reggae/reggae.00000.au'
x_reggae, sr_reggae = librosa.load(filename_reggae, duration=10)
```

- Ilustrasi Gambar (Contoh) : 6.37

```
In [8]: filename_reggae = 'genres/reggae/reggae.00000.au'
...: x_reggae, sr_reggae = librosa.load(filename_reggae, duration=10)
```

Figure 6.37: Contoh GTZAN Genre Collection - fadila

- Penjelasan Contoh Code :

- Baris Code 1 : Filenam reggae merupakan variabel yang berisikan direktori dari file yang dituju, pada code ini digunakan file audio dari genre reggae.
- Baris Code 2 : Membuat variabel X reggae dan sr reggae yang digunakan untuk meload file dari variabel filename reggae menggunakan librari Librosa dengan durasi 10, yang nantinya akan digunakan pada Mfcc.

2. Penjelasan Perbaris Kode Program Dari Display_Mfcc.

- Code Yang Digunakan :

```
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()

%%%
display_mfcc('98195_grrlrighter_whistling.wav')
```

- Penjelasan :

- Baris Code 1 : Membuat fungsi display mfcc untuk menampilkan vektorisasi dari sebuah suara dimana variabel parameter yaitu song
- Baris Code 2 : Membuat variabel Y dimana untuk melakukan load (meload) atau membaca variable parameter song dari perintah librosa load

- (c) Baris Code 3 : Membuat variabel mfcc untuk memanggil variabel Y (yang telah dibuat sebelumnya) dan mengubah suara menjadi vektor
 - (d) Baris Code 4 : Melakukan plotting gambar dengan ukuran 10x4 dari figsize
 - (e) Baris Code 5 : Menampilkan spektrogram/chromagram dari library librosa dimana untuk x_axis mendefinisikan time kemudian y_axis mendefinisikan mel.
 - (f) Baris Code 6 : Menambahkan colorbar pada plot yang dijalankan
 - (g) Baris Code 7 : Menetapkan atau memberikan judul untuk suara yang dieksekusi
 - (h) Baris Code 8 : Untuk menyesuaikan subplot params (label dll) sehingga subplot cocok dengan area gambar.
 - (i) Baris Code 9 : Fungsi untuk menampilkan hasil plot dari inputan yang telah dieksekusi.
- Code dibawah ini merupakan code lanjutan
- (j) Baris Code 10 : Fungsi Yang Akan Menampilkan Plot Dan Bar Dari Inputan code dan file suara whisling

- Ilustrasi Gambar : Ketika program tersebut dijalankan maka akan memberikan hasil seperti pada gambar berikut 6.38 dan 6.39.

```
In [2]: def display_mfcc(song):
    """
    :param song: str
    :return: None
    """
    # Load song
    y, sr = librosa.load(song)
    # Compute MFCCs
    mfcc = librosa.feature.mfcc(y)
    # Plot MFCCs
    librosa.display.specshow(mfcc, sr=sr, x_axis='time', y_axis='mel')
    plt.colorbar()
    plt.title(song)
    plt.show()
```

Figure 6.38: Display Mfcc 1 - fadila

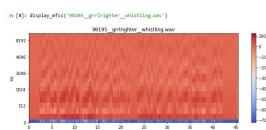


Figure 6.39: Display Mfcc 2 - fadila

3. Penjelasan Perbaris Code Dari Extract_Feature_Song().

- Code Yang Digunakan :

```
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.absolute(np.amax(mfcc))

return np.ndarray.flatten(mfcc)[:25000]

```

- Penjelasan Code :

- Baris Code 1 : Membuat fungsi extract feature song dengan inputan (parameter) f
- Baris Code 2 : Membuat variabel y dimana untuk meload atau membaca inputan (parameter) f dari perintah librosa load song
- Baris Code 3 : Membuat variabel mfcc yang difungsikan untuk membuat feature dari variabel y berdasarkan library librosa
- Baris Code 4 : Membuat normalisasi nilai antara -1 sampai 1 yang didapatkan dari eksekusi np.absolute
- Baris Code 5 : (return = mengembalikan) atau bisa didefinisikan untuk mengambil 25000 data pertama berdasarkan durasi suara atau musik lalu dikembalikan salinan arraynya dan dikecilkan menjadi satu.

- Ilustrasi Gambar : Ketika program tersebut dijalankan maka akan memberikan hasil seperti pada gambar berikut 6.40.

```

In [2]: def extract_features_song(f):
...:     y, sr = librosa.load(f)
...:     mfcc = librosa.feature.mfcc(y)
...:     mfcc /= np.absolute(np.amax(mfcc))
...:     return np.ndarray.flatten(mfcc)[:25000]
...:

```

Figure 6.40: Extract Feature Song - fadila

- Mengapa Yang Diambil Merupakan 25.000 Baris Data Pertama ?

Data yang digunakan hanya 25.000 saja (baris data pertama) sehingga menghindari down pada sistem yang dieksekusi. Apabila diganti dengan 100.00 (sebagai contoh pembuktian) angka pertama dapat dipastikan pemrosesan yang lama terhadap dataset tersebut. Jadi digunakan 25.000 baris data pertama dijadikan sebagai contoh untuk pengeksekusian.

4. Penjelasan Perbaris Code Dari Generate_Features_And_Labels().

- Code Yang Digunakan :

```

def generate_features_and_labels():
    all_features = []
    all_labels = []

    genres = [ 'blues' , 'classical' , 'country' , 'disco' , 'hiphop'
    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 )

```

- Penjelasan Code :

- Baris Code 1 : Membuat perintah untuk fungsi generate features and labels
- Baris Code 2 : Pembuatan variabel all_features dengan array/parameter kosong
- Baris Code 3 : Pembuatan variabel all_labels dengan array/parameter kosong
- Baris Code 4 : Membuat / mendefinisikan variable genres yang di dalamnya berisi nama folder-folder pada variabel genres tersebut
- Baris Code 5 : Membuat perintah fungsi looping (header)
- Baris Code 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 Code 7 : Memunculkan berapa (jumlah) song yang dieksekusi.
- Baris Code 8 : Membuat perintah fungsi dari sound_files
- Baris Code 9 : Membuat variabel features untuk memanggil fungsi extract features song (f) sebagai inputan. Setiap satu file array sound files dilakukan ekstrak fitur.
- Baris Code 10 : Memasukkan semua features menggunakan perintah append kedalam all_features
- Baris Code 11 : Memasukkan semua genres menggunakan perintah append ke dalam all_labels
- Baris Code 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.

- (m) Baris Code 13 : Membuat variabel label_row_ids untuk menentukan type dari variabel tersebut dengan type bit yang sesuai dengan yang digunakan.
 - (n) Baris Code 14 : Membuat variabel onehot_labels dimana mengeksekusi to_categorical dengan variabel parameter low_row_ids dan len(label_uniq_ids)
 - (o) Baris Code 15 : Mengembalikan dan menampilkan hasil eksekusi dari variabel parameter all_features dan onehot_labels perintah dari np.stack.
 - Ilustrasi Gambar : Ketika program tersebut dijalankan maka akan memberikan hasil seperti pada gambar berikut 6.41.

Figure 6.41: Generate_Features_And_Labels - fadila

5. Penjelasan Penggunaan Fungsi `Generate_Features_And_Labels()` Sangat Lama Ketika Meload Dataset Genre.

- Penjelasan :

Fungsi `Generate_Features_And_Labels()` sangat lama ketika meload dataset genre dikarenakan dalam dataset tersebut memiliki 1000 data dimana 1000 data tersebut dibagi menjadi 10 bagian genre. Pada setiap genre tersebut selain memiliki 100 data, atribut-atribut (feature dan label) yang dieksekusi juga terhitung banyak sehingga membutuhkan waktu yang cukup lama agar pemrosesan 1000 data tersebut sesuai dengan perintah yang diberikan.

- Code Yang Digunakan :

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

- Ilustrasi Gambar : 6.42

Penjelasan :

Hasil yang di dapatkan dari gambar yaitu dilakukan pemrosesan load pada data-data di masing-masing genre. Dimana setiap genrenya terdiri dari 100 data yang apabila di akumulatifkan dengan 10 jenis genre musik maka akan ada 1000 data secara keseluruhan.

```
In [4]: features, labels = generate_features_and_labels()
Processing 100 songs in blues genre...
Processing 100 songs in country genre...
Processing 100 songs in folk genre...
Processing 100 songs in hip hop genre...
Processing 100 songs in jazz genre...
Processing 100 songs in metal genre...
Processing 100 songs in pop genre...
Processing 100 songs in rock genre...
```

Figure 6.42: Generate_Features_Labels-2-fadila

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 :

```
# In [3]: fitur ekstraksi
training_split = 0.8
# In [3]: fitur ekstraksi
# last column has genre, turn it into unique ids
alldata = np.column_stack((features, labels))
# In [3]: fitur ekstraksi
np.random.shuffle(alldata)
splitidx = int(len(alldata) * training_split)
train, test = alldata[:splitidx, :], alldata[splitidx:, :]
# In [3]: fitur ekstraksi
print(np.shape(train))
print(np.shape(test))
# In [3]: fitur ekstraksi
```

- Penjelasan Perbaris (Tapi dalam soal tidak diperintahkan namun hanya saya jelaskan) :

- Baris Code 1 : Melakukan Proses Training split dimana akan memisahkan training set sebanyak 80%
- Baris Code 2 : Melakukan penumpukan features dan labels yang didefinisikan dalam variabel all_data
- Baris Code 3 : Melakukan Pengocokan (shuffle) untuk variabel all_data
- 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 : 6.43 dan 6.44

- 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.

```
In [3]: training_ratio = 0.8
In [4]: all_data = np.column_stack((features, labels))
In [5]: np.random.shuffle(all_data)
        ...: training_set, testing_set = np.split(all_data, [int(len(all_data)*training_ratio)])
        ...: train, test = all_data[:splitidx,:], all_data[splitidx:,:]
In [6]: print(np.shape(train))
        ...: print(np.shape(test))
(800, 25000)
(200, 25000)
```

Figure 6.43: Pemisahan Data Training Dan Dataset - fadila

```
In [11]: train_input = train[:, :-1]
In [12]: train_labels = train[:, -1]
In [13]: test_input = test[:, :-1]
        ...: test_labels = test[:, -1]
In [14]: print(np.shape(train_input))
        ...: print(np.shape(test_input))
        ...: print(np.shape(train_labels))
        ...: print(np.shape(test_labels))
(800, 25000)
(200, 25000)
(800, 10)
(200, 10)
```

Figure 6.44: Pemisahan Data Training Dan Dataset 2 - fadila

7. Penjelasan Parameter Dari Fungsi Sequential().

- Code Yang Digunakan :

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

Penjelasan :

- Ilustrasi Gambar : 6.45

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 keluarnnya menggunakan aktivasi yaitu fungsi Softmax.

```
In [14]: model = Sequential([
    ...
    Dense(100, input_dim=8, shape=(train_x_input[1])),
    Activation('relu'),
    ...
    Dense(10, shape=(train_x_input[2])),
    Activation('softmax'),
    ...
])

#WARNING:tensorflow:From c:\users\andrea\appdata\local\temp\pip-req-build-1111\TensorFlow\python\framework\ops.py:235: collocate_with from tensorflow.python.framework.ops is deprecated and will be removed in a future version.

```

Figure 6.45: Fungsi Sequential - fadila

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

- Code Yang Digunakan :

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

- Ilustrasi Gambar : 6.46

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.

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

```
In [55]: model.compile(optimizer='adam',
                      loss='categorical_crossentropy',
                      metrics=['accuracy'])
print(model.summary())

Layer (type)                 Output Shape              Param #   
dense_1 (Dense)              (None, 100)             2500000  
activation_1 (Activation)    (None, 100)             0        
dense_2 (Dense)              (None, 10)              1000      
activation_2 (Activation)    (None, 10)             0        
Total params: 2,500,100
Trainable params: 2,500,100
Non-trainable params: 0

None
```

Figure 6.46: Fungsi Compile - fadila

- Code Yang Digunakan :

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

- Penjelasan :

Berdasarkan gambar tersebut dapat dilihat bahwa pada model fit dilakukan pelatihan dengan epoch(iterasi berapa kali nilai digunakan) dengan rambatan 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 : 6.47

Figure 6.47: Fungsi Fit - fadila

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

- Code Yang Digunakan :

```
loss , acc = model.evaluate(test_input , test_labels , batch_size=100)
# In[3]: fitur ekstraksi
print("Done!")
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 sebesar 0,4200.

- Ilustrasi Gambar : 6.48

```
In [37]: loss, acc = model.evaluate(test_input, test_labels, batch_size=52)
2020-06-10 10:44:11.111252: I tensorflow/core/grappler/optimizers/resource_mgr.cc:104] - 91.2ms/step
In [38]: print("Loss: %f" % loss)
... print("Loss: %f, acc: %f" % (loss, acc))
Loss: 1.7985, accuracy: 0.4200
```

Figure 6.48: Fungsi Evaluate - fadila

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

- Code Yang Digunakan :

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

- Ilustrasi Gambar : 6.49

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 [39]: model.predict(test_input[:1])
Out[39]: array([ 0.00000000,  0.49999992,  0.17985121,  0.16999999,  0.00000000,
   0.00000000,  0.00000000,  0.00000000,  0.00000000,  0.00000000], dtype=float32)
```

Figure 6.49: Fungsi Predict - fadila

6.2.3 Fadila-Penanganan Error Chapter 6

1. Screenshoot Error : 6.50

```
In [51]: display.mfcc('266093_stereo-surgeon_kick-loop-5.wav')
Traceback (most recent call last):
  File "/usr/local/lib/python3.6/dist-packages/IPython/core/displayhook.py", line 14, in __call__
    display_mfcc('266093_stereo-surgeon_kick-loop-5.wav')
  File "/usr/local/lib/python3.6/dist-packages/IPython/core/displayhook.py", line 2, in display_mfcc
    'y', _=librosa.load('266093_stereo-surgeon_kick-loop-5.wav')
NameError: name 'librosa' is not defined
```

Figure 6.50: Error - fadila

2. Code Error :

```
display.mfcc('266093_stereo-surgeon_kick-loop-5.wav')
```

```
##%
```

```
#Ini Tulisan Errornya
NameError: name 'librosa' is not defined
```

3. Penyelesaian Error :

- (a) Pertama-tama silahkan perhatikan maksud errornya seperti apa
- (b) Pada error tersebut dikatakan bahwa "librosa" tidak terdefinisi
- (c) Selanjutnya silahkan definisikan library librosanya sebelum menjalankan perintah tersebut
- (d) Code yang harus dijalankan untuk mendefinisikan library librosa yaitu sebagai berikut :

```
import librosa
import librosa.feature
import librosa.display
```
- (e) Silahkan jalankan code tersebut, lalu jalankan kembali code yang menghasilkan error tadi
- (f) Setelah semua langkah-langkahnya diikuti maka tidak akan terjadi error lagi.

6.3 Lusia Violita Aprilian-1164080

6.3.1 Teori

1. Jelaskan kenapa file suara harus di lakukan MFCC.

- Jawaban :

MFCC (Mel Frequency Cepstral Coeficients) digunakan untuk mengubah file suara kedalam bentuk vektor. Dan mengapa harus dilakukan MFCC? Jawabannya ialah supaya file suara tersebut dapat diolah kembali menjadi output-an. Dan semua parameter inputan baik itu suara maupun dokumen harus dipersiapkan datanya terlebih dahulu.

- Ilustrasi Gambar :

Berikut adalah ilustrasi penggambaran dari MFCC, perhatikan gambar 6.51:

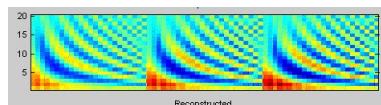


Figure 6.51: Lusia-MFCC

2. Jelaskan konsep dasar neural network.

- Jawaban :

Konsep dasar yang digunakan pada nural network mengadopsi mekanisme berpikir dari struktur dan proses kerja neuron pada otak manusia. Dimana otak manusia dapat bekerja untuk memberikan stimulasi atau rangsangan, melakukan proses, dan memberikan output-an. Baik untuk memproses sebagai sinyal elemen yang diterima, toleransi terhadap kesalahan, maupun pada proses paralel.

- Ilustrasi Gambar :

Berikut adalah ilustrasi penggambaran pada konsep dasar neural network, perhatikan gambar 6.52 :

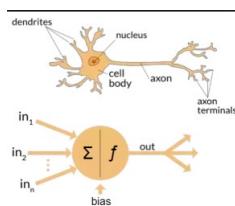


Figure 6.52: Lusia-Neural Network

3. Jelaskan konsep pembobotan dalam neural network.

- (a) Jawaban :

Bobot merupakan jembatan yang menghubungkan neuron satu dengan neuron yang lainnya. Dimana pada proses neural network dimulai dari input yang diterima neuron beserta nilai bobot. Nilai bobot tersebut sebagai penanda dari sebuah koneksi. Selanjutnya nilai input akan dihitung dan menghasilkan aktif atau tidak. Apabila data aktif, neuron mengirimkan nilai output melalui bobot output-nya ke semua neuron yang terhubung.

- (b) Ilustrasi Gambar :

Berikut adalah ilustrasi penggambaran pada konsep pembobotan, perhatikan gambar 6.53 :

4. Jelaskan konsep fungsi aktivasi dalam neural network.

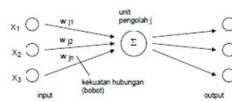


Figure 6.53: Lusia-Konsep Pembobotan

- Jawaban :

Fungsi aktivasi pada neural network berfungsi layaknya sinapsis pada neuron manusia. Dimana pada neural network, fungsi aktivasi sebagai penentu aktivasi dari sebuah nilai input-an yang sebelumnya telah dihitung pada hidden layer.

- Ilustrasi gambar :

Berikut adalah ilustrasi penggambaran pada konsep fungsi aktivasi, perhatikan gambar 6.54 :

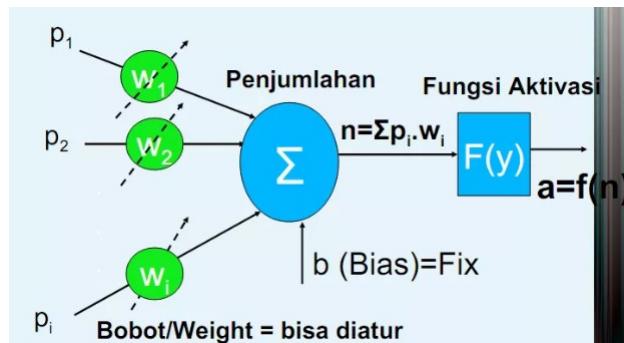


Figure 6.54: Lusia-Konsep Fungsi Aktivasi

5. Jelaskan cara membaca hasil plot dari MFCC.

- Jawaban :

Pada sebuah grafik hasil plot dari MFCC gamabar 6.55, terdapat 2 sumbu yaitu x dan y. Sumbu x merupakan waktu atau durasi dari sebuah suara/-musik/lagu, sedangkan sumbu y merupakan frekuensi dari suara yang dihasilkan, dan hasilnya merupakan desibel/power. Desibel yang dihasilkan memiliki range warna yang berbeda-beda. Range warna dari desibel adalah mulai dari biru tua hingga coklat tua. Range warna biru merupakan suara yang tidak dapat didengar manusia dan coklat merupakan suara yang dapat dapat didengar manusia.

- Ilustrasi gambar :

Berikut adalah ilustrasi dari cara membaca hasil plot dari file whistling.wav :

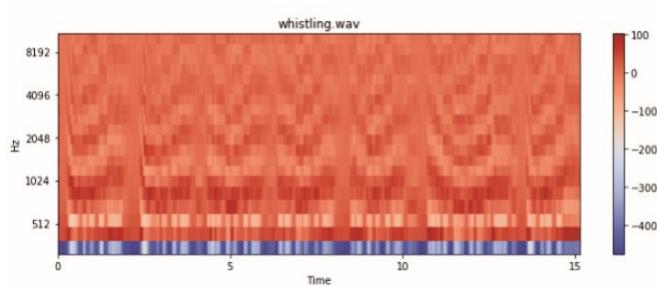


Figure 6.55: Lusia-Membaca hasil plot

6. Jelaskan apa itu one-hot encoding.

- Jawaban :

one-hot encoding adalah proses di mana variabel kategorikal dikonversi menjadi bentuk yang tersedia pada algoritma Machine Learning (vektor biner) untuk melakukan prediksi. Nilai kategorikal akan dipetakan ke bentuk biner, lalu nilai integer diubah atau dipresentasikan kedalam bentuk vektor biner yang semua nilainya bernilai nol kecuali indeks integer.

- Ilustrasi gambar :

Berikut adalah ilustrasi penggambaran one-hot encoding, perhatikan gambar 6.56 :

Label Encoding			One Hot Encoding			
Food Name	Categorical #	Calories	Apple	Chicken	Broccoli	Calories
Apple	1	95	1	0	0	95
Chicken	2	231	0	1	0	231
Broccoli	3	50	0	0	1	50

Figure 6.56: Lusia-one hot encoding

7. Jelaskan apa fungsi dari np.unique dan to_categorical dalam kode program.

- np.unique

np.unique atau numpy.unique yang berfungsi untuk menemukan elemen unik pada array. Perhatikan gambar 6.57 :

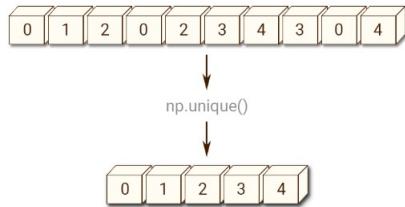


Figure 6.57: Lusia-np.unique

- `to_categorical`

`to_categorical` digunakan untuk meng-convert atau mengubah class vector berupa integer ke dalam matrix kelas biner. perhatikan gambar 6.58 :

```
# 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)
```

Figure 6.58: Lusia-to categorical

8. Jelaskan apa fungsi dari Sequential dalam kode program.

- Jawaban :

Sequential merupakan proses untuk membandingkan setiap satu larik elemen dengan cara satu persatu secara beruntun. Atau pada library keras, sequential merupakan sebuah model.

- Ilustrasi gambar :

Berikut adalah gambaran ilustrasi dari sequential, perhatikan gambar 6.59

```
keras.utils.Sequence()
```

Figure 6.59: Lusia-sequential

6.3.2 Praktek

1. Data GTZAN Genre Collection dan data dari freesound.

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

Berikut adalah kode program untuk meload data tersebut :

```
filename_rock = 'genres/rock/rock.00000.au'
x_rock, sr_rock = librosa.load(filename_rock, duration=10)
```

Berikut penjelasan dari kode program :

- membuat variabel untuk mendefinisikan direktori file yang akan digunakan.
- membuat variabel untuk meload atau memanggil variabel filename_rock.

Apabila kode program dijalankan maka akan menghasilkan seperti gambar 6.60

:

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

Figure 6.60: Lusia-Hasil Load

2. Fungsi dari display_mfcc() .

Berikut adalah kode program yang digunakan :

```
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()
```

Setelah kode program dijalankan, maka akan muncul seperti gambar 6.61 :

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

Figure 6.61: Lusia-Hasil display MFCC

Penjelasan Gambar :

- Baris 1 : membuat fungsi display mfcc untuk menampilkan vektorisasi dari sebuah suara

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

Untuk menampilkan hasil plotting dilakukan perintah berikut :

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

Apabila program tersebut dijalankan, maka akan muncul hasil seperti gambar 6.62 :

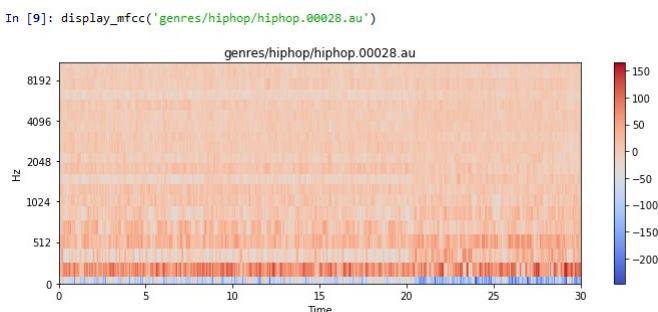


Figure 6.62: Lusia-Hasil display

Penjelasan kode : display MFCC digunakan untuk menampilkan hasil dari gambar 6.61. Dimana gambar tersebut menjelaskan kekuatan atau daya dari sebuah suara.

3. Fungsi dari extract_features_song().

Untuk menggunakan fungsi dari extract_features_song() diimplementasikan pada kode berikut :

```

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]

```

Jika kode di atas dijalankan maka didapat hasil seperti gambar 6.63 :

```

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 6.63: Lusia-Hasil extract feature

Berikut adalah penjelasan dari gambar 6.63:

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

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

4. Penjelasan kode program generate_features_and_labels()

Disini akan mengilustrasikan fungsi dari generate_feature_and_labels(). Kode program yang digunakan adalah seperti berikut :

```

def generate_features_and_labels():
    all_features = []
    all_labels = []

    genres = [ 'blues' , 'classical' , 'country' , 'disco' , 'hiphop' , 'rock' ]
    for genre in genres:

```

```

sound_files = glob.glob('genres/*'+genre+'/*.au')
print('Processing %d songs in %s genre...' % (len(sound_files),
for f in sound_files:
    features = extract_features_song(f)
    all_features.append(features)
    all_labels.append(genre)

```

Apabila kode program tersebut dijalankan akan menghasilkan seperti pada gambar 6.64.

```

In [11]: def generate_features_and_labels():
...:     all_features = []
...:     all_labels = []
...:     genres = ['blues', 'classical', 'country', 'disco', 'hiphop', 'jazz', 'metal', 'pop',
...:              'reggae', 'rock']
...:     for genre in genres:
...:         sound_files = glob.glob('genres/*'+genre+'/*.au')
...:         print('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 with blues = 1000000000, classical = 1100000000
...:     label_uniq_ids, label_row_ids = np.unique(all_labels, return_inverse=True) # the integer
...:     label_row_ids = label_row_ids.astype(np.int32, copy=False)
...:     onehot_labels = np.categorical(label_row_ids, len(label_uniq_ids)) # the one hot
...:     return np.stack(all_features), onehot_labels

```

Figure 6.64: Lusia-Hasil generate fature and label

Berikut adalah penjelasan dari hasil gambar ilustrasi 6.64:

- Baris 1 : membuat perintah fungsi generate features and labels
- Baris 2 : membuat variabel all features dengan array kosong
- Baris 3 : membuat variabel all labels dengan array kosong
- Baris 4 : Membuat variable yang berisi nama folder
- Baris 5 : membuat perintah fungsi looping
- Baris 6 : membuat atribut sound files yang berisi perintah looping perfolder dari folder genres dan mengambil semua file berekstensi au.
- Baris 7 : memunculkan berapa song.
- Baris 8 : membuat perintah fungsi
- Baris 9 : membuat variabel features untuk memanggil fungsi extract features song (f) sebagai inputan. Setiap satu file array sound files dilakukan ekstrak fitur.
- Baris 10 : memasukkan semua features kedalam all features
- Baris 11 : memasukkan semua genres kedalam all labels
- Baris 12 : medefinisikan label uniq ids dan row ids untuk mengeksekusi perintah unix yang memiliki parameter atribut all labels dan return inverse.

- Baris 13 : Membuat variabel label_row_ids untuk menentukan type data 32 byte dari variabel tersebut.
- Baris 14 : Membuat variabel onehot_labels yang akan mengeksekusi to_categorical dengan variabel parameter low_row_ids dan len(label_uniq_ids)
- Baris 15 : Mengembalikan (return) dan menampilkan hasil eksekusi dari variabel parameter all_features dan onehot_labels perintah dari np.stack.

5. penggunaan fungsi generate_features_and_labels()

Mengapa fungsi generate_features_and_labels() sangat lama ketika meload dataset genre? Itu karena data yang diload sangat banyak sehingga membutuhkan waktu yang banyak

Berikut kode program yang digunakan :

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

Dari kode program tersebut dapat dijelaskan bahwa kode tersebut digunakan untuk passing parameter dari fitur ekstraksi menggunakan mfcc. Sehingga ketika dijalankan menghasilkan seperti gambar 6.65.

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

Figure 6.65: Lusia-Hasil generate fature and label

Gambar 6.65 menjelaskan setelah kode dieksekusi, program memproses 100 song setiap genre musik. Sehingga membutuhkan waktu yang lama.

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

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

Berikut adalah kode program yang digunakan :

```
training_split = 0.8
```

Penjelasan kode program :

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

Setelah kode program tersebut dijalankan, maka akan menghasilkan seperti gambar 6.66:

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

Figure 6.66: Lusia-Hasil Pemisahan

7. Fungsi Sequential()

Kode program yang digunakan adalah sebagai berikut :

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

Penjelasan parameter :

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

Setelah program dijalankan, maka akan menghasilkan seperti pada gambar 6.67.

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

Figure 6.67: Lusia-Hasil Sequential

Penjelasan gambar 6.67:

- (a) Membuat model sequential

- (b) Inputan awal neutron adalah 100 yang diambil dari data train.
- (c) Melakukan aktivasi dengan parameter fungsi relu.
- (d) Dense(10) merupakan outputan dari neural network yang mengkategorikan 10 kategori jenis genre.
- (e) Melakukan aktivasi dengan menggunakan parameter fungsi softmax.

8. Fungsi compile()

Kode program yang digunakan adalah sebagai berikut :

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

Penjelasan parameter :

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

Setelah program dijalankan, maka akan menghasilkan seperti pada gambar 6.68.

```
IN [4]: model.compile(optimizer='adam',
                      loss='categorical_crossentropy',
                      metrics=['accuracy'])
print(model.summary())

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

Figure 6.68: Lusia-Hasil Compile

Berikut adalah penjelasan gambar 6.68 :

- (a) Mengcompile model yang sudah dibuat. Dengan menggunakan algoritma optimisasi, fungsi loss, dan fungsi metrik.
- (b) Lalu perintah print digunakan untuk menampilkan hasil yang berupa seperti tabel yang berisi tipe layer, dense, aktivasi, total params dan trainable param.

- (c) Dense_1 dan activation_1 menunjukkan neuron inputan sebanyak 100.
- (d) Dense_2 dan activation_2 menunjukkan neuron outputan sebanyak 10.
9. Fungsi fit()

Kode program yang digunakan adalah sebagai berikut :

```
model.fit(train_input, train_labels, epochs=10, batch_size=32, vali
```

Penjelasan parameter pada kode program :

- epochs = fungsi yang digunakan untuk menentukan berapa kali melakukan iterasi.
- batch_size = fungsi yang digunakan untuk menentukan berapa file dalam satu epochs.
- validation_split = melakukan pengecekan cross validasi.

Apabila kode program tersebut dijalankan, maka akan menghasilkan seperti pada gambar 6.69.

```
In [20]: model.fit(train_input, train_labels, epochs=10,
batch_size=32,
...: validation_split=0.2)
WARNING:tensorflow:From /anaconda3/lib/python3.7/site-packages/
tensorflow/python/ops/math_ops.py:3866: to_int32 (from
tensorflow.python.ops.math_ops) is deprecated and will be
removed in a future version.
Use tf.cast instead.
Use tf.cast instead.
Train on 640 samples, validate on 160 samples
Epoch 1/10
Epoch 1/10 [=====] - 2s 3ms/step - loss: 2.3186 - acc: 0.2562 - val_loss: 1.9249 - val_acc: 0.3250
Epoch 2/10 [=====] - 1s 2ms/step - loss: 1.6474 - acc: 0.3891 - val_loss: 1.6481 - val_acc: 0.4250
Epoch 3/10 [=====] - 1s 2ms/step - loss: 1.3256 - acc: 0.5922 - val_loss: 1.5341 - val_acc: 0.4688
Epoch 4/10 [=====] - 1s 2ms/step - loss: 1.1121 - acc: 0.6484
```

Figure 6.69: Lusia-Hasil Fungsi Fit

Penjelasan hasil, perhatikan gambar 6.69 :

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

Dari gambar 6.69 kita dapat melihat bahwa setiap satu epoch terdapat nilai akurasi dan loss.

10. Fungsi evaluate()

Berikut adalah kode program yang digunakan :

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

Parameter yang digunakan pada kode program :

- batch_size = fungsi yang digunakan untuk menentukan berapa file yang digunakan.

Apabila kode program tersebut dijalankan, maka akan menghasilkan seperti pada gambar 6.70.

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

Figure 6.70: Lusia-Hasil Fungsi Evaluate

Penjelasan hasil pada gambar 6.70 :

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

11. Fungsi predict()

Berikut adalah kode program yang digunakan :

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

Parameter yang digunakan pada kode program :

:1 = digunakan untuk menentukan batasa row.

Apabila kode program tersebut dijalankan, maka akan menghasilkan seperti pada gambar 6.71.

```
In [23]: model.predict(test_input[:1])
Out[23]:
array([[2.5113127e-01, 7.7192662e-03, 5.1448198e-02, 6.4860745e-03,
       6.4208927e-03, 6.2923378e-01, 3.7388336e-06, 2.6338635e-02,
       3.9711967e-03, 1.7254945e-02]], dtype=float32)
```

Figure 6.71: Lusia-Hasil Fungsi Predic

Penjelasan hasil pada gambar 6.71 :

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

6.3.3 Penanganan Error

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

1. skrinsut error

(a) Error 1

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

Figure 6.72: Lusia-skrinsut error 1

(b) Error 2

```
In [3]: display_mfcc('266093_stereo-surgeon_kick-loop-5.wav')
Traceback (most recent call last):
  File "<ipython-input-3-58919770e4d6>", line 1, in <module>
    display_mfcc('266093_stereo-surgeon_kick-loop-5.wav')
  File "<ipython-input-2-59953e4547ad>", line 5, in display_mfcc
    plt.figure(figsize=(10, 4))
NameError: name 'plt' is not defined
```

Figure 6.73: Lusia-skrinsut error 2

2. Tuliskan kode error dan jenis errornya

(a) Error 1 :

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

(b) Error 2 :

- Kode error : NameError: name 'plt' is not defined
- Jenis error : Name Error

3. Solusi pemecahan masalah error

(a) Error 1

Error yang terjadi pada gambar 6.72 akibat dari kesalahan pada pengala-matan direktori sehingga file yang akan digunakan tidak ditemukan, oleh karena itu pengalamatan direktori disesuaikan dengan kebutuhan penggu-naan.

(b) Error 2

Error yang terjadi pada gambar 6.73 akibat dari library matplotlib.pyplot yang belum diinstall.

6.3.4 Cek plagiarisme

Dari teori hingga praktek yang telah dilakukan telah dilakukan cek plagiarisme seperti gamabar 6.74 berikut :

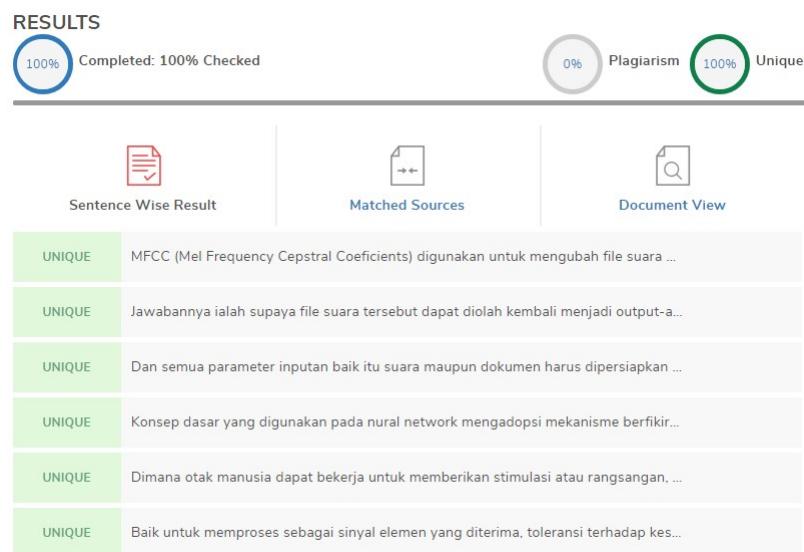


Figure 6.74: Lusia-Cek Plagiarisme

Chapter 7

NCC

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

7.1 Lusia Violita Aprilian-1184080

7.1.1 Teori

1. Menjelaskan kenapa file teks harus dilakukan tokenizer

Tokenizer adalah untuk membuat vektor dari teks. Dan mengapa harus dilakukan tokenizer? itu karena dengan memfungsikan tokenizer, teks dapat di-vektorkan. Sehingga teks yang telah telah divektorkan tersebut dapat terbaca pada Machine Learning.

Berikut adalah ilustrasi pemakaian pada tokenizer, perhatikan gambar 7.1

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

Figure 7.1: Lusia-Tokenizer

2. Menjelaskan konsep dasar K-Fold Cross Validation

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

Berdasarkan kode listing tersebut dapat dijelaskan bahwa :

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

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

Berikut adalah gambar ilustrasi dari kosep dasar kfold, perhatikan gambar 7.2.

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

Figure 7.2: Lusia-StratifiedKFold

3. Menjelaskan kode program for train, test in splits.

```
for train, test in splits
```

Berdasarkan kode program tersebut, maka dapat dijelaskan bahwa kode tersebut digunakan untuk mencetak posisi pengujian pada train dan test yang telah dipisahkan. Dan memastikan bahwa kedua data tersebut tidak terjadi overlapping atau tumpang tindih dalam setiap pemisahan.

Berikut adalah gambar ilustrasi dari for train, test in splits, perhatikan gambar 7.3.

```
In [31]: for train, test in splits:
    | print("Split")
    | print(test)
```

Figure 7.3: Lusia-for train, test in splits

Apabila kode program pada gambar 7.3 dijalankan, maka akan menghasilkan row_id sebanyak 5 bagian. Dan jika diperhatikan, setiap bagian tidak terjadi pengulangan sehingga bisa bisa dibilang tidak terjadi overlapping pada data. Perhatikan gambar 7.4.

Region 1	Region 2
1	1
2	2
3	3
4	4
5	5
6	6
7	7
8	8
9	9
10	10
11	11
12	12
13	13
14	14
15	15
16	16
17	17
18	18
19	19
20	20
21	21
22	22
23	23
24	24
25	25
26	26
27	27
28	28
29	29
30	30
31	31
32	32
33	33
34	34
35	35
36	36
37	37
38	38
39	39
40	40
41	41
42	42
43	43
44	44
45	45
46	46
47	47
48	48
49	49
50	50
51	51
52	52
53	53
54	54
55	55
56	56
57	57
58	58
59	59
60	60
61	61
62	62
63	63
64	64
65	65
66	66
67	67
68	68
69	69
70	70
71	71
72	72
73	73
74	74
75	75
76	76
77	77
78	78
79	79
80	80
81	81
82	82
83	83
84	84
85	85
86	86
87	87
88	88
89	89
90	90
91	91
92	92
93	93
94	94
95	95
96	96
97	97
98	98
99	99
100	100
101	101
102	102
103	103
104	104
105	105
106	106
107	107
108	108
109	109
110	110
111	111
112	112
113	113
114	114
115	115
116	116
117	117
118	118
119	119
120	120
121	121
122	122
123	123
124	124
125	125
126	126
127	127
128	128
129	129
130	130
131	131
132	132
133	133
134	134
135	135
136	136
137	137
138	138
139	139
140	140
141	141
142	142
143	143
144	144
145	145
146	146
147	147
148	148
149	149
150	150
151	151
152	152
153	153
154	154
155	155
156	156
157	157
158	158
159	159
160	160
161	161
162	162
163	163
164	164
165	165
166	166
167	167
168	168
169	169
170	170
171	171
172	172
173	173
174	174
175	175
176	176
177	177
178	178
179	179
180	180
181	181
182	182
183	183
184	184
185	185
186	186
187	187
188	188
189	189
190	190
191	191
192	192
193	193
194	194
195	195
196	196
197	197
198	198
199	199
200	200
201	201
202	202
203	203
204	204
205	205
206	206
207	207
208	208
209	209
210	210
211	211
212	212
213	213
214	214
215	215
216	216
217	217
218	218
219	219
220	220
221	221
222	222
223	223
224	224
225	225
226	226
227	227
228	228
229	229
230	230
231	231
232	232
233	233
234	234
235	235
236	236
237	237
238	238
239	239
240	240
241	241
242	242
243	243
244	244
245	245
246	246
247	247
248	248
249	249
250	250
251	251
252	252
253	253
254	254
255	255
256	256
257	257
258	258
259	259
260	260
261	261
262	262
263	263
264	264
265	265
266	266
267	267
268	268
269	269
270	270
271	271
272	272
273	273
274	274
275	275
276	276
277	277
278	278
279	279
280	280
281	281
282	282
283	283
284	284
285	285
286	286
287	287
288	288
289	289
290	290
291	291
292	292
293	293
294	294
295	295
296	296
297	297
298	298
299	299
300	300
301	301
302	302
303	303
304	304
305	305
306	306
307	307
308	308
309	309
310	310
311	311
312	312
313	313
314	314
315	315
316	316
317	317
318	318
319	319
320	320
321	321
322	322
323	323
324	324
325	325
326	326
327	327
328	328
329	329
330	330
331	331
332	332
333	333
334	334
335	335
336	336
337	337
338	338
339	339
340	340
341	341
342	342
343	343
344	344
345	345
346	346
347	347
348	348
349	349
350	350
351	351
352	352
353	353
354	354
355	355
356	356
357	357
358	358
359	359
360	360
361	361
362	362
363	363
364	364
365	365
366	366
367	367
368	368
369	369
370	370
371	371
372	372
373	373
374	374
375	375
376	376
377	377
378	378
379	379
380	380
381	381
382	382
383	383
384	384
385	385
386	386
387	387
388	388
389	389
390	390
391	391
392	392
393	393
394	394
395	395
396	396
397	397
398	398
399	399
400	400
401	401
402	402
403	403
404	404
405	405
406	406
407	407
408	408
409	409
410	410
411	411
412	412
413	413
414	414
415	415
416	416
417	417
418	418
419	419
420	420
421	421
422	422
423	423
424	424
425	425
426	426
427	427
428	428
429	429
430	430
431	431
432	432
433	433
434	434
435	435
436	436
437	437
438	438
439	439
440	440
441	441
442	442
443	443
444	444
445	445
446	446
447	447
448	448
449	449
450	450
451	451
452	452
453	453
454	454
455	455
456	456
457	457
458	458
459	459
460	460
461	461
462	462
463	463
464	464
465	465
466	466
467	467
468	468
469	469
470	470
471	471
472	472
473	473
474	474
475	475
476	476
477	477
478	478
479	479
480	480
481	481
482	482
483	483
484	484
485	485
486	486
487	487
488	488
489	489
490	490
491	491
492	492
493	493
494	494
495	495
496	496
497	497
498	498
499	499
500	500
501	501
502	502
503	503
504	504
505	505
506	506
507	507
508	508
509	509
510	510
511	511
512	512
513	513
514	514
515	515
516	516
517	517
518	518
519	519
520	520
521	521
522	522
523	5

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

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

Berikut ilustrasi dari kode program tersebut, perhatikan gambar 7.5.

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

Figure 7.5: Lusia-Ilustrasi Kode Program No.4

5. Menjelaskan maksud fungsi

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

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

Sedangkan pada baris kedua dari kode program tersebut menjelaskan bahwa tokenizer difungsikan pada data train yang telah di fitting.

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

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

Figure 7.6: Lusia-Ilustrasi Maksud fungsi No.5

6. Menjelaskan maksud fungsi

```
d_train_inputs = tokenizer.texts_to_matrix(train_content, mode='tfidf')
d_test_inputs = tokenizer.texts_to_matrix(test_content, mode='tfidf')
```

Dari fungsi pada kode program tersebut dapat dijelaskan bahwa :

- Baris pertama membuat variabel d_train_inputs untuk memanggil fungsi tokenizer dan merubah data train yang berupa teks ke dalam bentuk matrix dengan menggunakan model tfidf.

- (b) Baris kedua membuat variabel d_test_inputs untuk memanggil fungsi tokrnizer dan merubah data test yang berupa teks ke dalam bentuk matrix dengan menggunakan model tfidf.

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

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

Figure 7.7: Lusia-Illustrasi Maksud fungsi No.6

7. Menjelaskan maksud fungsi

```
d_train_inputs = d_train_inputs/np.amax(np.absolute(d_train_inputs))
d_test_inputs = d_test_inputs/np.amax(np.absolute(d_test_inputs))
```

Dari fungsi pada kode program tersebut dapat dijelaskan bahwa fungsi tersebut akan membagi matrix tfidf tadi dengan amax yaitu mengembalikan maksimum array atau maksimum sepanjang sumbu. Yang hasilnya akan dimasukan kedalam variabel d_train_inputs untuk data train dan d_test_inputs untuk data test dengan nominal absolut atau tanpa ada bilangan negatif dan koma.

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

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

Figure 7.8: Lusia-Illustrasi Maksud fungsi No.7

8. Menjelaskan maksud fungsi

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

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

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

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

Figure 7.9: Lusia-Illustrasi Maksud fungsi No.8

9. Menjelaskan maksud fungsi

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

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

Berikut adalah gambar ilustrasi dari fungsi pada kode program tersebut, perhatikan gambar 7.10.

```
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'))
```

Figure 7.10: Lusia-Illustrasi Maksud fungsi No.9

10. Menjelaskan maksud fungsi

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

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

Berikut adalah gambar ilustrasi dari fungsi pada kode program tersebut, perhatikan gambar 7.11.

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

Figure 7.11: Lusia-Illustrasi Maksud fungsi No.10

11. Menjelaskan apa itu Deep Learning

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

12. Menjelaskan 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.

Yang membedakan Deep Learning dengan Deep Neural Network (DNN) adalah DNN merupakan algoritma yang digunakan pada Deep Learning, sedangkan Deep Learning merupakan model yang menggunakan algoritma DNN.

13. Menjelaskan perhitungan algoritma konvolusi

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

Karena NPM saya 1164080 dan hasil dari $(NPM \bmod 3) + 1 = 3$, maka saya menggunakan matrik kernel berukuran 3×3 . Sehingga ilustrasi gambar yang digunakan adalah seperti gambar 7.12. Misalkan $f(x,y)$ yang digunakan berukuran 5×5 dan kernel atau mask berukuran 3×3 masing-masing adalah sebagai berikut:

$$f(x,y) = \begin{bmatrix} 4 & 4 & 3 & 5 & 4 \\ 6 & 6 & 5 & 3 & 1 \\ 5 & 6 & 6 & 6 & 2 \\ 6 & 7 & 5 & 5 & 3 \\ 2 & 5 & 2 & 4 & 4 \end{bmatrix} \quad \text{dan} \quad g(x,y) = \begin{bmatrix} 0 & -1 & 0 \\ -1 & 0.4 & -1 \\ 0 & -1 & 0 \end{bmatrix}$$

(keterangan: tanda · menyatakan posisi (0,0) dari kernel)

Figure 7.12: Lusia-Illustrasi Gambar

Penyelesaian dari operasi konvolusi antara $f(x,y)$ dengan kernel $g(x,y)$ pada gambar 7.12 adalah $f(x,y) * g(x,y)$ dengan ilustrasi sebagai berikut :

- Tempatkan matrik kernel di sebelah kiri atas, lalu hitung nilai piksel pada posisi (0,0) dari kernel tersebut seperti gambar 7.13.

4	4	3	5	4
6	6	5	6	2
5	6	6	6	1
6	7	5	5	3
3	5	2	4	4

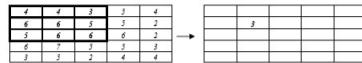


Figure 7.13: Lusia-Ilustrasi konvolusi 1

Konvolusi dihitung dengan cara berikut :

$$(0 \times 4) + (-1 \times 4) + (0 \times 3) + (-1 \times 6) + (4 \times 6) + (-1 \times 5) + (0 \times 5) + (-1 \times 6) + (0 \times 6)$$

Sehingga didapat hasil konvolusi = 3

- (b) Lalu geser kernel satu piksel ke kanan kemudian hitung kembali nilai piksel pada posisi (0,0) dari kernel.

Konvolusi dihitung dengan cara berikut :

$$(0 \times 4) + (-1 \times 3) + (0 \times 5) + (-1 \times 6) + (4 \times 5) + (-1 \times 5) + (0 \times 6) + (-1 \times 6) + (0 \times 6)$$

Sehingga didapat hasil konvolusi = 0 seperti pada gambar 7.14.

4	4	3	5	4
6	6	5	6	2
5	6	6	6	1
6	7	5	5	3
3	5	2	4	4

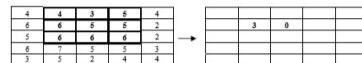


Figure 7.14: Lusia-Ilustrasi konvolusi 2

- (c) Lalu geser kernel satu piksel ke kanan kemudian hitung kembali nilai piksel pada posisi (0,0) dari kernel.

Konvolusi dihitung dengan cara berikut :

$$(0 \times 3) + (-1 \times 5) + (0 \times 4) + (-1 \times 5) + (4 \times 5) + (-1 \times 2) + (0 \times 6) + (-1 \times 6) + (0 \times 2)$$

Sehingga didapat hasil konvolusi = 2 seperti pada gambar 7.15.

4	4	3	5	4
6	6	5	6	2
5	6	6	6	1
6	7	5	5	3
3	5	2	4	4

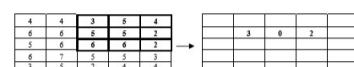


Figure 7.15: Lusia-Ilustrasi konvolusi 3

- (d) Kemudian geser matriks kernel kebawah, lalu mulai hitung kembali dari sisi kiri. Setiap kali perhitungan konvolusi dilakukan, geser matriks kernel atau piksel ke kanan seperti pada gambar 7.16, 7.17, dan 7.18.

- Konvolusi pada gambar 7.16 dihitung dengan cara berikut :

$$(0x6) + (-1x6) + (0x5) + (-1x5) + (4x6) + (-1x6) + (0x6) + (-1x7) + (0x5)$$

Sehingga didapat hasil konvolusi = 0 seperti pada gambar 7.16.

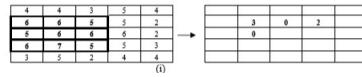


Figure 7.16: Lusia-Illustrasi konvolusi 4

- Konvolusi pada gambar 7.17 dihitung dengan cara berikut :

$$(0x6) + (-1x5) + (0x5) + (-1x6) + (4x6) + (-1x6) + (0x7) + (-1x5) + (0x5)$$

Sehingga didapat hasil konvolusi = 2 seperti pada gambar 7.17.

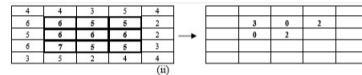


Figure 7.17: Lusia-Illustrasi konvolusi 5

- Konvolusi pada gambar 7.18 dihitung dengan cara berikut :

$$(0x5) + (-1x5) + (0x2) + (-1x6) + (4x6) + (-1x2) + (0x5) + (-1x5) + (0x3)$$

Sehingga didapat hasil konvolusi = 6 seperti pada gambar 7.18.

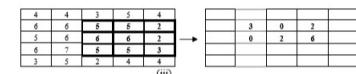


Figure 7.18: Lusia-Illustrasi konvolusi 6

- (e) Dengan langkah-langkah yang sama, piksel-piksel pada baris ketiga dilakukan operasi konvolusi sehingga menghasilkan seperti gambar 7.19.

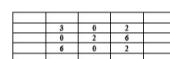


Figure 7.19: Lusia-Illustrasi konvolusi baris ketiga

Apabila nilai piksel hasil konvolusi adalah negatif, maka nilai tersebut dijadikan 0. Sebaliknya, bila nilai piksel hasil konvolusi lebih besar dari nilai kabuan maksimum (255), maka nilai tersebut dijadikan ke nilai keabuan maksimum.

7.1.2 Praktek

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

Berikut adalah kode program yang digunakan :

Listing 7.1: Kode Program 1

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

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

- Baris 1 : Melakukan pengimportan file csv
- Baris 2 : Melakukan pemanggilan atau memasukkan module image sebagai pil_image dari library PIL
- Baris 3 : Melakukan pengimportan fungsi keras.preprocessing.image

Sehingga dari kode program tersebut bila dijalankan, maka menghasilkan seperti pada gambar 7.20.

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

Figure 7.20: Lusia-Hasil Kode Program 1

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

Berikut adalah kode program yang digunakan :

Listing 7.2: Kode Program 2

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

```

        imgs . append ( ( row [ 0 ] , row [ 2 ] , img ) )
        classes . append ( row [ 2 ] )
i += 1

```

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

- Baris 1 : Membuat variabel imgs tanpa ada parameter di dalamnya
- Baris 2 : Membuat variabel classes tanpa ada parameter didalamnya
- Baris 3 : Membuka file csv dari HASYv2/hasy-data-labels.csv sebagai csv-file
- Baris 4 : Membuat variabel csvreader yang difungsikan untuk membaca dari file csv yang dimasukkan
- Baris 5 : Membuat variabel i dengan parameter 0 atau nilai 0
- Baris 6 : Digunakan untuk melakukan eksekusi baris dari pembacaan csv
- Baris 7 : Mengaplikasikan atau menggunakan perintah "if" dengan variabel i lebih besar dari 0, yang selanjutnya akan dilanjutkan ke perintah berikutnya
- Baris 8 : Membuat variabel img yang berfungsi untuk mengubah image atau gambar menjadi bentuk array (bilangan) dari file HASYv2 yang dibuka dengan row berparameter 0.
- Baris 9 : Membuat variabel img dengan nilai bukan sama dengan 255.0
- Baris 10 : Mendefinisikan fungsi imgs.append yang digunakan untuk melakukan proses penggabungan data dengan file lain atau dataset lain yang telah ditentukan dengan 3 parameter yaitu row[0], row[2] dan variabel img.
- Baris 11 : Mendefinisikan fungsi append dari variabel classes dengan menggunakan parameter row[2].
- Baris 12 : Mengartikan $i = i + 1$ yang dimana nilai sari variabel i akan ditambah 1 sehingga akan bernilai 1.

Sehingga dari kode program tersebut bila dijalankan, maka menghasilkan seperti pada gambar 7.21.

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

Berikut adalah kode program yang digunakan :

```

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

```

Figure 7.21: Lusia-Hasil Kode Program 2

Listing 7.3: Kode Program 3

```

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

```

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

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

Sehingga dari kode program tersebut bila dijalankan, maka menghasilkan seperti pada gambar 7.22.

```

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

```

Figure 7.22: Lusia-Hasil Kode Program 3

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

Berikut adalah kode program yang digunakan :

Listing 7.4: Kode Program 4

```
import numpy as np

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

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

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

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

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

Sehingga dari kode program tersebut bila dijalankan, maka menghasilkan seperti pada gambar 7.23.

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

Figure 7.23: Lusia-Hasil Kode Program 4

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

Berikut adalah kode program yang digunakan :

Listing 7.5: Kode Program 5

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

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

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

Sehingga dari kode program tersebut bila dijalankan, maka menghasilkan seperti pada gambar 7.24.

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

Figure 7.24: Lusia-Hasil Kode Program 5

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

Berikut adalah kode program yang digunakan :

Listing 7.6: Kode Program 6

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

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

- Baris 1 : Membuat variabel label_encoder dengan penerapan modul / fungsi LabelEncoder tanpa parameter
- Baris 2 : Membuat variabel integer_encoded dengan penerapan fungsi label_encoder.fit_transform yang berfungsi untuk melakukan ekstrasi fitur object dari variabel classes yang akan mengembalikan beberapa data yang diubah kembali.

Sehingga dari kode program tersebut bila dijalankan, maka menghasilkan seperti pada gambar 7.25.

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

Figure 7.25: Lusia-Hasil Kode Program 6

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

Berikut adalah kode program yang digunakan :

Listing 7.7: Kode Program 7

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

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

- Baris 1 : Membuat variabel onehot_encoder yang memanggil fungsi OneHotEncoder tanpa mengembalikan matriks karena sparse=false.
- Baris 2 : Membuat variabel integer_encoded memanggil variabel integer_encoded pada kode program 6 untuk dieksekusi memberikan bentuk baru ke array tanpa mengubah datanya dari mengembalikan panjang nilai dari integer_encoded.

- Baris 3 : Onehotencoding melakukan fitting pada integer_encoded.

Sehingga dari kode program tersebut bila dijalankan, maka menghasilkan seperti pada gambar 7.26.

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

Figure 7.26: Lusia-Hasil Kode Program 7

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

Berikut adalah kode program yang digunakan :

Listing 7.8: Kode Program 8

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

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

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

- Baris 1 : Membuat variabel train_output_int yang mengeksekusi label_encoder dengan mengubah nilai dari parameter variabel train_output.
- Baris 2 : Membuat variabel train_output yang mengeksekusi variabel one-hot_encoder dari kode program 7 dengan mengubah nilai dari variabel parameter train_output_int yang datanya sudah diubah kedalam bentuk array dan panjang nilai dari train_output_int telah dikembalikan.
- Baris 3 : Membuat variabel test_output_int yang mengeksekusi label_encoder dengan mengubah nilai dari parameter variabel test_output.
- Baris 4 : Membuat variabel test_output yang mengeksekusi variabel one-hot_encoder dari kode program 7 dengan mengubah nilai dari variabel parameter test_output_int yang datanya sudah diubah kedalam bentuk array dan panjang nilai dari test_output_int telah dikembalikan.

- Baris 5 : Membuat variabel num_classes untuk mengetahui jumlah class dari label_encoder
- Baris 6 : Perintah print digunakan untuk memunculkan hasil dari variabel num_classes

Sehingga dari kode program tersebut bila dijalankan, maka menghasilkan seperti pada gambar 7.27.

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

Figure 7.27: Lusia-Hasil Kode Program 8

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

Berikut adalah kode program yang digunakan :

Listing 7.9: Kode Program 9

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

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

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

Sehingga dari kode program tersebut bila dijalankan, maka menghasilkan seperti pada gambar 7.28.

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

Figure 7.28: Lusia-Hasil Kode Program 9

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

Berikut adalah kode program yang digunakan :

Listing 7.10: Kode Program 10

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

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

print(model.summary())
```

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

- Baris 1 :
- Baris 2 :

Sehingga dari kode program tersebut bila dijalankan, maka menghasilkan seperti pada gambar 7.29.

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

Berikut adalah kode program yang digunakan :

Listing 7.11: Kode Program 11

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

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

- Baris 1 : Melakukan importing library keras.callbacks yang memiliki fungsi penulisan log untuk TensorBoard, yang memungkinkan untuk memvisualisasikan grafik dinamis dari training dan metrik pengujian.

```

In [10]: model = Sequential()
...:     model.add(Conv2D(32, kernel_size=(3, 3), activation='relu',
...:                     input_shape=inp.shape[train_input[0]]))
...:     model.add(MaxPooling2D(pool_size=(2, 2)))
...:     model.add(Conv2D(64, kernel_size=(3, 3), activation='relu'))
...:     model.add(MaxPooling2D(pool_size=(2, 2)))
...:     model.add(Flatten())
...:     model.add(Dense(1024, activation='tanh'))
...:     model.add(Dense(512, activation='tanh'))
...:     model.add(Dense(num_classes, activation='softmax'))
...: 
...:     model.compile(loss='categorical_crossentropy', optimizer='adam',
...:                   metrics=['accuracy'])
...: 
...:     print(model.summary())
WARNING:tensorflow:Using or importing the following from tensorflow.python.framework
        op_def_library.py:263: colocate_with (from tensorflow.python.framework.ops) is deprecated and will be
        removed in a future version.
        Instructions for updating:
        Colocate operations explicitly by placing
WARNING:tensorflow:From C:\Users\livaap\Anaconda3\lib\site-packages\keras\backend\tensorflow_backend.py:
        3444: calling dropout (from tensorflow.python.ops.nn_ops) with keep_prob is deprecated and will be removed
        in a future version.
        Instructions for updating:
        Please use rate instead of 'keep_prob'. Rate should be set to 'rate = 1 - keep_prob'.


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


Total params: 1,569,041
Trainable params: 1,569,041
Non-trainable params: 0
None

```

Figure 7.29: Lusia-Hasil Kode Program 10

- Baris 2 : Membuat variabel tensorboard yang menggunakan fungsi TensorBoard dari keras.callbacks yang berfungsi sebagai alat visualisasi yang telah disediakan oleh TensorFlow. Kemudian untuk fungsi log_dir memanggil data yaitu './logs/mnist-style' dari direktori.

Sehingga dari kode program tersebut bila dijalankan, maka menghasilkan seperti pada gambar 7.30.

```

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

```

Figure 7.30: Lusia-Hasil Kode Program 11

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

Berikut adalah kode program yang digunakan :

Listing 7.12: Kode Program 12

```

model.fit(train_input, train_output,
          batch_size=32,
          epochs=10,
          verbose=2,
          validation_split=0.2,
          callbacks=[tensorboard])

score = model.evaluate(test_input, test_output, verbose=2)
print('Test loss:', score[0])
print('Test accuracy:', score[1])

```

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

- Baris 1 : Menerapkan fungsi model.fit yang didalamnya memproses train_input, train_output dengan batch_size, epochs, verbose, validation_split, dan callbacks.
 - Batch_size merupakan jumlah sampel per pembaharuan sampel dari data yang diolah, sehingga apabila batch_size-nya tidak ditemukan maka otomatis akan dijadikan nilai 32.
 - Epochs berfungsi untuk melakukan perulangan dimana perulangan dari berapa kali nilai yang digunakan untuk data, dan jumlahnya ialah 10.
 - Verbose digunakan sebagai opsi untuk menghasilkan informasi logging dari data yang ditentukan dengan nilai 2.
 - Validation_split berfungsi untuk memecah nilai dari perhitungan dengan validasinya sebesar 0,2. (Fraksi data pelatihan untuk digunakan sebagai data validasi).
 - Callsbacks mengeksekusi tensorboard yang berfungsi untuk memvisualisasikan parameter training, metrik, hiperparameter pada nilai/data yang diproses.
- Baris 2 : Membuat variabel score dengan menggunakan fungsi evaluate dari model yang ada dengan variabel parameter test_input, test_output dan verbose=2 yang berfungsi memprediksi output untuk input yang diberikan dan kemudian menghitung fungsi metrik yang ditentukan dalam modelnya.
- Baris 3 : Mencetak variabel score optimasi dari test dengan ketentuan nilai parameter 0
- Baris 4 : Mencetak variabel score akurasi dari test dengan ketentuan nilai parameter 1

Sehingga dari kode program tersebut bila dijalankan, maka menghasilkan seperti pada gambar 7.31.

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

Berikut adalah kode program yang digunakan :

Listing 7.13: Kode Program 13

```

In [12]: model.fit(train_input, train_output,
...:     batch_size=32,
...:     epochs=10,
...:     verbose=2,
...:     validation_split=0.2,
...:     callbacks=[tensorboard])
...:     score = model.evaluate(test_input, test_output, verbose=2)
...: print('Test loss:', score[0])
...: print('Test accuracy:', score[1])
ValueError: from tensorflow.python.libsite-packages\tensorflow\python\ops\math_ops.py:906: to_int32 (from tensorflow.python.ops.math_ops) is deprecated and will be removed in a future version.
Instructions for updating:
Use tf.cast instead.
Train on 26910 samples, validate on 26910 samples
Epoch 1/10
- 226s - loss: 1.5626 - acc: 0.6232 - val_loss: 1.0003 - val_acc: 0.7257
Epoch 2/10
- 226s - loss: 0.9856 - acc: 0.7277 - val_loss: 0.9087 - val_acc: 0.7443
Epoch 3/10
- 219s - loss: 0.8764 - acc: 0.7508 - val_loss: 0.8710 - val_acc: 0.7556
Epoch 4/10
- 219s - loss: 0.8005 - acc: 0.7679 - val_loss: 0.8437 - val_acc: 0.7564
Epoch 5/10
- 219s - loss: 0.7533 - acc: 0.7770 - val_loss: 0.8118 - val_acc: 0.7684
Epoch 6/10
- 219s - loss: 0.7091 - acc: 0.7860 - val_loss: 0.8451 - val_acc: 0.7617
Epoch 7/10
- 244s - loss: 0.6774 - acc: 0.7927 - val_loss: 0.8559 - val_acc: 0.7622
Epoch 8/10
- 255s - loss: 0.6447 - acc: 0.8002 - val_loss: 0.8580 - val_acc: 0.7683
Epoch 9/10
- 278s - loss: 0.6219 - acc: 0.8054 - val_loss: 0.8877 - val_acc: 0.7621
Epoch 10/10
- 221s - loss: 0.6066 - acc: 0.8092 - val_loss: 0.8827 - val_acc: 0.7693
Test loss: 0.871598460094306
Test accuracy: 0.764228608794817

```

Figure 7.31: Lusia-Hasil Kode Program 12

```
import time
```

```

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

```

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

- Baris 1 :
- Baris 2 :

Sehingga dari kode program tersebut bila dijalankan, maka menghasilkan seperti pada gambar 7.32.

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

Berikut adalah kode program yang digunakan :

```

In [13]: import time
...
...: results = []
...: for conv2d_count in [1, 2]:
...:     for dense_size in [128, 256, 512, 1024, 2048]:
...:         for dropout in [0.0, 0.25, 0.50, 0.75]:
...:             model = Sequential()
...:             for i in range(conv2d_count):
...:                 if i == 0:
...:                     model.add(Conv2D(32, kernel_size=(3, 3), activation='relu', input_shape=np.shape(train_input[0])))
...:                 else:
...:                     model.add(Conv2D(32, kernel_size=(3, 3), activation='relu'))
...:             model.add(MaxPooling2D(pool_size=(2, 2)))
...:             model.add(Dense(dense_size, activation='tanh'))
...:             if dropout > 0.0:
...:                 model.add(Dropout(dropout))
...:             model.add(Dense(num_classes, activation='softmax'))
...:             model.compile(loss='categorical_crossentropy', optimizer='adam', metrics=['accuracy'])
...:             log_dir = './logs/conv2d_3d-dense_%.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,
...:             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: %.2f, Dropout: %.2f - Loss: %.2f, Accuracy: %.2f, Time: %.2f sec" % (conv2d_count, dense_size, dropout,
...:             score[0], score[1], elapsed))
...:             results.append((conv2d_count, dense_size, dropout, score[0], score[1], elapsed))
Conv2D count: 1, Dense size: 128, Dropout: 0.00 - Loss: 1.09, Accuracy: 0.74, Time: 1472 sec
Conv2D count: 1, Dense size: 128, Dropout: 0.25 - Loss: 0.76, Accuracy: 0.76, Time: 1453 sec
Conv2D count: 1, Dense size: 128, Dropout: 0.50 - Loss: 0.89, Accuracy: 0.75, Time: 1525 sec
Conv2D count: 1, Dense size: 128, Dropout: 0.75 - Loss: 0.79, Accuracy: 0.77, Time: 1537 sec
Conv2D count: 1, Dense size: 256, Dropout: 0.00 - Loss: 1.32, Accuracy: 0.74, Time: 1967 sec
Conv2D count: 1, Dense size: 256, Dropout: 0.25 - Loss: 1.12, Accuracy: 0.76, Time: 2030 sec
Conv2D count: 1, Dense size: 256, Dropout: 0.50 - Loss: 0.90, Accuracy: 0.78, Time: 2047 sec

```

Figure 7.32: Lusia-Hasil Kode Program 13

Listing 7.14: Kode Program 14

```

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

```

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

- Baris 1 :
- Baris 2 :

Sehingga dari kode program tersebut bila dijalankan, maka menghasilkan seperti pada gambar 7.33.

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

Berikut adalah kode program yang digunakan :

Listing 7.15: Kode Program 15

```
model.fit(np.concatenate((train_input, test_input)),
```

```

In [14]: model = Sequential()
...: model.add(Conv2D(32, kernel_size=(3, 3), activation='relu', input_shape=np.shape(train_input[0])))
...: model.add(MaxPooling2D(pool_size=(2, 2)))
...: model.add(Conv2D(32, kernel_size=(3, 3), activation='relu'))
...: model.add(MaxPooling2D(pool_size=(2, 2)))
...: model.add(Flatten())
...: model.add(Dense(128, activation='tanh'))
...: model.add(Dropout(0.5))
...: model.add(Dense(num_classes, activation='softmax'))
...: model.compile(loss='categorical_crossentropy', optimizer='adam', metrics=['accuracy'])
...: print(model.summary())

```

Layer (type)	Output Shape	Param #
conv2d_11 (Conv2D)	(None, 30, 30, 32)	896
max_pooling2d_11 (MaxPooling)	(None, 15, 15, 32)	0
conv2d_12 (Conv2D)	(None, 13, 13, 32)	9248
max_pooling2d_12 (MaxPooling)	(None, 6, 6, 32)	0
flatten_10 (Flatten)	(None, 1152)	0
dense_19 (Dense)	(None, 128)	147584
dropout_8 (Dropout)	(None, 128)	0
dense_20 (Dense)	(None, 369)	47601

```

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

```

Figure 7.33: Lusia-Hasil Kode Program 14

```

np.concatenate((train_output, test_output)),
batch_size=32, epochs=10, verbose=2)

```

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

- Baris 1 :
- Baris 2 :

Sehingga dari kode program tersebut bila dijalankan, maka menghasilkan seperti pada gambar 7.34.

```

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
- 247s - loss: 1.7758 - acc: 0.5868
Epoch 2/10
- 227s - loss: 1.0771 - acc: 0.7063
Epoch 3/10
- 228s - loss: 0.9602 - acc: 0.7312
Epoch 4/10
- 229s - loss: 0.9026 - acc: 0.7441
Epoch 5/10
- 229s - loss: 0.8662 - acc: 0.7515
Epoch 6/10
- 229s - loss: 0.8316 - acc: 0.7597
Epoch 7/10
- 228s - loss: 0.8081 - acc: 0.7632
Epoch 8/10
- 229s - loss: 0.7907 - acc: 0.7664
Epoch 9/10
- 230s - loss: 0.7750 - acc: 0.7705
Epoch 10/10
- 230s - loss: 0.7636 - acc: 0.7723
Out[15]: <keras.callbacks.History at 0x22c12d05b70>

```

Figure 7.34: Lusia-Hasil Kode Program 15

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

Berikut adalah kode program yang digunakan :

Listing 7.16: Kode Program 16

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

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

- Baris 1 : Menyimpan model sebagai mathsymbols.model

Sehingga dari kode program tersebut bila dijalankan, maka menghasilkan seperti pada gambar 7.35.

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

Figure 7.35: Lusia-Hasil Kode Program 16

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

Berikut adalah kode program yang digunakan :

Listing 7.17: Kode Program 17

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

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

- Baris 1 : Menyimpan array dari label_encoder.classes_ ke file biner dalam format NumPy .npy.

Sehingga dari kode program tersebut bila dijalankan, maka menghasilkan seperti pada gambar 7.36.

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

Figure 7.36: Lusia-Hasil Kode Program 17

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

Berikut adalah kode program yang digunakan :

Listing 7.18: Kode Program 18

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

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

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

Sehingga dari kode program tersebut bila dijalankan, maka menghasilkan seperti pada gambar 7.37.

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

Figure 7.37: Lusia-Hasil Kode Program 18

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

Berikut adalah kode program yang digunakan :

Listing 7.19: Kode Program 19

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

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

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

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

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

- Baris 1 :
- Baris 2 :
- Baris 3 :
- Baris 4 :

Sehingga dari kode program tersebut bila dijalankan, maka menghasilkan seperti pada gambar 7.38.

```
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 7.38: Lusia-Hasil Kode Program 19

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

Berikut adalah kode program yang digunakan :

Listing 7.20: Kode Program 20

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

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

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

Sehingga dari kode program tersebut bila dijalankan, maka menghasilkan seperti pada gambar 7.39.

```

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: 0.82
Prediction: \pi, confidence: 0.80
Prediction: \alpha, confidence: 0.88

```

Figure 7.39: Lusia-Hasil Kode Program 20

7.1.3 Penanganan Error

1. skrinsut error

(a) Error 1 : gambar 7.40

```

File "C:\Users\lsvapr\Anaconda3\lib\site-packages\keras\backend\tensorflow_backend.py", line 5, in
<module>
    import tensorflow as tf
ModuleNotFoundError: No module named 'tensorflow'

```

Figure 7.40: Lusia-skrinsut error 1

(b) Error 2 : gambar 7.41

```

File "<ipython-input-3-7a17bf52a37d>", line 3, in <module>
    with open('HASYv2/hasy-data-labels.csv') as csvfile:
FileNotFoundError: [Errno 2] No such file or directory: 'HASYv2/hasy-data-labels.csv'

```

Figure 7.41: Lusia-skrinsut error 2

2. Tuliskan kode eror dan jenis errornya

(a) Error 1

Berdasarkan gambar 7.40 berikut adalah kode eror dan jenis errornya :

- Kode error : ModuleNotFoundError: No module named 'tensorflow'
- Jenis error : Module Not Found Error

(b) Error 2

Berdasarkan gambar 7.41 berikut adalah kode eror dan jenis errornya :

- Kode error : FileNotFoundError: [Errno 2] No such file or directory: 'HASYv2/hasy-data-labels.csv'
- Jenis error : File Not Found Error

3. Solusi pemecahan masalah error tersebut

(a) Error 1

Berdasarkan gambar 7.40 error yang terjadi akibat modul tensorflow belum diinstal, maka penyelesaiannya adalah dengan menginstal terlebih dahulu modul yang akan digunakan.

(b) Error 2

Berdasarkan gambar 7.41 error yang terjadi akibat direktori file yang akan digunakan tidak sesuai, maka solusinya dengan menyesuaikan direktori dari file yang akan digunakan.

7.1.4 Cek Plagiarisme

Dari hasil kerja pada chapter 7, jika dicek plagiarisme menghasilkan seperti gambar 7.42.

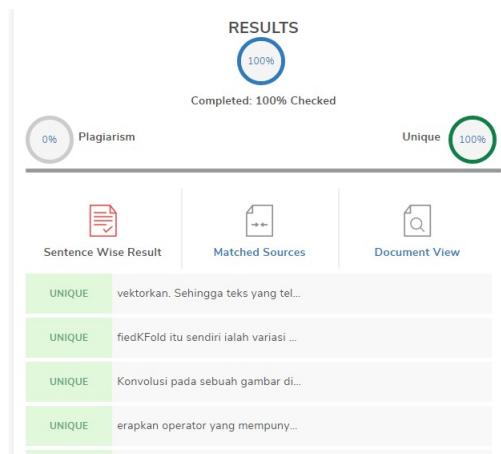


Figure 7.42: Lusia-Plagiarisme

7.2 Rahmi Roza-1164085

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 7.43



Figure 7.43: Tokenizer Roza

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

```
kfold = StratifiedKFold ( n_splits =5)
splits = kfold . 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 7.44

```
Cross validation using cross_val_score for five-fold CV

In [88]: cv_scores = cross_val_score(regr_model, X, y, cv= 5)
print("CV scores for each fold: ", cv_scores, '\nMean of CV scores: ', np.mean(cv_scores))

CV scores for each fold: [ 0.97467604  0.97751403  0.96845209  0.97719745  0.97445288]
Mean of CV scores:  0.974458498631
```

Figure 7.44: K-Fold Cross Validation Roza

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 7.45

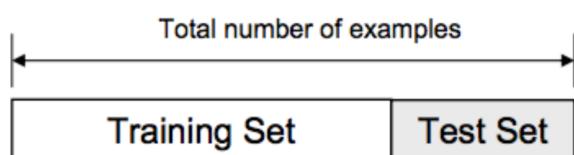


Figure 7.45: No 3 Roza

4. Jelaskan apa maksudnya kode program `train_content = d['CONTENT'].iloc[train_idx]` dan `test_content = d['CONTENT'].iloc[test_idx]`.
 - 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.
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 7.46

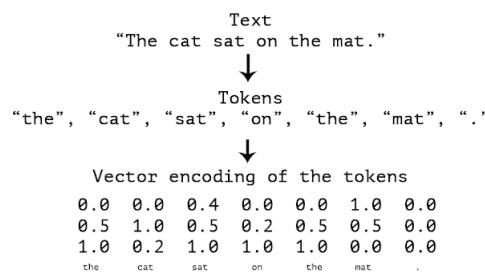


Figure 7.46: No 5 Roza

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 `tfidf` begitu juga dengan variabel `d_test_inputs` untuk data `test`. Berikut gambar ilustrasinya
- Ilustrasi Gambar
- No 6 7.47

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 7.47: No 6 Roza

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 `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.

- Ilustrasi Gambar
- No 7 7.48

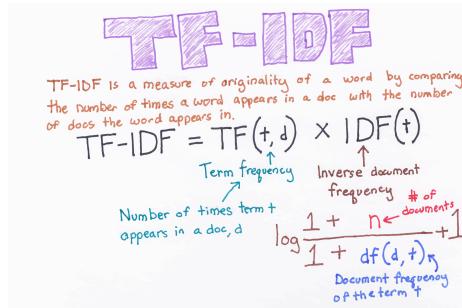


Figure 7.48: No 7 Roza

8. Jelaskan apa maksud fungsi dari `d_train_outputs = np_utils.to_categorical(d['CLASS'].iloc[train]` dan `d_test_outputs = np_utils.to_categorical(d['CLASS'].iloc[test_idx])` dalam kode program
 - 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
 - No 8 7.49

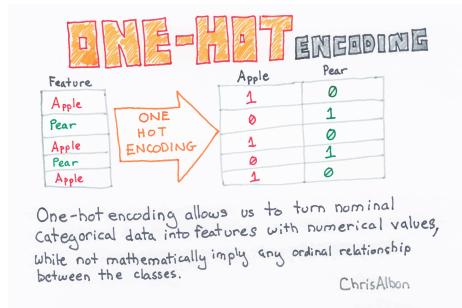


Figure 7.49: No 8 Roza

9. Jelaskan apa maksud dari fungsi di listing!

```

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

```

- 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!

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

- Penjelasan: Dari fungsi pada kode program tersebut model yang telah dibuat selanjutnya dicompile dengan menggunakan algoritma optimisasi dengan 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.

13. Perhitungan Algoritma Konvolusi Dengan NPM 1164085. Seperti gambar di bawah penyelesaiannya:

- Ilustrasi Gambar
- No 13 7.51

NPM saya 1164085 dan hasil dari $(NPM \bmod 3) + 1 = 2$, maka saya menggunakan matrik kernel berukuran 2×2 . Misalkan $f(x,y)$ yang digunakan berukuran 3×3 dan kernel atau mask berukuran 2×2 masing-masing adalah sebagai berikut:

Soal:

$$f(x,y) = \begin{matrix} 4 & 3 & 2 \\ 1 & 2 & 3 \\ 1 & 2 & 0 \end{matrix} \quad \text{dan } g(x,y) = \begin{matrix} 0 & 1 \\ 2 & 1 \end{matrix}$$

Penyelesaian dari operasi konvolusi antara $f(x,y)$ dengan kernel $g(x,y)$ pada gambar di atas adalah $f(x,y) * g(x,y)$. Menghitung posisi matrix dari ujung kiri atas pada $f(x,y)$ ke $g(x,y)$ seperti berikut:

1. $(4 \times 0) + (3 \times 1) + (1 \times 2) + (2 \times 1) = 0 + 3 + 2 + 2 = 7$
2. $(3 \times 0) + (2 \times 1) + (2 \times 2) + (3 \times 1) = 0 + 2 + 4 + 3 = 9$
3. $(1 \times 0) + (2 \times 1) + (1 \times 0) + (2 \times 1) = 0 + 2 + 0 + 2 = 4$
4. $(2 \times 0) + (3 \times 1) + (2 \times 2) + (0 \times 1) = 0 + 3 + 4 + 0 = 7$

Seperti hasil dari perhitungan di atas adalah sebagai berikut:

$$\begin{matrix} 7 & 9 \\ 4 & 7 \end{matrix}$$

Figure 7.50: No 13 Roza

14. Plagiarisme Roza



Figure 7.51: Plagiarisme Roza

7.2.2 Praktek

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

- Kode Program:

Listing 7.21: Praktek1.py

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

Hasil 7.52 :

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

Figure 7.52: In 1 Roza

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:

Listing 7.22: Praktek2.py

```
# In [2]: load all images (as numpy arrays) and save their classes

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

```

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

```

Figure 7.53: In 2 Roza

Hasil 7.53 :

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.preprocessing 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:

Listing 7.23: Praktek3.py

```
# In [3]: shuffle the data , split into 80% train , 20% test

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

Hasil 7.54 :

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

Figure 7.54: In 3 Roza

Baris 1: Import modul random

Baris 2: Melakukan pengocokan terhadap modul dengan parameter di mana variabelnya imgs

Baris 3: Membagi dan memecah index dalam bentuk integer dengan mengaklikan 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:

Listing 7.24: Praktek4.py

```
# 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)))
```

Hasil 7.55 :

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

Figure 7.55: In 4 Roza

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 mengkoleksikan 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 mengkoleksikan 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:

Listing 7.25: Praktek5.py

```
# In [5]: import encoder and one hot
from sklearn.preprocessing import LabelEncoder
from sklearn.preprocessing import OneHotEncoder
```

Hasil 7.56 :

```
In [8]: from sklearn.preprocessing import LabelEncoder
...: from sklearn.preprocessing import OneHotEncoder
```

Figure 7.56: In 5 Roza

Baris 1: Import labelEncoder dari sklearn.preprocessing 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.preprocessing yang digunakan untuk mendefinisikan fitur input dimana mengambil nilai dalam kisaran [0, maks (nilai)].

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

- Kode Program:

Listing 7.26: Praktek5.py

```
# In [6]: convert class names into one-hot encoding

# first, convert class names into integers
label_encoder = LabelEncoder()
integer_encoded = label_encoder.fit_transform(classes)
```

Hasil 7.58 :

```
In [12]: label_encoder = LabelEncoder()
...: integer_encoded = label_encoder.fit_transform(classes)
```

Figure 7.57: In 6 Roza

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:

Listing 7.27: Praktek7.py

```
# In [7]: then convert integers into one-hot encoding
onehot_encoder = OneHotEncoder(sparse=False)
integer_encoded = integer_encoded.reshape(len(integer_encoded), 1)
onehot_encoder.fit(integer_encoded)
```

Hasil ?? :

```
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 7.58: In 7 Roza

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:

Listing 7.28: Praktek8.py

```
# In [8]: convert train and test output to one-hot
train_output_int = label_encoder.transform(train_output)
train_output = onehot_encoder.transform(train_output_int.reshape(-1, 1))
test_output_int = label_encoder.transform(test_output)
test_output = onehot_encoder.transform(test_output_int.reshape(-1, 1))

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

Hasil 7.59 :

```

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 7.59: In 8 Roza

Baris 1: Membuat variabel train_output_int yang mengeksekusi label_encoder dengan mengubah nilai dari parameter variabel train_output.

Baris 2: Membuat variabel train_output yang mengeksekusi variabel one-hot_encoder dari kode program 7 dengan mengubah nilai dari variabel parameter train_output_int yang datanya sudah diubah kedalam bentuk array dan panjang nilai dari train_output_int telah dikembalikan.

Baris 3: Membuat variabel test_output_int yang mengeksekusi label_encoder dengan mengubah nilai dari parameter variabel test_output.

Baris 4: Membuat variabel test_output yang mengeksekusi variabel one-hot_encoder dari kode program 7 dengan mengubah nilai dari variabel parameter test_output_int yang datanya sudah diubah kedalam bentuk array dan panjang nilai dari test_output_int telah dikembalikan.

Baris 5: Membuat variabel num_classes untuk mengetahui jumlah class dari label_encoder

Baris 6: Perintah print digunakan untuk memunculkan hasil dari variabel num_classes

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

- Kode Program:

Listing 7.29: Praktek9.py

```

# In [9]: import sequential

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

```

Hasil 7.60 :

Baris 1: Memanggil atau melakukan importing fungsi model sequential dari library keras.

```
In [11]: from keras.models import Sequential
...: from keras.layers import Dense, Dropout, Flatten
...: from keras.layers import Conv2D, MaxPooling2D
```

Figure 7.60: In 9 Roza

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:

Listing 7.30: Praktek10.py

```
# In [10]: desain jaringan
model = Sequential()
model.add(Conv2D(32, kernel_size=(3, 3), activation='relu',
                 input_shape=np.shape(train_input[0])))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Conv2D(32, (3, 3), activation='relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Flatten())
model.add(Dense(1024, activation='tanh'))
model.add(Dropout(0.5))
model.add(Dense(num_classes, activation='softmax'))

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

print(model.summary())
```

Hasil 7.61 :

Baris 1:

Baris 2:

Baris 3:

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

- Kode Program:

Listing 7.31: Praktek11.py

```
# In [11]: import sequential
```

```

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 7.61: In 10 Roza

```

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

```

Hasil 7.62 :

```

In [13]: import keras.callbacks
...: tensorboard = keras.callbacks.TensorBoard(log_dir='./logs/mnist-style')

```

Figure 7.62: In 11 Roza

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:

Listing 7.32: Praktek12.py

```
# In [12]: 5menit kali 10 epoch = 50 menit
```

```

model.fit(train_input, train_output,
          batch_size=32,
          epochs=10,
          verbose=2,
          validation_split=0.2,
          callbacks=[tensorboard])

score = model.evaluate(test_input, test_output, verbose=2)
print('Test loss:', score[0])
print('Test accuracy:', score[1])

```

Hasil 7.63 :

```

2021-07-12 14:54:42.444
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 7.63: In 12 Roza

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 parameteranya 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:

Listing 7.33: Praktek13.py

```
# In [13]: try various model configurations and parameters to find the best one

import time

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

Hasil 7.64 :

```

...
...
    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 7.64: In 13 Roza

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:

Listing 7.34: Praktek14.py

```
# In [14]: rebuild/retrain a model with the best parameters (from
model = Sequential()
model.add(Conv2D(32, kernel_size=(3, 3), activation='relu', input_shape=(28, 28, 1)))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Conv2D(32, (3, 3), activation='relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Flatten())
model.add(Dense(128, activation='tanh'))
model.add(Dropout(0.5))
model.add(Dense(num_classes, activation='softmax'))
model.compile(loss='categorical_crossentropy', optimizer='adam')
print(model.summary())
```

Hasil 7.65 :

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

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 7.65: In 14 Roza

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 untuk 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:

Listing 7.35: Praktek15.py

```
# In [15]: join train and test data so we train the network on all
model.fit(np.concatenate((train_input, test_input)),
          np.concatenate((train_output, test_output)),
          batch_size=32, epochs=10, verbose=2)
```

Hasil 7.66 :

Baris 1: Melakukan fit dengan join data train dan test agar dapat dilakukan pelatihan untuk jaringan pada semua data yang dimiliki.

```

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 7.66: In 15 Roza

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

- Kode Program:

Listing 7.36: Praktek1.py

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

Hasil 7.67 :

```
In [27]: model.save("mathsymbols.model")
```

Figure 7.67: In 16 Roza

Baris 1: Menyimpan atau save model yang telah di latih dengan nama mathsymbols.model

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

- Kode Program:

Listing 7.37: Praktek1.py

```
# In [17]: save label encoder (to reverse one-hot encoding)
np.save('classes.npy', label_encoder.classes_)
```

Hasil 7.68 :

Baris 1: Simpan label enkoder (untuk membalikkan one-hot encoder) dengan nama classes.npy

```
In [28]: np.save('classes.npy', label_encoder.classes_)
```

Figure 7.68: In 17 Roza

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

- Kode Program:

Listing 7.38: Praktek18.py

```
# In [18]: load the pre-trained model and predict the math symbol
# the code below could be placed in a separate file
```

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

Hasil 7.69 :

```
...: 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 7.69: In 18 Roza

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:

Listing 7.39: Praktek19.py

```
# In [19]: restore the class name to integer encoder
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
    inverted = label_encoder2.inverse_transform([np.argmax(prediction)])
    print("Prediction: %s, confidence: %.2f" % (inverted[0], np.max(prediction)))
```

Hasil 7.70 :

```
In [30]: 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 7.70: In 19 Roza

- 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:

Listing 7.40: Praktek20.py

```
# In [20]: grab an image (we'll just use a random training image
predict("HASYv2/hasy-data/v2-00010.png")

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

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

Hasil 7.71 :

```
In [31]: predict("HASYv2/hasy-data/v2-00010.png")
...
...
...
...
...
...
Prediction: A, confidence: 0.86
Prediction: \prod, confidence: 0.53
Prediction: \alpha, confidence: 0.94
```

Figure 7.71: In 20 Roza

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

Listing 7.41: Error Chapter7 Roza.py

```
File "<ipython-input-2-dd892d606c29>", line 2, in <module>
    random.shuffle(imgs)
```

```
NameError: name 'imgs' is not defined
```

(a) Skrinsut Error

```
In [2]: import random
...: random.shuffle(imgs)
...: split_idx = int(0.8*len(imgs))
...: train = imgs[:split_idx]
...: test = imgs[split_idx:]
Traceback (most recent call last):

File "<ipython-input-2-dd892d606c29>", line 2, in <module>
    random.shuffle(imgs)

NameError: name 'imgs' is not defined
```

Figure 7.72: Error Roza

- (b) Kode Error dan Jenis Errornya

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

Jenis Error: Images tidak terdefinisi

- (c) Penanganan

Menentukan atau memebuka file explorer dari file yang ditempatkan atau disesuaikan dengan letak filenya.

7.3 Fadila-1164072

7.3.1 Teori

Penjelasan Tugas Harian 12 (No 1-11).

1. Mengapa File Teks Harus Dilakukan Tokenizer Besera Ilustrasi Gambar :

- Tokenizer :

Difungsikan untuk membuat vektor dari text. Lebih detailnya, tokenizer merupakan langkah pertama yang diperlukan dalam banyak tugas pemrosesan bahasa alami, seperti penghitungan kata, penguraian, pemeriksaan ejaan, pembuatan corpus, dan analisis statistik teks.

- Mengapa Text Harus Dilakukan Tokenizer ? :

Text harus dilakukan tokenizer agar dapat dirubah menjadi vektor. Dari perubahan ke vektor tersebut maka data/textnya dapat dibaca oleh komputer (terkomputerisasi).

- Ilustrasi Gambar (Contoh): 7.73

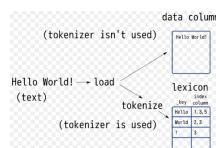


Figure 7.73: Tokenizer - fadila

2. Konsep Dasar K Fold Cross Validation Pada Dataset Komentar Youtube Pada Kode Listing 1 Beserta Dengan Ilustrasi Gambar :

- Code :

Listing 7.42: K Fold Cross Validation

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

- Penjelasan :

Untuk kejelasan dari StartifiedKFold yang dicontohkan ialah digunakan dan berisikan presentasi sampel untuk setiap kelas yang ada (youtube). Pada ilusrasinya sampel dibagi menjadi 5 dalam setiap kelas yang diproses. Kemudian sampelnya sendiri akan dimasukan kedalam class dari dataset youtube yang digunakan.

- Ilustrasi Gambar (Contoh): 7.74

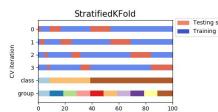


Figure 7.74: Startified K-Fold Cross - fadila

3. Jelaskan Apa Maksud Kode Program For Train Dan Test In Splits Dilengkapi Dengan Ilustrasi Gambar :

- Penjelasan :

Kode Program For Train dan Test In Splits sendiri digunakan ataupun difungsikan untuk pengujian. Pegujiannya yaitu menguji apakah setiap data pada dataset yang dieksekusi sudah di split dan tidak terjadi penumpukan pada data tersebut. Dengan tidak terjadinya penumpukan seperti hal tersebut maka setiap class nya tidak akan memunculkan id yang sama.

- Ilustrasi Kalimat (Contoh):

Dimisalkan dengan contoh dimana terdapat 6 meja dengan model / desain yang berbeda. Langkah selanjutnya dilakukan pendistribusian terhadap meja-meja tersebut kepada 3 investor (penjual) maka setiap penjual tersebut tidak akan menjual model meja yang sama dipasaran.

- Ilustrasi Gambar (Contoh Lainnya): 7.75

4. Apa Maksud Kode Program $train_content = d['CONTENT'].iloc[train_idx]$ dan $test_content = d['CONTENT'].iloc[test_idx]$. Dilengkapi Dengan Ilustrasi Gambar :

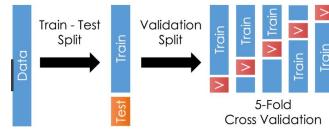


Figure 7.75: chapter-7-test-in-splits-fadila

- Penjelasan :

Maksud dari code program tersebut ialah difungsikan dalam pengambilan data pada kolom atau index CONTENT. index Content tersebut merupakan bagian dari train_idx dan test_idx. Dengan pendefinisian dan pengaplikasian index Content tersebut pada bagian training data dan test data akan dilakukan pengambilan data yang sesuai dan sama.

- Ilustrasi Kalimat (Contoh):

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.

- Ilustrasi Gambar (Contoh Lainnya): 7.76

Apabila ingin dicontohkan dengan contoh yang lain, dapat dilihat pada gambar berikut dimana contoh ini lebih sederhana hanya menjelaskan tentang penerapan d.iloc yang lebih simple sehingga apabila diterapkan dengan parameter / fungsi lain seperti content dapat lebih dimudah untuk dikerjakan (berdasarkan basicnya)

	a	b	c	d
0	1	2	3	4
1	100	200	300	400
2	1000	2000	3000	4000

Figure 7.76: Contoh Penerapan .Iloc- fadila

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

- Penjelasan :

Fungsi dari Tokenizer diatas ialah untuk melakukan vektorisasi kata tentunya. Fungsi tokenizer ini mengeksekusi jumlah data yang akan diubah sebesar 2000 kata.

Kemudian untuk `tokenizer.fit_on_texts(train_content)` digunakan untuk melakukan fit tokenizer. Fungsi tersebut hanya direalisasikan pada data train dan tidak untuk data test kemudian hanya dalam / untuk kolom content seperti yang diperlihatkan pada codingannya.

- Ilustrasi Kalimat (Contoh):

Jika dicontohkan dengan sebuah penjelasan maka bisa dikatakan apabila ada kata seperti " My Name Is Far " , " My Name Is " , " Your Name Is " , jika dibuatkan atau dijalankan dengan perintah yang dikatan sama maka akan nampak hasilnya seperti 'far': 4, 'is' : 1, 'my' : 3, 'name' : 2, 'your':5

- Ilustrasi Gambar (Contoh Lainnya): 7.77



```

from keras.preprocessing.text import Tokenizer
texts = ['My name is Far', 'My name is', 'Your name is']
tokenizer = Tokenizer()
tokenizer.fit_on_texts(texts)

```

and applying on it

```

>>> tokenizer.texts_to_matrix(texts)
<tf.Tensor: shape=(3, 10), dtype=int32, numpy=
array([[0, 0, 0, 0, 0, 0, 0, 0, 0, 4],
       [0, 0, 0, 0, 0, 0, 0, 0, 1, 0],
       [0, 0, 0, 0, 0, 0, 0, 0, 2, 5]])>

```

Figure 7.77: Tokenizer Fit On Text- fadila

6. Apa Maksud Dari Fungsi code berikut (`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 :

Dapat dikatakan bahwa maksud dari codingan diatas ialah untuk variabel `d_train_inputs` dimana akan melakukan tokenizer dari bentuk teks ke / menjadi matrix dari data `train_content` menggunakan mode matriks. Mode matriksnya sendiri yaitu `tfidf`, dan pengaplikasian mode ini juga digunakan untuk dengan variabel `d_test_inputs` untuk data test.

- Ilustrasi Code Dan Gambar (Contoh): 7.78

```

from keras.preprocessing.text import Tokenizer

tokenizer = Tokenizer(num_words = 3)
texts = ['a b b c c c', 'a b c']
tokenizer.fit_on_texts(texts)
print(tokenizer.texts_to_matrix(texts, mode='count'))

```

```
[[ 3.  2.  1.]
 [ 1.  1.  1.]]
```

Figure 7.78: Tokenizer Text To Matrix- fadila

7. Jelaskan Apa Maksud Dari Fungsi Berikut ($d_train_inputs = d_train_inputs / np.amax(np.abs(d_train_inputs))$ dan $d_test_inputs = d_test_inputs / np.amax(np.abs(d_test_inputs))$) Kemudian Dilengkapi Dengan Ilustrasi Gambar :

- Penjelasan :

Berdasarkan code diatas, menjelaskan bahwa fungsi tersebut akan membagi matrix tfidf yang sudah dieksekusi sebelumnya dengan amax. Amaxnya sendiri berfungsi dalam pengembalian maksimum array atau maksimum sepanjang sumbu. Untuk hasilnya akan dimasukan kedalam variabel d_train_inputs untuk data train dan d_test_inputs untuk data test berupa nilai absolute atautanpa ada bilangan negatif.

- Ilustrasi Code Dan Gambar (Contoh): 7.79 dan 7.80

- Contoh 1 :

```
np.max([[-50], [10]], axis=-1, initial=0)
```

```
array([ 0, 10])
```

Figure 7.79: Contoh Penerapan Np Amax- fadila

- Contoh 2 :

```
x = np.array([-1.2, 1.2])
np.absolute(x)
```

```
array([ 1.2, 1.2])
```

Figure 7.80: Contoh Penerapan Np Absolute- fadila

8. Jelaskan Apa Maksud Dari $d_train_outputs = np.utils.to_categorical(d['CLASS'].iloc[train])$ Dan $d_test_outputs = np.utils.to_categorical(d['CLASS'].iloc[test_idx])$ Dalam Kode Program Dilengkapi Dengan Ilustrasi Gambar :

- Penjelasan :

Yang dimaksudkan dari kode program tersebut dapat dijelaskan bahwa fungsinya ditujukan untuk melakukan one-hot encoding. One-hot encoding itu direalisasikan supaya bisa masuk dan digunakan pada neural network. One-hot encoding diambil dari 'CLASS' yang berarti hanya terdapat 2 neuron, yaitu dengan nilai satu nol(1,0) atau nol satu(0,1). Mengapa demikian? dikarenakan pilihannya hanya ada dua yaitu spam atau bukan spam.

- Ilustrasi Code Dan Gambar (Contoh): 7.81

```
y_train = [1, 0, 3, 4, 5, 0, 2, 1]
```

```
###
```

```
np_utils.to_categorical(y_train, num_classes=6)
```

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

Figure 7.81: Contoh Penerapan Np.utils.to categorical - fadila

9. Jelaskan Maksud Dari Fungsi Di Listing 7.2. Gambarkan Ilustrasi Neural Networknya Dari Model Kode Tersebut.

- Code :

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

- Penjelasan :

Berdasarkan code tersebut, dimaksudkan atau ditujukan untuk melakukan pemodelan dengan sequential. Sequential tersebut untuk membandingkan setiap satu larik elemen dengan cara satu persatu secara beruntun. Terdapat 512 neuron inputan dengan input shape 2000 vektor yang sudah dinormalisasi.

Selanjutnya model dilakukan aktivasi dengan fungsi 'relu', yang setelahnya terjadi pemotongan bobot sebesar 50 persen dari neuron inputan 512 (agar tidak overfitting). Pada layer output terdapat 2 neuron outputan yaitu nol(1,0) atau nol satu(0,1). Setelah semua ketentuan tersebut, dimunculkan outputan yang diaktivasi menggunakan fungsi softmax.

10. Jelaskan Maksud Dari Fungsi Di Listing 7.3. Dengan Parameter Berikut :

- Code :

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

- Penjelasan :

Berdasarkan code tersebut , dimaksudkan bahwa model yang telah dibuat akan dicompile dengan menggunakan algoritma optimisasi (adamax), fungsi loss(categorical_crossentropy), dan fungsi metrik untuk perhitungan akurasinya.

11. Apa itu Deep Learning :

- Penjelasan :

Deep learning merupakan sub bidang pembelajaran mesin yang berkaitan dengan algoritma yang terinspirasi oleh struktur dan fungsi otak yang disebut jaringan saraf tiruan.

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. Deep neural network menggunakan pemodelan matematika yang canggih untuk memproses data dengan cara yang kompleks.

- 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.

Untuk pemahaman yang lebih baik, diberikan sebuah contoh terkait dengan penjelasan perbedaan antara deep neural network dan deep learning.

- Ilustrasi Gambar (Contoh): 7.82

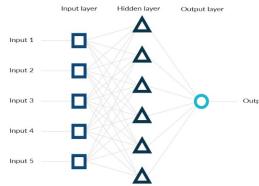


Figure 7.82: Perbedaan Deep NW Dan Deep Learn- fadila

13. Bagaimana Perhitungan Algoritma Dengan Ukuran Stride (NPM mod3+1)x(NPM mod3+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 : 7.83

Karena NPM saya 1164072 dan hasil dari $(NPM \bmod 3)+1 = 1$, maka saya menggunakan matrik kernel berukuran 1x1. Sehingga ilustrasi gambar yang digunakan adalah seperti gambar berikut. Misalkan $f(x,y)$ yang digunakan berukuran

3x3 dan kernel atau mask berukuran 1x1 masing-masing adalah sebagai berikut:

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

Keterangan = tanda * menyatakan posisi (0,0) dari kernel

Penyelesaian dari operasi konvolusi antara $f(x,y)$ dengan kernel $g(x,y)$ pada gambar di atas 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. $(-1x-1)+(2x-1)+(4x-1) = 1, -2, -4$
2. $(6x-1)+(2x-1)+(5x-1) = -6, 2, -5$
3. $(1x-1)+(2x-1)+(-3x-1) = -1, -2, 3$

Langsung bisa menjadi matriks yang dijadikan sebagai hasil, seperti pada gambar dibawah :

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

Matriks tersebut diubah dimana angka minus diubah menjadi 0

$$\begin{matrix} 1 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 3 \end{matrix}$$

Figure 7.83: Langkah Algoritma Konvolusi Berdasarkan NPM- fadila

- Plagiarisme Fadila: 7.84

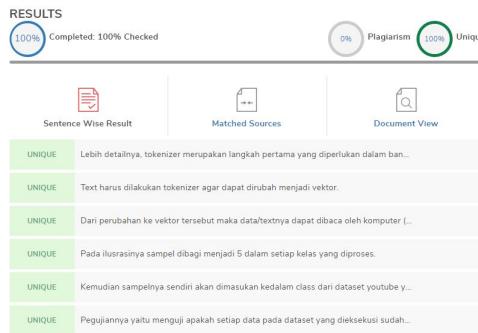


Figure 7.84: Plagiarisme- fadila

7.3.2 Praktek

Penjelasan Tugas Harian 12 (No 1-20).

1. Jelaskan Kode Program Pada Blok # In[1]. Jelaskan Arti Dari Setiap Baris Kode Yang Dibuat Dan Hasil Keluarannya Dari Komputer Sendiri.

- Code Yang Digunakan : 7.43.

Listing 7.43: File chapter-7-1-fadila.py

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

- Penjelasan Code Perbaris :

- (a) Baris Code 1 : Memasukkan / Mengimport file csv
- (b) Baris Code 2 : Memasukkan module image sebagai pil.image dari library PIL dimana PIL dapat mendukung pembukaan, pemanipulasi, dan menyimpan banyak format file gambar yang berbeda
- (c) Baris Code 3 : Memasukkan / mengimport fungsi keras.preprocessing.image

- Hasil : 7.85

```
In [2]: import csv
...: from PIL import Image as pil_image
...: import keras.preprocessing.image
Using TensorFlow backend.
```

Figure 7.85: Code Program Pada In [1] - fadila

2. Jelaskan Kode Program Pada Blok # In[2]. Jelaskan Arti Dari Setiap Baris Kode Yang Dibuat Dan Hasil Keluarannya Dari Komputer Sendiri.

- Code Yang Digunakan : 7.44.

Listing 7.44: File chapter-7-2-fadila.py

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

- Penjelasan Code Perbaris :

- Baris Code 1 : Membuat variabel imgs tanpa parameter
- Baris Code 2 : Membuat variabel classes tanpa parameter
- Baris Code 3 : Membuka file HASYv2/hasy-data-labels.csv sebagai csvfile
- Baris Code 4 : Membuat variabel csvreader yang memfungsikan pembacaan dari file csv yang dimasukkan
- Baris Code 5 : Membuat variabel i dengan parameter 0
- Baris Code 6 : Mengeksekusi baris dari pembacaan csv
- Baris Code 7 : Mengaplikasikan perintah "if" dengan ketentuan variabel i lebih besar dari angka 0, maka akan dilanjutkan ke perintah berikutnya
- Baris Code 8 : Membuat variabel img yang mengubah image menjadi bentuk array (bilangan) dari file HASYv2 yang dibuka dengan row berparameter 0.
- Baris Code 9 : Membuat variabel img tidak sama dengan nilai 255.0
- Baris Code 10 : Mendefinisikan fungsi imgs.append dimana merupakan proses melampirkan atau menggabungkan data dengan file lain atau set data yang ditentukan dengan 3 parameter yaitu row[0], row[2] dan variabel img.

- (k) Baris Code 11 : Mendefinisikan fungsi append kembali dari variabel classes dengan parameternya row[2].
- (l) Baris Code 12 : Mendefinisikan fungsi dimana i variabel i akan ditambah nilainya sehingga akan bernilai 1 (0 dengan penambahan maka akan jadi 1)

- Hasil : 7.86

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

Figure 7.86: Code Program Pada In [2] - fadila

3. Jelaskan Kode Program Pada Blok # In[3]. Jelaskan Arti Dari Setiap Baris Kode Yang Dibuat Dan Hasil Keluarannya Dari Komputer Sendiri.

- Code Yang Digunakan : 7.45.

Listing 7.45: File chapter-7-3-fadila.py

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

- Penjelasan Code Perbaris :

- (a) Baris Code 1 : Memasukkan dan memfungsikan module random
- (b) Baris Code 2 : Melakukan pengocokan pada module random dengan parameter variabelnya imgs
- (c) Baris Code 3 : Membagi dan memecah index dalam bentuk integer dengan mengkalikannilai 0,8 dan fungsi len yang akan mengembalikan jumlah item dalam datanya dari variabel imgs
- (d) Baris Code 4 : Membuat variabel train yang mengeksekusi imgs dengan pemecahan index awal pada data
- (e) Baris Code 5 : Membuat variabel test yang mengeksekusi imgs dengan pemecahan index akhir pada data

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

Figure 7.87: Code Program Pada In [3] - fadila

- Hasil : 7.87
4. Jelaskan Kode Program Pada Blok # In[4]. Jelaskan Arti Dari Setiap Baris Kode Yang Dibuat Dan Hasil Keluarannya Dari Komputer Sendiri.
- Code Yang Digunakan : 7.46.

Listing 7.46: File chapter-7-4-fadila.py

```
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)))
```

- Penjelasan Code Perbaris :

- Baris Code 1 : Memasukkan / Mengimport library numpy sebagai np
- Baris Code 2 : Membuat variabel train_input dimana mengubah input menjadi sebuah array dari np dengan menggunakan fungsi list untuk mengkoleksikan 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 Code 3 : Membuat variabel test_input dengan fungsi yang sama seperti train_input yang membedakan hanya datanya / inputan yang diproses berasal dari variabel test
- Baris Code 4 : Membuat variabel train_output dimana mengubah keluaran menjadi sebuah array dari np dengan menggunakan fungsi list untuk mengkoleksikan 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.

- (e) Baris Code 5 : Membuat variabel test_output dengan fungsi yang sama seperti train_output yang membedakan hanya datanya / inputan yang diproses berasal dari variabel test

- Hasil : 7.88

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

Figure 7.88: Code Program Pada In [4] - fadila

5. Jelaskan Kode Program Pada Blok # In[5]. Jelaskan Arti Dari Setiap Baris Kode Yang Dibuat Dan Hasil Keluarannya Dari Komputer Sendiri.

- Code Yang Digunakan : 7.47.

Listing 7.47: File chapter-7-5-fadila.py

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

- Penjelasan Code Perbaris :

- Baris Code 1 : Memasukkan modul / fungsi LabelEncoder dari sklearn.preprocessing yang digunakan untuk dapat menormalkan label dimana label encoder hanya didefinisikan dengan nilai antara 0 dan -1.
- Baris Code 2 : Memasukkan modul / fungsi OneHotEncoder dari sklearn.preprocessing yang digunakan untuk mendefinisikan fitur input dimana mengambil nilai dalam kisaran [0, maks (nilai)).

- Hasil : 7.89

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

Figure 7.89: Code Program Pada In [5] - fadila

6. Jelaskan Kode Program Pada Blok # In[6]. Jelaskan Arti Dari Setiap Baris Kode Yang Dibuat Dan Hasil Keluarannya Dari Komputer Sendiri.

- Code Yang Digunakan : 7.48.

Listing 7.48: File chapter-7-6-fadila.py

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

- Penjelasan Code Perbaris :

- (a) Baris Code 1 : Membuat variabel label_encoder dengan penerapan modul / fungsi dari LabelEncoder tanpa parameter
- (b) Baris Code 2 : Membuat variabel integer_encoded dengan penerapan fungsi label_encoder.fit_transform (ekstrasi fitur object) dari variabel classes yang akan mengembalikan beberapa data yang diubah kembali.

- Hasil : 7.90

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

Figure 7.90: Code Program Pada In [6] - fadila

7. Jelaskan Kode Program Pada Blok # In[7]. Jelaskan Arti Dari Setiap Baris Kode Yang Dibuat Dan Hasil Keluarannya Dari Komputer Sendiri.

- Code Yang Digunakan : 7.49.

Listing 7.49: File chapter-7-7-fadila.py

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

- Penjelasan Code Perbaris :

- (a) Baris Code 1 : Membuat variabel onehot_encoder yang memanggil fungsi OneHotEncoder tanpa mengembalikan matriks karena sparse=false.
- (b) Baris Code 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.
- (c) Baris Code 3 : Onehotencoding melakukan fitting pada integer_encoded.

- Hasil : 7.91

8. Jelaskan Kode Program Pada Blok # In[8]. Jelaskan Arti Dari Setiap Baris Kode Yang Dibuat Dan Hasil Keluarannya Dari Komputer Sendiri.

```

In [7]: onehot_encoder = OneHotEncoder(sparse=False)
...:     ...: integer_encoded = integer_encoded.reshape(len(integer_encoded), 1)
...:     ...: onehot_encoder.fit(integer_encoded)
D:\Anaconda\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.
1) If you want the future behaviour and silence this warning, you can specify
"categories='auto'".
2) In case you used a LabelEncoder before this OneHotEncoder to convert the categories
to integers, then you can now use the OneHotEncoder directly.
    warnings.warn(msg, FutureWarning)
Out[7]:
OneHotEncoder(categorical_features=None, categories=None,
               dtype=<class 'numpy.float64'>, handle_unknown='error',
               n_values=None, sparse=False)

```

Figure 7.91: Code Program Pada In [7] - fadila

- Code Yang Digunakan : 7.50.

Listing 7.50: File chapter-7-8-fadila.py

```

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

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

```

- Penjelasan Code Perbaris :

- Baris Code 1 : Mendefinisikan dan Membuat variabel `train_output_int` yang mengeksekusi `label_encoder` dengan mengubah nilai dari parameter variabel `train_output`.
- Baris Code 2 : Mendefinisikan dan 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 Code 3 : Mendefinisikan dan Membuat variabel `test_output_int` yang mengeksekusi `label_encoder` dengan mengubah nilai dari parameter variabel `test_output`.
- Baris Code 4 : Mendefinisikan dan Membuat variabel `test_output` yang mengeksekusi variabel `onehot_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 Code 5 : Mendefinisikan dan Membuat variabel `num_classes` untuk mengetahui jumlah class dari `label_encoder`

- (f) Baris Code 6 : Mencetak dan menampilkan hasil dari variabel num_classes menggunakan perintah print

- Hasil : 7.92

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

Figure 7.92: Code Program Pada In [8] - fadila

9. Jelaskan Kode Program Pada Blok # In[9]. Jelaskan Arti Dari Setiap Baris Kode Yang Dibuat Dan Hasil Keluarannya Dari Komputer Sendiri.

- Code Yang Digunakan : 7.51.

Listing 7.51: File chapter-7-9-fadila.py

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

- Penjelasan Code Perbaris :

- Baris Code 1 : Difungsikan pemanggilan untuk melakukan importing fungsi model sequential dari library keras.
- Baris Code 2 : Difungsikan pemanggilan untuk melakukan importing fungsi layer dense, dropout, dan flatten dari library keras.
- Baris Code 3 : Difungsikan pemanggilan untuk melakukan importing fungsi layer Conv2D dan MaxPooling2D dari library keras.

- Hasil : 7.93

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

Figure 7.93: Code Program Pada In [9] - fadila

10. Jelaskan Kode Program Pada Blok # In[10]. Jelaskan Arti Dari Setiap Baris Kode Yang Dibuat Dan Hasil Keluarannya Dari Komputer Sendiri.

- Code Yang Digunakan : 7.52.

Listing 7.52: File chapter-7-10-fadila.py

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

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

print(model.summary())
```

• Penjelasan Code Perbaris :

- (a) Baris Code 1 : Merealisasikan dan Mengfungsikan pemodelan Sequential
- (b) Baris Code 2 : Menambahkan fungsi konvolusi 2D dengan 32 filter konvolusi masing-masing berukuran 3x3 dengan algoritam activation relu sesuai dengan ketentuan
- (c) Baris Code 3 : Dilanjutkan dengan data dari train_input mulai dari baris nol dari fungsi input_shape
- (d) Baris Code 4 : Melakukan penambahan fungsi max pooling dengan matriks 2x2 terhadap data
- (e) Baris Code 5 : Dilakukan lagi penambahan Konvolusi 2D dengan 32 filter konvolusi masing-masing berukuran 3x3 dengan algoritam activation relu.
- (f) Baris Code 6 : Memfungsikan / Menambahkan kembali fungsi max pooling dengan matriks 2x2
- (g) Baris Code 7 : Memfungsikan flatten untuk mengembalikan salinan array asli dari data
- (h) Baris Code 8 : Mendefinisikan inputan dengan 1024 neuron dan menggunakan algoritma tanh untuk aktivasinya (activation)

- (i) Baris Code 9 : Mendefinisikan fungsi Dropout dimana terdiri dari pengaturan secara acak tingkat pecahan unit input ke 0 pada setiap pembaruan selama waktu pelatihan, yang membantu mencegah overfitting sebesar 50% yang dituliskan dengan nilai 0,5.
- (j) Baris Code 10 : Untuk output layer menggunakan data dari variabel num_classes dengan fungsi activationnya softmax.
- (k) Baris Code 11 : Melakukan konfigurasi dari proses pembelajaran, yang dilakukan melalui metode compile, sebelum melatih suatu model.
- (l) Baris Code 12 : Mencetak dan menampilkan representasi ringkasan model yang telah dibuat berdasarkan fungsi-fungsi yang diterapkan

- Hasil : 7.94

```

....: model.add(Flatten())
....: model.add(Dense(1024, activation='tanh'))
....: model.add(Dropout(0.5))
....: model.add(Dense(num_classes, activation='softmax'))
....:
....: model.compile(loss='categorical_crossentropy', optimizer='adam',
....: metrics=['accuracy'])
....:
....: print(model.summary())
WARNING:tensorflow:From D:\Anaconda\lib\site-packages\tensorflow\python\framework\op_def_library.py:263: colocate_with (from tensorflow.python.framework.ops) is deprecated and will be removed in a future version.
Instructions for updating:
Colocations handled automatically by placer.
WARNING:tensorflow:From D:\Anaconda\lib\site-packages\keras\backend\tensorflow_backend.py:3445: calling dropout (from tensorflow.python.ops.nn_ops) with keep_prob is deprecated and will be removed in a future version.
Instructions for updating:
Please use `rate` instead of `keep_prob`. Rate should be set to `rate = 1 - keep_prob`.
Layer (type)                 Output Shape              Param #
=================================================================
conv2d_1 (Conv2D)            (None, 30, 30, 32)      896
max_pooling2d_1 (MaxPooling2D) (None, 15, 15, 32)    0
conv2d_2 (Conv2D)            (None, 13, 13, 32)      9248
max_pooling2d_2 (MaxPooling2D) (None, 6, 6, 32)       0
flatten_1 (Flatten)          (None, 1152)            0
dense_1 (Dense)              (None, 1024)            1180672
dropout_1 (Dropout)          (None, 1024)            0
dense_2 (Dense)              (None, 369)             378225
=====
Total params: 1,569,041
Trainable params: 1,569,041
Non-trainable params: 0
=====
None

```

Figure 7.94: Code Program Pada In [10] - fadila

11. Jelaskan Kode Program Pada Blok # In[11]. Jelaskan Arti Dari Setiap Baris Kode Yang Dibuat Dan Hasil Keluarannya Dari Komputer Sendiri.

- Code Yang Digunakan : 7.53.

Listing 7.53: File chapter-7-11-fadila.py

```

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

```

- Penjelasan Code Perbaris :

- (a) Baris Code 1 : Memasukkan / Mengimport library keras.callbacks dimana digunakan dalam penulisan log untuk TensorBoard, yang memungkinkan untuk memvisualisasikan grafik dinamis dari pelatihan dan metrik pengujian.
- (b) Baris Code 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'
- Hasil : 7.95

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

Figure 7.95: Code Program Pada In [11] - fadila

12. Jelaskan Kode Program Pada Blok # In[12]. Jelaskan Arti Dari Setiap Baris Kode Yang Dibuat Dan Hasil Keluarannya Dari Komputer Sendiri.

- Code Yang Digunakan : 7.54.

Listing 7.54: File chapter-7-12-fadila.py

```
model.fit(train_input, train_output,
          batch_size=32,
          epochs=10,
          verbose=2,
          validation_split=0.2,
          callbacks=[tensorboard])

score = model.evaluate(test_input, test_output, verbose=2)
print('Test loss:', score[0])
print('Test accuracy:', score[1])
```

- Penjelasan Code Perbaris :

- (a) Baris Code 1 : Menerapkan fungsi model.fit yang didalamnya memproses train_input, train_output
- (b) Baris Code 2 : Selanjutnya pada penerapan fungsi yang sama difungsikan batch_size (jumlah sampel per pembaharuan sampel dari data yang diolah) apabila batch_sizenya tidak ditemukan maka otomatis akan dijadikan nilai 32

- (c) Baris Code 3 : Pada penerapan fungsi yang sama, difungsikan epochs dimana perulangan dari berapa kali nilai yang digunakan untuk data, dan jumlahnya ialah 10
- (d) Baris Code 4 : Mendefinisikan fungsi verbose dimana digunakan sebagai opsi untuk menghasilkan informasi logging dari data yang ditentukan dengan nilai 2
- (e) Baris Code 5 : Mendefinisikan fungsi validation_split untuk memecah nilai dari perhitungan validasinya sebesar 0,2. (Fraksi data pelatihan untuk digunakan sebagai data validasi)
- (f) Baris Code 6 : Mendefinisikan fungsi callbacks dengan parameternya yang mengeksekusi tensorboard dimana digunakan untuk Visualisasi parameter training, metrik, hiperparameter pada nilai/data yang diproses.
- (g) Baris Code 7 : Mendefinisikan variabel score dengan fungsi evaluate dari model yang ada dengan parameter test_input, test_output dan verbose=2 dimana memprediksi output untuk input yang diberikan dan kemudian menghitung fungsi metrik yang ditentukan dalam modelnya
- (h) Baris Code 8 : Mencetak score optimasi dari test dengan ketentuan nilai parameter 0
- (i) Baris Code 9 : Mencetak score akurasi dari test dengan ketentuan nilai parameter 1

- Hasil : 7.96

```

In [12]: model.fit(train_input, train_output,
...:           batch_size=32,
...:           epochs=10,
...:           verbose=2,
...:           validation_split=0.2,
...:           callbacks=[tensorboard])
...:
...: score = model.evaluate(test_input, test_output, verbose=2)
...: print('Test loss:', score[0])
...: print('Test accuracy:', score[1])
WARNING:tensorflow:From D:\Anaconda\lib\site-packages\tensorflow\python\ops\math_ops.py:3066: to_int32 (from tensorflow.python.ops.math_ops) is deprecated and will be removed in a future version.
Instructions for updating:
Use tf.cast instead.
Train on 107668 samples, validate on 26918 samples
Epoch 1/10
- 2007s - loss: 1.5334 - acc: 0.6294 - val_loss: 0.9824 - val_acc: 0.7285
Epoch 2/10
- 918s - loss: 0.9694 - acc: 0.7321 - val_loss: 0.9162 - val_acc: 0.7494
Epoch 3/10
- 562s - loss: 0.8543 - acc: 0.7556 - val_loss: 0.8883 - val_acc: 0.7530
Epoch 4/10
- 580s - loss: 0.7854 - acc: 0.7699 - val_loss: 0.8441 - val_acc: 0.7666
Epoch 5/10
- 361s - loss: 0.7364 - acc: 0.7796 - val_loss: 0.8491 - val_acc: 0.7663
Epoch 6/10
- 366s - loss: 0.6936 - acc: 0.7899 - val_loss: 0.8522 - val_acc: 0.7692
Epoch 7/10
- 366s - loss: 0.6591 - acc: 0.7962 - val_loss: 0.8588 - val_acc: 0.7668
Epoch 8/10
- 389s - loss: 0.6344 - acc: 0.8017 - val_loss: 0.8496 - val_acc: 0.7654
Epoch 9/10
- 357s - loss: 0.6076 - acc: 0.8072 - val_loss: 0.8597 - val_acc: 0.7638
Epoch 10/10
- 359s - loss: 0.5905 - acc: 0.8116 - val_loss: 0.8889 - val_acc: 0.7637
Test loss: 0.874204677150739
Test accuracy: 0.7632181175230488

```

Figure 7.96: Code Program Pada In [12] - fadila

13. Jelaskan Kode Program Pada Blok # In[13]. Jelaskan Arti Dari Setiap Baris Kode Yang Dibuat Dan Hasil Keluarannya Dari Komputer Sendiri.

- Code Yang Digunakan : 7.55.

Listing 7.55: File chapter-7-13-fadila.py

```
import time

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

- Penjelasan Code Perbaris :

- Baris Code 1 : Melakukan impor modul time dari python anaconda
- Baris Code 2 : Membuat variabel result berisikan array kosong.
- Baris Code 3 : Menggunakan convolution 2D yang dimana akan memiliki 1 atau 2 layer.
- Baris Code 4 : Mendefinisikan dan memfungsiikan dense_size dengan ukuran 128, 256, 512, 1024, 2048
- Baris Code 5 : Mendefinsikan dan memfungsiikan drop_out dengan 0, 25%, 50%, dan 75%
- Baris Code 6 : Menerapkan dan Melakukan pemodelan Sequential
- Baris Code 7 : Perlu memasukkan bentuk input apabila data tersebut merupakan layer pertama
- Baris Code 8 : Apabila tidak maka hanya menambahkan layer

- (i) Baris Code 9 : Selanjutnya, ketika penambahan layer konvolusi selesai, maka dilakukan hal yang sama dengan max pooling.
 - (j) Baris Code 10 : Kemudian meratakan atau flatten pada data dan menambahkan dense size ukuran apa pun yang berasal dari dense_size, dimana menggunakan algoritma tanh
 - (k) Baris Code 11 : Apabila dropout digunakan, dilakukan penambahan layer dropout. Untuk dropout berikut katakanlah 50%, bahwa setiap kali ia memperbarui bobot setelah setiap batch, ada peluang 50% untuk setiap bobot yang tidak akan diperbarui pada data tersebut
 - (l) Baris Code 12 : Melakukan penempatan hasil dropout di antara dua lapisan padat untuk dihindarkan dari melindunginya dari overfitting.
 - (m) Baris Code 13 : Lapisan terakhir akan selalu menjadi jumlah kelas karena itu harus, dan menggunakan softmax. Itu dikompilasi dengan cara yang sama.
 - (n) Baris Code 14 : Mengatur direktori log yang berbeda untuk TensorBoard sehingga dapat membedakan konfigurasi yang berbeda.
 - (o) Baris Code 15 : Membuat variabel start yang akan memanggil modul time atau waktu
 - (p) Baris Code 16 : Melakukan dan memfungsikan fit atau compile
 - (q) Baris Code 17 : Melakukan dan memfungsikan scoring dengan .evaluate yang akan menampilkan data loss dan accuracy dari model
 - (r) Baris Code 18 : Untuk end merupakan variabel untuk melihat waktu akhir pada saat pemodelan berhasil dilakukan.
 - (s) Baris Code 19 : Mencetak dan Menampilkan hasil dari run skrip berdasarkan fungsi-fungsi yang telah diterapkan diatas
- Hasil : 7.97

14. Jelaskan Kode Program Pada Blok # In[14]. Jelaskan Arti Dari Setiap Baris Kode Yang Dibuat Dan Hasil Keluarannya Dari Komputer Sendiri.

- Code Yang Digunakan : 7.56.

Listing 7.56: File chapter-7-14-fadila.py

```
model = Sequential()
model.add(Conv2D(32, kernel_size=(3, 3), activation='relu', input_shape=(28, 28, 1)))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Flatten())
model.add(Dense(128, activation='relu'))
model.add(Dropout(0.5))
model.add(Dense(10, activation='softmax'))
```

```

...
    model.compile(loss='categorical_crossentropy', optimizer='adam',
metrics=['accuracy'])
...
    log_dir = './logs/conv2d_%d-dense_%d-dropout_%d' % (conv2d_count,
dense_size, dropout)
    tensorboard = keras.callbacks.TensorBoard(log_dir=log_dir)
...
    start = time.time()
    model.fit(train_input, train_output, batch_size=32, epochs=10,
verbose=0, validation_split=0.2, callbacks=[tensorboard])
    score = model.evaluate(test_input, test_output, verbose=2)
    end = time.time()
    elapsed = end - start
    print("Conv2D count: %d, Dense size: %d, Dropout: %d - Loss: %.2f,
Accuracy: %.2f, Time: %.2f sec" % (conv2d_count, dense_size, dropout, score[0],
score[1], elapsed))
    results.append((conv2d_count, dense_size, dropout, score[0],
score[1], elapsed))
Conv2D count: 1, Dense size: 128, Dropout: 0.00 - Loss: 1.13, 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 7.97: Code Program Pada In [13] - fadila

```

model.add(Conv2D(32, (3, 3), activation='relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Flatten())
model.add(Dense(128, activation='tanh'))
model.add(Dropout(0.5))
model.add(Dense(num_classes, activation='softmax'))
model.compile(loss='categorical_crossentropy', optimizer='adam')
print(model.summary())

```

- Penjelasan Code Perbaris :

- Baris Code 1 : Memfungsikan dan melakukan pemodelan Sequential pada code
- Baris Code 2 : Memfungsikan dan menambahkan Convolution 2D dengan dmensi 32 untuk layer pertama,dan ukuran matriks 3x3 dengan function aktivasi yang digunakan yaitu relu dan menampilkan input_shape
- Baris Code 3 : Memfungsikan max pooling 2D dengan ukuran matriks 2x2
- Baris Code 4 : Melakukan Convolusi lagi dengan kriteria yang sama tanpa menambahkan input untuk di layer kedua, hal ini dilakukan untuk mendapatkan data yang terbaik dari yang telah diproses
- Baris Code 5 : Menambahkan fungsi Max pooling 2D dengan ukuran poolnya yaitu 2x2.
- Baris Code 6 : Mendefinisikan fungsi flatten dimana diugnakan untuk meratakan inputan yang ada dan mengembalikan salinan bilangan / array data
- Baris Code 7 : Menambahkan dense input sebanyak 128 neuron dengan menggunakan function aktivasi tanh.

- (h) Baris Code 8 : Menambahkan fungsi dropout sebanyak 50% untuk menghindari overfitting pada data yang diproses
- (i) Baris Code 9 : Memfungsiakan dense pada model untuk output, yang dimana layer berikut akan menjadi jumlah dari class yang ada.
- (j) Baris Code 10 : Melakukan compile / Mengcompile model yang telah dibuat dan didefinisikan sebelumnya (diatas)
- (k) Baris Code 11 : Menampilkan dan mencetak ringkasan dari pemodelan yang dilakukan dari berbagai fungsi yang diterapkan.

- Hasil : 7.98

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: 285,329		
Trainable params: 285,329		
Non-trainable params: 0		
<hr/>		
None		

Figure 7.98: Code Program Pada In [14] - fadila

15. Jelaskan Kode Program Pada Blok # In[15]. Jelaskan Arti Dari Setiap Baris Kode Yang Dibuat Dan Hasil Keluarannya Dari Komputer Sendiri.

- Code Yang Digunakan : 7.57.

Listing 7.57: File chapter-7-15-fadila.py

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

- Penjelasan Code Perbaris : (Cuman ada 1 perintah namun pada penulisannya ada 3 baris)
 - (a) Baris Code 1 : Memfungsiakan dan Melakukan fit (pencocokan data) dengan join data train_input dan test_input agar dapat dilakukan pelatihan untuk jaringan pada semua data yang dimiliki.
 - (b) Baris Code 2 : Memfungsiakan kembali np.concatenate (join data) dengan data dari train_output dan test_output

- (c) Baris Code 3 : Dilanjutkan dengan penerapan batch_size sebesar 32, perulangan data 10 kali (epochs) dan verbose yang digunakan sebagai opsi untuk menghasilkan informasi logging dari data yang ditentukan dengan nilai 2

- Hasil : 7.99

```
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
- 237s - 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
- 239s - loss: 0.8137 - acc: 0.7631
Epoch 8/10
- 239s - loss: 0.7988 - 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 7.99: Code Program Pada In [15] - fadila

16. Jelaskan Kode Program Pada Blok # In[16]. Jelaskan Arti Dari Setiap Baris Kode Yang Dibuat Dan Hasil Keluarannya Dari Komputer Sendiri.

- Code Yang Digunakan : 7.58.

Listing 7.58: File chapter-7-16-fadila.py

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

- Penjelasan Code Perbaris :

- (a) Baris Code 1 : Melakukan penyimpanan atau save model yang telah di latih dengan nama mathsymbols.model

- Hasil : 7.100

```
In [22]: model.save("mathsymbols.model")
```

Figure 7.100: Code Program Pada In [16] - fadila

17. Jelaskan Kode Program Pada Blok # In[17]. Jelaskan Arti Dari Setiap Baris Kode Yang Dibuat Dan Hasil Keluarannya Dari Komputer Sendiri.

- Code Yang Digunakan : 7.59.

Listing 7.59: File chapter-7-17-fadila.py

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

- Penjelasan Code Perbaris :

- (a) Baris Code 1 : Melakukan penyimpanan terhadap label enkoder (untuk membalikkan one-hot encoder) dengan nama classes.npy

- Hasil : 7.101

```
In [23]: np.save('classes.npy', label_encoder.classes_)
```

Figure 7.101: Code Program Pada In [17] - fadila

18. Jelaskan Kode Program Pada Blok # In[18]. Jelaskan Arti Dari Setiap Baris Kode Yang Dibuat Dan Hasil Keluarannya Dari Komputer Sendiri.

- Code Yang Digunakan : 7.60.

Listing 7.60: File chapter-7-18-fadila.py

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

- Penjelasan Code Perbaris :

- (a) Baris Code 1 : Memasukkan / mengimpor models dari librari Keras (keras.models)
- (b) Baris Code 2 : Membuat variabel model2 yang akan difungsikan untuk memanggil model yang telah disave sebelumnya yaitu "mathsymbols.model"
- (c) Baris Code 3 : Menampilkan dan mencetak ringkasan dari hasil pemodelan pada variabel model2

- Hasil : 7.102

```
In [24]: import keras.models
... model2 = keras.models.load_model("mathsymbols.model")
... print(model2.summary())
Layer (type)                 Output Shape              Param #
conv2d_3 (Conv2D)            (None, 30, 30, 32)       896
max_pooling2d_3 (MaxPooling2D) (None, 15, 15, 32)       0
conv2d_4 (Conv2D)            (None, 13, 13, 32)       9248
max_pooling2d_4 (MaxPooling2D) (None, 6, 6, 32)       0
flatten_2 (Flatten)          (None, 1152)            0
dense_3 (Dense)              (None, 128)             147584
dropout_2 (Dropout)          (None, 128)             0
dense_4 (Dense)              (None, 369)             47601
=====
Total params: 205,329
Trainable params: 205,329
Non-trainable params: 0
None
```

Figure 7.102: Code Program Pada In [18] - fadila

19. Jelaskan Kode Program Pada Blok # In[19]. Jelaskan Arti Dari Setiap Baris Kode Yang Dibuat Dan Hasil Keluarannya Dari Komputer Sendiri.

- Code Yang Digunakan : 7.61.

Listing 7.61: File chapter-7-19-fadila.py

```
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
    inverted = label_encoder2.inverse_transform([np.argmax(prediction)])
    print("Prediction: %s, confidence: %.2f" % (inverted[0], np.max(prediction)))
```

- Penjelasan Code Perbaris :

- Baris Code 1 : Melakukan pemanggilan fungsi LabelEncoder
- Baris Code 2 : Membuat variabel label_encoder akan memanggil class yang disave sebelumnya yaitu "classes.npy"
- Baris Code 3 : Menerapkan function predict yang akan mengubah gambar kedalam bentuk array
- Baris Code 4 : Pada fungsi prediksi dibuat variabel newimg yang akan memproses perubahan format gambar menjadi array (bilangan)
- Baris Code 5 : Untuk Variabel newimg didefinisikan tidak sama dengan nilai 255.0
- Baris Code 5 : Kemudian untuk variabel prediction akan melakukan prediksi untuk model2 dengan reshape variabel newimg dengan bentuk array 4D yaitu (1, 32, 32 dan 3).
- Baris Code 5 : Membuat variabel inverted yang akan mencari nilai tertinggi output dari hasil prediksi yang dilakukan sebelumnya
- Baris Code 6 : Menampilkan dan mencetak hasil dari variabel prediction dan inverted

- Hasil : 7.103

```

In [25]: 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 7.103: Code Program Pada In [19] - fadila

20. Jelaskan Kode Program Pada Blok # In[20]. Jelaskan Arti Dari Setiap Baris Kode Yang Dibuat Dan Hasil Keluarannya Dari Komputer Sendiri.

- Code Yang Digunakan : 7.62.

Listing 7.62: File chapter-7-20-fadila.py

```

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

```

- Penjelasan Code Perbaris :

- Baris Code 1 : Menerapkan dan Melakukan prediksi dari pelatihan dari gambar v2-00010.png
- Baris Code 2 : Menerapkan dan Melakukan prediksi dari pelatihan dari gambar v2-00500.png
- Baris Code 3 : Menerapkan dan Melakukan prediksi dari pelatihan dari gambar v2-00700.png

- Hasil : 7.104

```

In [26]: predict("HASYv2/hasy-data/v2-00010.png")
...:
...: predict("HASYv2/hasy-data/v2-00500.png")
...:
...: predict("HASYv2/hasy-data/v2-00700.png")
Prediction: \pitchfork, confidence: 0.00
Prediction: \copyright, confidence: 0.00
Prediction: 3, confidence: 0.00

```

Figure 7.104: Code Program Pada In [20] - fadila

7.3.3 Penanganan Error

Penjelasan Tugas Harian 12 (No 1-20).

1. Error 1 :

```

File "C:\Users\fadila\Documents\GitHub\keras\keras\__init__.py", line 5, in <module>
    import keras.preprocessing.image
File "C:\ProgramData\Anaconda3\lib\site-packages\keras\__init__.py", line 5, in <module>
    from . import utils
File "C:\ProgramData\Anaconda3\lib\site-packages\keras\utils\__init__.py", line 4, in <module>
    from . import conv_utils
File "C:\ProgramData\Anaconda3\lib\site-packages\keras\utils\conv_utils.py", line 5, in <module>
    from . import np
ImportError: No module named 'np'
File "C:\ProgramData\Anaconda3\lib\site-packages\keras\backend\__init__.py", line 5, in <module>
    from . import keras
ImportError: No module named 'keras'
File "C:\ProgramData\Anaconda3\lib\site-packages\keras\backend\keras_backend.py", line 5, in <module>
    from . import np
ImportError: No module named 'np'
ImportError: No module named 'tensorflow'
ModuleNotFoundError: No module named 'tensorflow'

```

Figure 7.105: Error 1 - fadila

- Screenshot Error : 7.105
- Code Error :


```
ModuleNotFoundError: No module named 'tensorflow'
```
- Penanganan Error :
 - Pertama-tama pastikan salahnya seperti apa (model dan jenis errornya)
 - Kemudian, silahkan proses error tersebut dengan cara yang sesuai
 - Berdasarkan error maka penyelesaiannya ialah melakukan instalasi terhadap module 'tensorflow' sehingga code dapat dijalankan
 - Buka Anaconda Prompt kemudian lakukan perintah seperti pada gambar berikut (conda install -c conda-forge tensorflow) : 7.106

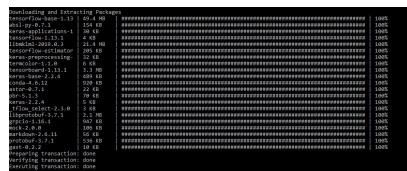


Figure 7.106: Penanganan Error 1 - fadila

- (e) Setelah melakukan perintah berikut maka silahkan jalankan kembali code maka tidak akan terjadi lagi error tersebut.

2. Error 2 :

- Screenshot Error : 7.107

```

Traceback (most recent call last):
  File "C:\Python\Python37-32\mycode\text.py", line 1, in <module>
    model.fit(np.concatenate((train_input, text_input)),
NameError: name 'model' is not defined

```

Figure 7.107: Error 2 - fadila

- Code Error :


```
NameError: name 'model' is not defined
```

- Penanganan Error :

- Pertama-tama pastikan salahnya seperti apa (model dan jenis errornya)
- Kemudian, silahkan proses error tersebut dengan cara yang sesuai
- Berdasarkan error maka penyelesaiannya ialah melakukan pendefinisian variabel model sehingga code dapat dijalankan

```
model = Sequential()
model.add(Conv2D(32, kernel_size=(3, 3), activation='relu',
               input_shape=(28, 28, 1)))
model.add(MaxPooling2D(pool_size=(2, 2), strides=(2, 2)))
model.add(Flatten())
model.add(Dense(128, activation='relu'))
model.add(Dense(10, activation='softmax'))
model.compile(loss='categorical_crossentropy', optimizer='adam',
               metrics=['accuracy'])
print(model.summary())
```

Figure 7.108: Penanganan Error 2 - fadila

- Setelah melakukan pendefinisian variabel tersebut maka silahkan jalankan kembali code maka tidak akan terjadi lagi error tersebut.

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. Tak termanfaatkan (0) b. Kurang informatif atau komplementer (0,5) c. Informatif dan komplementer (1)
7	Cara Pengacuan dan Pengutipan		1	a. Tidak baku (0) b. Kurang baku (0,5) c. Baku (1)
8	Penyusunan Daftar Pustaka	Penyusunan Daftar Pustaka	1	a. Tidak baku (0) b. Kurang baku (0,5) c. Baku (1)
9	Peristilahan dan Kebahasaan		2	a. Buruk (0) b. Baik (1) c. Cukup (2)
10	Makna Sumbangan bagi Kemajuan		4	a. Tidak ada (0) b. Kurang (1) c. Sedang (2) d. Cukup (3) e. Tinggi (4)

Figure A.1: Form nilai bagian 1.

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

Figure A.2: form nilai bagian 2.

Appendix B

FAQ

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

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

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

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

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

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

M : Pa saya bingung D : Coba lihat standar alasan

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

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

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

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

1. Tidak Mengerti : anda boleh menggunakan alasan ini jika anda sudah melakukan tahapan membaca dan meresumekan 15 jurnal. Sudah mencoba dan mempraktekkan teorinya dengan mencari di youtube dan google minimal 6 jam sehari selama 3 hari berturut-turut.
2. Bingung : anda boleh mengatakan alasan bingung setelah maksimal dalam berusaha menyelesaikan tugas bimbingan dari dosen(sudah dilakukan semua). Anda belum bisa mengatakan alasan bingung jika anda masih belum menyelesaikan tugas bimbingan dan poin nomor 1 diatas. Setelah anda menyelesaikan tugas bimbingan secara maksimal dan tahap 1 poin diatas, tapi anda masih tetap bingung maka anda boleh memakai alasan ini.

Bibliography

- [1] Joshua Eckroth. *Python Artificial Intelligence Projects for Beginners: Get up and running with Artificial Intelligence using 8 smart and exciting AI applications.* Packt Publishing Ltd, 2018.
- [2] Stuart J Russell and Peter Norvig. *Artificial intelligence: a modern approach.* Malaysia; Pearson Education Limited,, 2016.