

# Modul Praktikum

## Kecerdasan Buatan



Rolly Maulana Awangga

0410118609

Applied Bachelor of Informatics Engineering

Program Studi D4 Teknik Informatika

Applied Bachelor Program of Informatics Engineering

*Politeknik Pos Indonesia*

Bandung 2019

‘Jika Kamu tidak dapat menahan lelahnya belajar,  
Maka kamu harus sanggup menahan perihnya Kebodohan.’  
Imam Syafi’i

## **Acknowledgements**

Pertama-tama kami panjatkan puji dan syukur kepada Allah SWT yang telah memberikan rahmat dan hidayah-Nya sehingga Buku Pedoman Tingkat Akhir ini dapat diselesaikan.

## **Abstract**

Buku Pedoman ini dibuat dengan tujuan memberikan acuan, bagi mahasiswa Tingkat Akhir dan dosen Pembimbing. Pada intinya buku ini menjelaskan secara lengkap tentang Standar penggerjaan Intership dan Tugas Akhir di Program Studi D4 Teknik Informatika, dan juga mengatur mekanisme, teknik penulisan, serta penilaiannya. Dengan demikian diharapkan semua pihak yang terlibat dalam aktivitas Bimbingan Mahasiswa Tingkat Akhir berjalan lancar dan sesuai dengan standar.

# Contents

<b>1 Mengenal Kecerdasan Buatan dan Scikit-Learn</b>	<b>1</b>
1.1 Teori . . . . .	1
1.2 Instalasi . . . . .	2
1.3 Penanganan Error . . . . .	2
1.4 Andri Fajar S/1164065 . . . . .	2
1.4.1 TEORI . . . . .	2
1.4.2 Instalasi . . . . .	4
1.4.3 Mencoba Learning and predicting . . . . .	4
1.4.4 Mencoba Model Persistance . . . . .	5
1.4.5 Mencoba Conventions . . . . .	9
1.5 Penanganan Error . . . . .	12
1.6 Yusniar Nur Syarif Sidiq/1164089 . . . . .	13
1.6.1 Teori . . . . .	13
1.6.2 Instalasi . . . . .	15
1.6.3 Learning And Predicting . . . . .	16
1.6.4 Model Persistence . . . . .	18
1.6.5 Conventions . . . . .	20
1.6.6 Penanganan Error . . . . .	21
1.7 Imron Sumadireja / 1164076 . . . . .	22
1.7.1 Teori . . . . .	22
1.7.2 Instalasi . . . . .	25
1.7.2.1 Proses Instalasi Anaconda dan Library Scikit . . . . .	25
1.7.3 Mencoba Loading Dataset . . . . .	27
1.7.4 Learning and Predicting . . . . .	27
1.7.5 Model Persistence . . . . .	29
1.7.6 Conventions . . . . .	31
1.7.7 Penanganan Error . . . . .	38

<b>2 Related Works</b>	<b>40</b>
2.1 Same Topics . . . . .	40
2.1.1 Topic 1 . . . . .	40
2.1.2 Topic 2 . . . . .	40
2.2 Same Method . . . . .	40
2.2.1 Method 1 . . . . .	40
2.2.2 Method 2 . . . . .	40
2.3 Yusniar Nur Syarif Sidiq/1164089 . . . . .	41
2.3.1 Binary Classification . . . . .	41
2.3.2 Supervised Learning, Unsupervised Learning, Dan Classtering	42
2.3.3 Evaluasi Dan Akurasi . . . . .	43
2.3.4 Confusion Matrix . . . . .	44
2.3.5 Cara Kerja K-Fold Cross Validation . . . . .	45
2.3.6 Decision Tree . . . . .	46
2.3.7 Gain Dan Entropi . . . . .	46
2.4 Scikit Learn . . . . .	47
2.5 Penangan Erorr . . . . .	55
2.6 Andri Fajar Sunandhar/1164065 . . . . .	56
2.6.1 binary classification dilengkapi ilustrasi gambar . . . . .	56
2.6.2 supervised learning dan unsupervised learning dan clustering dengan ilustrasi gambar . . . . .	57
2.6.3 evaluasi dan akurasi dari buku dan disertai ilustrasi contoh dengan gambar . . . . .	60
2.6.4 bagaimana cara membuat dan membaca confusion matrix, buat confusion matrix . . . . .	60
2.6.5 bagaimana K-fold cross validation bekerja dengan gambar ilustrasi . . . . .	61
2.6.6 decision tree dengan gambar ilustrasi . . . . .	61
2.6.7 Information Gain dan entropi dengan gambar ilustrasi . . . . .	62
2.7 Imron Sumadireja / 1164076 . . . . .	63
2.7.1 Binary Classification . . . . .	63
2.7.2 Supervised Learning, Unsupervised Learning, dan Classtering	63
2.7.3 Evaluasi dan Akurasi . . . . .	64
2.7.4 Confusion Matrix . . . . .	65
2.7.5 Cara kerja K-Fold Cross Validation . . . . .	66
2.7.6 Decision Tree . . . . .	67

2.7.7	Information Gain dan Entropi . . . . .	67
2.8	Imron Sumadireja / 1164076 . . . . .	68
2.8.1	scikit-learn . . . . .	68
2.8.2	Penanganan Error . . . . .	73
2.9	scikit-learn . . . . .	74
2.10	Penanganan Error . . . . .	80
2.10.1	Error Graphviz . . . . .	80
<b>3</b>	<b>Methods</b>	<b>90</b>
3.1	The data . . . . .	90
3.2	Method 1 . . . . .	90
3.3	Method 2 . . . . .	90
3.4	Yusniar Nur Syarif Sidiq/1164089 . . . . .	90
3.4.1	Teori . . . . .	90
3.4.2	Praktek Program/Yusniar Nur Syarif Sidiq/1164089 . . . . .	93
3.4.3	Penanganan Erorr/Yusniar Nur Syarif Sidiq/1164089 . . . . .	101
3.5	Imron Sumadireja/1164076 . . . . .	101
3.5.1	Teori . . . . .	101
3.5.2	Praktik Program / Imron Sumadireja / 1164076 . . . . .	103
3.5.3	Penanganan Error / Imron Sumadireja / 1164076 . . . . .	107
3.6	Andri Fajar Sunandhar/1164065 . . . . .	108
3.6.1	Teori . . . . .	108
3.6.2	Praktek Program . . . . .	110
3.6.3	Penanganan Eror . . . . .	114
<b>4</b>	<b>Experiment and Result</b>	<b>157</b>
4.1	Experiment . . . . .	157
4.2	Result . . . . .	157
4.3	Imron Sumadireja/1164076 . . . . .	157
4.3.1	Teori . . . . .	157
4.3.2	Praktikum / Imron Sumadireja / 1164076 . . . . .	159
4.4	Penanganan Error . . . . .	170
4.4.1	Imron Sumadireja /1164076 . . . . .	170
4.5	Yusniar Nur Syarif Sidiq/1164089 . . . . .	171
4.5.1	Teori / Yusniar Nur Syarif Sidiq / 1164089 . . . . .	171
4.5.2	Praktek Program / Yusniar Nur Syarif Sidiq / 1164089 . . . . .	174
4.5.3	Penanganan Error / Yusniar Nur Syarif Sidiq / 1164089 . . . . .	183

<b>5 Conclusion</b>	<b>184</b>
5.1 Conclusion of Problems . . . . .	184
5.2 Conclusion of Method . . . . .	184
5.3 Conclusion of Experiment . . . . .	184
5.4 Conclusion of Result . . . . .	184
5.5 Yusniar Nur Syarif Sidiq/1164089 . . . . .	184
5.5.1 Pemahaman Teori / Yusniar Nur Syarif Sidiq / 1164089 . . .	184
5.5.2 Praktek Pemrograman / Yusniar Nur Syarif Sidiq / 1164089 .	188
5.5.3 Penanganan Error / Yusniar Nur Syarif Sidiq / 1164089 . . .	199
5.6 Imron Sumadireja / 1164076 . . . . .	199
5.6.1 Teori . . . . .	199
5.6.2 Praktikum / Imron Sumadireja / 1164076 . . . . .	202
<b>6 Discussion</b>	<b>210</b>
<b>7 Discussion</b>	<b>211</b>
<b>8 Discussion</b>	<b>212</b>
<b>9 Discussion</b>	<b>213</b>
<b>10 Discussion</b>	<b>214</b>
<b>11 Discussion</b>	<b>215</b>
<b>12 Discussion</b>	<b>216</b>
<b>13 Discussion</b>	<b>217</b>
<b>14 Discussion</b>	<b>218</b>
<b>A Form Penilaian Jurnal</b>	<b>219</b>
<b>B FAQ</b>	<b>222</b>
<b>Bibliography</b>	<b>224</b>

# List of Figures

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

1.29 Melihat Version . . . . .	17
1.30 Install pip. . . . .	17
1.31 Hasil Kompile. . . . .	18
1.32 Dataset. . . . .	18
1.33 Install Joblib. . . . .	22
1.34 Hasil Joblib. . . . .	22
1.35 Download Aplikasi Anaconda . . . . .	25
1.36 Proses Instalasi Aplikasi . . . . .	26
1.37 Proses Instalasi Aplikasi . . . . .	27
1.38 Proses Instalasi Aplikasi . . . . .	28
1.39 Proses Instalasi Aplikasi . . . . .	29
1.40 Proses Instalasi Aplikasi . . . . .	30
1.41 Proses Instalasi Aplikasi . . . . .	31
1.42 Proses Instalasi Aplikasi . . . . .	32
1.43 Proses Instalasi Aplikasi . . . . .	33
1.44 Proses Instalasi Aplikasi . . . . .	34
1.45 Instalasi Library . . . . .	34
1.46 Instalasi Library . . . . .	35
1.47 Instalasi Library . . . . .	35
1.48 Instalasi Library . . . . .	36
1.49 Loading dataset . . . . .	36
1.50 Error . . . . .	38
1.51 Instalasi . . . . .	39
1.52 Hasil . . . . .	39
2.1 Binary Classification. . . . .	41
2.2 Supervised Learning. . . . .	42
2.3 Unsupervised Learning. . . . .	43
2.4 Clustering. . . . .	44
2.5 Evaluasi Dan Akurasi . . . . .	45
2.6 K-Fold Cross Validation. . . . .	46
2.7 Decision Tree. . . . .	47
2.8 Gain Dan Entropi. . . . .	48
2.9 Output No 1. . . . .	48
2.10 Output No 2. . . . .	49
2.11 Output No 3. . . . .	49

2.12 Output No 4 . . . . .	50
2.13 Output No 5 . . . . .	51
2.14 Output No 6 . . . . .	52
2.15 Output No 7 . . . . .	52
2.16 Output No 8 . . . . .	52
2.17 Output No 9 . . . . .	53
2.18 Output No 10. . . . .	54
2.19 Output No 11. . . . .	55
2.20 Output No 12. . . . .	56
2.21 Graphviz Errrr. . . . .	57
2.22 Graphviz Install 1. . . . .	57
2.23 Graphviz Install 2. . . . .	58
2.24 Binary Classification . . . . .	58
2.25 Supervised Learning . . . . .	58
2.26 Unsupervised Learning . . . . .	59
2.27 Cluster . . . . .	60
2.28 Evaluasi dan Akurasi . . . . .	60
2.29 K-fold cross validation . . . . .	62
2.30 Decision Tree . . . . .	62
2.31 Information gain . . . . .	63
2.32 Binary Classification. . . . .	64
2.33 Supervised Learning. . . . .	65
2.34 Unsupervised Learning. . . . .	66
2.35 Clustering. . . . .	67
2.36 Evaluasi dan Akurasi. . . . .	68
2.37 K-Fold Cross Validation. . . . .	69
2.38 Decision Tree. . . . .	70
2.39 Information Gain dan Entropi. . . . .	71
2.40 Source Code. . . . .	71
2.41 Source Code. . . . .	72
2.42 Source Code. . . . .	72
2.43 Source Code. . . . .	73
2.44 Source Code. . . . .	73
2.45 Source Code. . . . .	74
2.46 Source Code. . . . .	74
2.47 Source Code. . . . .	75

2.48	Source Code.	75
2.49	Source Code.	76
2.50	Source Code.	76
2.51	Source Code.	77
2.52	Source Code.	77
2.53	Source Code.	78
2.54	Source Code.	78
2.55	Source Code.	79
2.56	Source Code.	79
2.57	Source Code.	80
2.58	Source Code.	80
2.59	Source Code.	81
2.60	Source Code.	82
2.61	Source Code.	82
2.62	Source Code.	83
2.63	Source Code.	83
2.64	Error.	84
2.65	Resolve.	84
2.66	Error.	84
2.67	Resolve.	84
2.68	Resolve.	85
2.69	Loading Dataset	85
2.70	Generate Binary Label	85
2.71	One-hot Encoding	86
2.72	Shuffle Rows	86
2.73	Fit Decision Tree	86
2.74	Fit Decision Tree	86
2.75	Fit Decision Tree	87
2.76	Score	87
2.77	Cross Val Score	87
2.78	Max Depth	87
2.79	Depth in Range	87
2.80	Matplotlib	88
2.81	Error Graphviz	88
2.82	install Graphviz	88
2.83	Solving Environment	88

2.84 Evaluasi Eror . . . . .	89
3.1 Random Forest. . . . .	91
3.2 Confusion Matrix. . . . .	92
3.3 Voting. . . . .	93
3.4 Aplikasi Sederhana Dengan Pandas. . . . .	94
3.5 Aplikasi Sederhana Dengan Numpy. . . . .	95
3.6 Aplikasi Sederhana Dengan Matplotlib Grafik. . . . .	96
3.7 Aplikasi Sederhana Dengan Matplotlib Diagram Batang. . . . .	97
3.8 Random Forest. . . . .	98
3.9 Random Forest. . . . .	115
3.10 Random Forest. . . . .	116
3.11 Random Forest. . . . .	117
3.12 Random Forest. . . . .	117
3.13 Random Forest. . . . .	118
3.14 Random Forest. . . . .	118
3.15 Random Forest. . . . .	119
3.16 Random Forest. . . . .	120
3.17 Random Forest. . . . .	121
3.18 Random Forest. . . . .	121
3.19 Random Forest. . . . .	122
3.20 Random Forest. . . . .	122
3.21 Random Forest. . . . .	123
3.22 Random Forest. . . . .	123
3.23 Random Forest. . . . .	123
3.24 Random Forest. . . . .	123
3.25 Confusion Matrix. . . . .	124
3.26 Confusion Matrix. . . . .	124
3.27 Confusion Matrix. . . . .	125
3.28 Confusion Matrix. . . . .	125
3.29 Confusion Matrix. . . . .	126
3.30 Klarifikasi SVM Dan Decission Tree. . . . .	126
3.31 Klarifikasi SVM Dan Decission Tree. . . . .	126
3.32 Program Cross Validation. . . . .	126
3.33 Program Cross Validation. . . . .	127
3.34 Program Cross Validation. . . . .	127

3.35 Komponen Informasi . . . . .	127
3.36 Komponen Informasi . . . . .	127
3.37 Penanganan Erorr. . . . .	128
3.38 Penanganan Erorr. . . . .	128
3.39 Random Forest. . . . .	128
3.40 Confusion Matrix. . . . .	129
3.41 Voting. . . . .	129
3.42 Hasil dari pandas. . . . .	130
3.43 Hasil dari numpy. . . . .	130
3.44 Hasil dari matplotlib. . . . .	131
3.45 Klasifikasi Random Forest1. . . . .	132
3.46 Klasifikasi Random Forest2. . . . .	133
3.47 Klasifikasi Random Forest3. . . . .	133
3.48 Klasifikasi Random Forest4. . . . .	134
3.49 Klasifikasi Random Forest5. . . . .	134
3.50 Klasifikasi Random Forest6. . . . .	135
3.51 Klasifikasi Random Forest7. . . . .	135
3.52 Klasifikasi Random Forest8. . . . .	136
3.53 Klasifikasi Random Forest9. . . . .	136
3.54 Klasifikasi Random Forest10. . . . .	137
3.55 Klasifikasi Random Forest11. . . . .	138
3.56 Klasifikasi Random Forest12. . . . .	139
3.57 Klasifikasi Random Forest13. . . . .	140
3.58 Klasifikasi Random Forest14. . . . .	140
3.59 Klasifikasi Random Forest15. . . . .	140
3.60 Klasifikasi Random Forest16. . . . .	141
3.61 Klasifikasi Random Forest17. . . . .	141
3.62 Klasifikasi Random Forest18. . . . .	141
3.63 Confusion Matrix1. . . . .	142
3.64 Confusion Matrix2. . . . .	142
3.65 Confusion Matrix3. . . . .	143
3.66 Confusion Matrix4. . . . .	143
3.67 Confusion Matrix5. . . . .	144
3.68 Decision Tree1. . . . .	144
3.69 SVM1. . . . .	144
3.70 Cross Validation1. . . . .	145

3.71 Cross Validation2 . . . . .	145
3.72 Cross Validation3 . . . . .	146
3.73 Pengamatan Komponen Informasi1 . . . . .	146
3.74 Pengamatan Komponen Informasi2 . . . . .	147
3.75 Error1 . . . . .	147
3.76 Error2 . . . . .	147
3.77 Solusi1 . . . . .	148
3.78 Random Forest . . . . .	148
3.79 Kode membaca file.csv . . . . .	148
3.80 Window Console . . . . .	148
3.81 Variable Explorer . . . . .	149
3.82 Dataset Cell . . . . .	149
3.83 Pohon Keputusan . . . . .	149
3.84 Data Testing . . . . .	150
3.85 Voting . . . . .	150
3.86 Aplikasi Pandas . . . . .	150
3.87 Hasil Pandas . . . . .	150
3.88 Aplikasi Numpy . . . . .	150
3.89 Hasil Numpy . . . . .	151
3.90 Aplikasi Matplotlib . . . . .	151
3.91 Hasil Matplotlib . . . . .	151
3.92 Gambar1 . . . . .	151
3.93 Gambar2 . . . . .	151
3.94 Gambar3 . . . . .	152
3.95 Gambar 4 . . . . .	152
3.96 Gambar 5 . . . . .	152
3.97 Gambar 6 . . . . .	152
3.98 Gambar 7 . . . . .	152
3.99 Gambar 8 . . . . .	152
3.100Gambar 9 . . . . .	152
3.101Gambar 10 . . . . .	153
3.102Gambar 11 . . . . .	153
3.103Gambar 12 . . . . .	153
3.104Gambar 13 . . . . .	153
3.105Gambar 14 . . . . .	153
3.106Gambar 15 . . . . .	153

3.107	Gambar 16 . . . . .	154
3.108	Gambar 17 . . . . .	154
3.109	Gambar 18 . . . . .	154
3.110	Memetakan ke confusion matrix . . . . .	154
3.111	Melihat hasil . . . . .	154
3.112	Melakukan Plot . . . . .	154
3.113	Plotting nama data . . . . .	155
3.114	Melakukan perintah plot . . . . .	155
3.115	SVM . . . . .	155
3.116	Decission Tree . . . . .	155
3.117	Pengecekan cross validation random forest . . . . .	155
3.118	Pengecekan cross validation decision tree . . . . .	155
3.119	Pengamatan Komponen . . . . .	156
3.120	Plot informasi . . . . .	156
3.121	Error . . . . .	156
4.1	Klasifikasi teks. . . . .	157
4.2	Klasifikasi Bunga. . . . .	158
4.3	Klasifikasi teks Youtube. . . . .	158
4.4	Bag of Words. . . . .	159
4.5	TF-IDF. . . . .	160
4.6	Data Frame. . . . .	160
4.7	Data Frame. . . . .	161
4.8	Vektorisasi dan Klasifikasi. . . . .	161
4.9	Spam dan NoSpam. . . . .	162
4.10	Vektorisasi. . . . .	162
4.11	Daftar Kata. . . . .	163
4.12	Vektorisasi. . . . .	163
4.13	Pembagian data training dan testing. . . . .	164
4.14	Pemecahan data pada table training. . . . .	164
4.15	Pemecahan data pada table testing. . . . .	164
4.16	Menampilkan kolom class. . . . .	165
4.17	Menampilkan kolom class. . . . .	166
4.18	SVM. . . . .	167
4.19	Decision Tree. . . . .	167
4.20	Confusion Matrix. . . . .	168

4.21	Cross Validation.	169
4.22	Cross Validation.	169
4.23	Komponen Informasi.	171
4.24	Screenshot error.	171
4.25	Solusi error.	171
4.26	Contoh Klasifikasi Teks YN	172
4.27	Contoh Klasifikasi Bunga YN	172
4.28	Teknik Pembelajaran Mesi Pada Teks Youtube YN	173
4.29	Bag Of Words YN	173
4.30	TF-IDF YN	174
4.31	Import DataFrame 500 Baris	174
4.32	Membagi Data Menjadi 2 DataFrame	175
4.33	Membaca File csv	175
4.34	Membagi Data Spam Dan Bukan Spam	175
4.35	Fungsi Bag Of Word	176
4.36	Melakukan Vektorisasi Pada Colom Content	176
4.37	Data Yang Sudah Di Vektorisasi	176
4.38	Pengacakan Data	177
4.39	Membagi Data Yang Sudah Di Vektorisasi	177
4.40	Vektorisasi Pada Data Training	177
4.41	Vektorisasi Pada Data Testing	178
4.42	Mengambil Data Spam Dan Bukan Dari Traning Dan Testing Data	178
4.43	Score Prediksi Dari SVM	178
4.44	Score Prediksi Dari Decision Tree	179
4.45	Import Matplotlib	179
4.46	Matrix Normalisasi	180
4.47	Score Prediksi Akurasi RF	180
4.48	Score Prediksi Akurasi DT	181
4.49	Score Prediksi Akurasi SVM	181
4.50	Pengulangan Data Vektorisasi	182
4.51	Grafik Data Pengulangan	182
4.52	Syntax Error	183
5.1	Kata-Kata Hasil Vektorisasi	185
5.2	Dataset Google	186
5.3	Vektorisasi Kata	186

5.4	Vektorisasi Document . . . . .	187
5.5	Kata-Kata Hasil Vektorisasi . . . . .	188
5.6	Vektorisasi Kata . . . . .	188
5.7	Vektorisasi Kata . . . . .	189
5.8	Vektorisasi Kata . . . . .	189
5.9	Vektorisasi Kata . . . . .	189
5.10	Vektorisasi Kata . . . . .	189
5.11	Vektorisasi Kata . . . . .	190
5.12	Vektorisasi Kata . . . . .	190
5.13	Vektorisasi Kata . . . . .	190
5.14	Vektorisasi Kata . . . . .	190
5.15	Vektorisasi Kata . . . . .	191
5.16	Vektorisasi Kata . . . . .	191
5.17	Perbandingan Kata . . . . .	191
5.18	Perbandingan Kata . . . . .	191
5.19	Perbandingan Kata . . . . .	192
5.20	Perbandingan Kata . . . . .	192
5.21	Perbandingan Kata . . . . .	192
5.22	Import Data Training. . . . .	194
5.23	Import Data Training. . . . .	195
5.24	Import Data Training. . . . .	195
5.25	Proses Instansiasi Pengocokan Data. . . . .	195
5.26	Proses Instansiasi Pembersihan Data. . . . .	196
5.27	Save Data Training. . . . .	196
5.28	Proses Instansiasi Penghapusan Temporary Training Data. . . . .	197
5.29	Hasil Infer Vector. . . . .	197
5.30	Consine Similarity. . . . .	197
5.31	Cross Validation CLF. . . . .	198
5.32	Cross Validation Random Forest. . . . .	198
5.33	Cross Validation CountVectorizer. . . . .	198
5.34	Name Error. . . . .	199
5.35	Penangan Error. . . . .	199
5.36	Ilustrasi Vektorisasi Kata. . . . .	200
5.37	Ilustrasi Vektorisasi Dataset Google. . . . .	200
5.38	Ilustrasi Vektorisasi Kata. . . . .	201
5.39	Ilustrasi Vektorisasi Dokumen. . . . .	201

5.40 Ilustrasi Skip-Gram. . . . .	202
5.41 Hasil import gensim dan logging. . . . .	202
5.42 Data vektor dari kata love. . . . .	203
5.43 Hasil similaritas. . . . .	203
5.44 Hasil similaritas. . . . .	204
5.45 Hasil similaritas. . . . .	204
5.46 Hasil similaritas. . . . .	204
5.47 Hasil similaritas. . . . .	204
5.48 Import library. . . . .	206
5.49 Menambahkan data training. . . . .	207
5.50 Menambahkan data training. . . . .	207
5.51 Menambahkan data training. . . . .	207
5.52 Random dan Clear Data. . . . .	208
5.53 Save dan Delete Temporary Training Data. . . . .	208
5.54 Infer Code. . . . .	209
5.55 Cosine Similarity. . . . .	209
A.1 Form nilai bagian 1. . . . .	220
A.2 form nilai bagian 2. . . . .	221

# Chapter 1

## Mengenal Kecerdasan Buatan dan Scikit-Learn

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

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

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

### 1.1 Teori

Praktek teori penunjang yang dikerjakan :

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

## **1.2 Instalasi**

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

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

## **1.3 Penanganan Error**

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

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

## **1.4 Andri Fajar S/1164065**

### **1.4.1 TEORI**

1. Definisi, Sejarah, Dan Perkembangan Sejarah AI

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

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

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

Supervised learning merupakan sebuah pendekatan dimana sudah terdapat data yang dilatih, dan terdapat variable yang ditargetkan sehingga tujuan dari pendekatan ini adalah mengelompokan suatu data ke data yang sudah ada. Sedangkan unsupervised learning tidak memiliki data latih, sehingga dari data yang ada, kita mengelompokan data tersebut menjadi 2 bagian atau 3 bagian dan seterusnya.

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

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

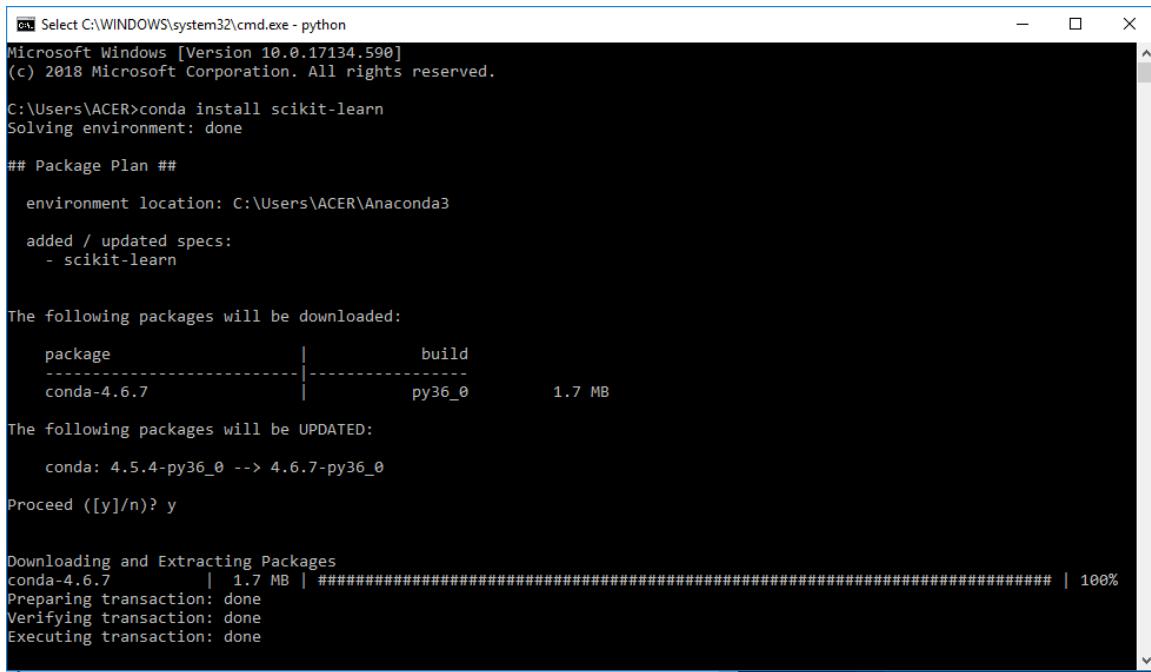
Data set adalah cabang aplikasi dari Artificial Intelligence/Kecerdasan Buatan yang fokus pada pengembangan sebuah sistem yang mampu belajar sendiri tanpa harus berulang kali di program oleh manusia.

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

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

### 1.4.2 Instalasi

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



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

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

## Package Plan ##

environment location: C:\Users\ACER\Anaconda3

added / updated specs:
- scikit-learn

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

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

Proceed ([y]/n)? y

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

Figure 1.1: conda install scikit-learn.

### 1.4.3 Mencoba Learning and predicting

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

```
C:\Users\ACER>conda --version  
conda 4.6.7  
  
C:\Users\ACER>python --version  
Python 3.6.5
```

Figure 1.2: Melihat Version.

```
C:\Users\ACER>pip install -U scikit-learn  
Collecting scikit-learn  
  Downloading https://files.pythonhosted.org/packages/ee/c8/c89ebdc0d7dbba6e6fd222daabd257da3c28a967dd7c352d4272b2e1cef6  
/scikit_learn-0.20.2-cp36-cp36m-win32.whl (4.3MB)  
    100% |██████████| 4.3MB 622kB/s  
Collecting numpy>=1.8.2 (from scikit-learn)  
  Downloading https://files.pythonhosted.org/packages/84/10/f1f99ba67aff4c3fb033571e87876ed0403114b13bc70cc125372b0c1dc6  
/numpy-1.16.1-cp36-cp36m-win32.whl (10.0MB)  
    100% |██████████| 10.0MB 863kB/s  
Collecting scipy>=0.13.3 (from scikit-learn)  
  Downloading https://files.pythonhosted.org/packages/b7/b6/eedfa8b002ffae365c3f957154a9c29cb91a8e657808749c78758f0f8110  
/scipy-1.2.1-cp36-cp36m-win32.whl (26.9MB)  
    100% |██████████| 27.0MB 89kB/s  
Installing collected packages: numpy, scipy, scikit-learn  
Successfully installed numpy-1.16.1 scikit-learn-0.20.2 scipy-1.2.1  
You are using pip version 18.1, however version 19.0.3 is available.  
You should consider upgrading via the 'python -m pip install --upgrade pip' command.
```

Figure 1.3: Install pip.

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

#### 1.4.4 Mencoba Model Persistance

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

```
C:\Users\ACER>python
Python 3.6.5 (v3.6.5:f59c0032b4, Mar 28 2018, 16:07:46) [MSC v.1900 32 bit (Intel)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> print('andri')
andri
```

Figure 1.4: Hasil Kompile.

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

Figure 1.5: Hasil Kompile.

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

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

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

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

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

Figure 1.6: Hasil Kompile.

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

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

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

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

Figure 1.7: Membuka Python

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

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

Figure 1.8: Estimator Sklearn

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

Figure 1.9: Mendefinisikan Classifier

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

Figure 1.10: Memanggil Classifier

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

Figure 1.11: Memprediksi Nilai Baru

```
>>> clf.fit(X, y)
SVC(C=1.0, cache_size=200, class_weight=None, coef0=0.0,
     decision_function_shape='ovr', degree=3, gamma='scale', kernel='rbf',
     max_iter=-1, probability=False, random_state=None, shrinking=True,
     tol=0.001, verbose=False)
>>>
```

Figure 1.12: Hasil Classifier

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

Figure 1.13: Hasil Classifier

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

Figure 1.14: Pickle Python

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

Figure 1.15: Classifier Pickle

#### 1.4.5 Mencoba Conventions

1. Import numpy as np, digunakan untuk mengimport Numpy sebagai np.

From sklearn import randomprojection artinya modul yang mengimplementasikan cara sederhana dan efisien secara komputasi untuk mengurangi dimensi data dengan memperdagangkan sejumlah akurasi yang terkendali (sebagai varians tambahan) untuk waktu pemrosesan yang lebih cepat dan ukuran model yang lebih kecil.

Pada gambar diatas dapat dijelaskan bahwa :

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

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

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

3. Dalam contoh ini, X adalah float32, yang dilemparkan ke float64 oleh fittransform (X).

4. Target regresi dilemparkan ke float64 dan target klasifikasi dipertahankan.

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

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

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

Figure 1.16: Joblib

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

Figure 1.17: Deklarasi Numpy

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

Figure 1.18: Contoh Casting

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

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

## 5. Refitting dan Memperbaharui Parameter

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

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

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

## 6. MultiClass VS MultiLabel Classifier

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

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

Figure 1.19: FitTransform

```

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

```

Figure 1.20: Regresi Yang Dilempar

```

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

```

Figure 1.21: Memperbaharui Parameter

```

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

```

Figure 1.22: MultiClass

```

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

```

Figure 1.23: MultiClass biner 2D

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

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

## 1.5 Penanganan Error

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

```

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

```

Figure 1.24: MultiLabel

```

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

```

Figure 1.25: Eror Import

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

## 1.6 Yusniar Nur Syarif Sidiq/1164089

### 1.6.1 Teori

#### 1. Sejarah Perkembangan Dan Definisi AI

Kecerdasan buatan merupakan sebuah bidang dalam ilmu computer yang begitu penting di zaman ini dan masa yang akan datang guna mewujudkan sebuah sistem computer yang begitu cerdas. Kecerdasan buatan sudah berkembang begitu pesat dalam 20 tahun terakhir seiring dengan adanya kebutuhan perangkat yang cerdas pada bidang industry dan rumah tangga.

Artificial Intelligence atau biasa disingkat dengan AI berasal dari bahasa latin yang dimana intelligence berarti saya paham. AI dimulai dari kemunculan sebuah komputer pada tahun 1940-an, akan tetapi perkembangannya

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

C:\Users\ACER>
```

Figure 1.26: Instal Library Joblib

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

Figure 1.27: Import Library Joblib

dapat dilacak pada zaman Mesir Kuno. Dalam masa ini dimana perhatian difokuskan dengan kemampuan komputer dalam mengerjakan sesuatu yang dapat dilakukan oleh manusia sehingga komputer tersebut dapat meniru kemampuan dan prilaku manusia secara cerdas.

Pada tahun 1955, Newell dan juga Simon telah mengembangkan The Logic Theorist, yaitu program AI pertama. Dimana program tersebut mempresentasikan sebuah masalah sebagai model pohon, lalu diselesaikan dengan cara memilih cabang yang akan mewujudkan kesimpulan terbenar dan tepat. Program AI tersebut berdampak sangat besar dan dapat mendaji batu loncatan yang cukup penting dalam mengembangkan bidang AI. Sekitar tahun 1956 dimana orang yang dianggap sebagai bapak AI yaitu John McCarthy telah menyelenggarakan konferensi guna menarik para ahli dibidang komputer untuk bertemu, dengan acara yang diberi nama The Dartmouth Summer Research Project On Artificial Intelligence. Dalam konferensi tersebut telah mempertemukan pendiri dan pengembang AI. Pada konferensi tersebut bapak AI John McCarthy mengusulkan definisi AI yaitu merupakan cabang dari sebuah ilmu komputer yang dapat berfokus terhadap pengembangan computer sehingga da-

pat memiliki kemampuan dan juga prilaku seperti manusia.[1].

## 2. Definisi Supervised Learning Dan Unsupervised Learning

Supervised Learning merupakan sebuah pendekatan yang dimana terdapat data dan variable yang telah ditargetkan sehingga pendekatan tersebut bertujuan untuk mengelompokkan sebuah data ke data yang sudah ada, beda dengan Unsupervised learning yang tidak mempunyai data, sehingga data yang ada harus di kelompokkan menjadi beberapa bagian.

## 3. Definisi Klasifikasi Dan Regresi

Klasifikasi merupakan sebuah kegiatan penggolongan atau pengelompokkan. Menurut kamus besar bahasa Indonesia yang dimana klasifikasi merupakan penyusunan sistem di dalam kelompok atau golongan berdasarkan kaidah atau standar yang telah ditetapkan. Regresi merupakan sebuah metode analisis statistic yang akan digunakan untuk melihat pengaruh variable.

## 4. Devinisi Dataset, Training Set, Dan Testing Set

Dataset merupakan sebuah objek yang akan mempresentasikan sebuah data dan relasinya di memory. Struktur pada dataset ini mirip dengan data yang ada di dalam database. Training set merupakan bagian dari dataset yang berperan dalam membuat prediksi atau algoritma sesuai tujuan masing masing. Testing set merupakan bagian dari dataset yang akan di tes guna melihat keakuratan atau ketepatan datanya.

### 1.6.2 Instalasi

Untuk melakukan instalasi Anaconda ikuti tutorial berikut.

- Memberikan perintah

```
conda install scikit-learn
```

di cmd, lihat pada figure 1.28

- Melihat versinya dengan memberikan perintah

```
conda --version
```

dan

```
python --version
```

lihat pada figure 1.29

- Install pip, lihat pada figure 1.30
- Hasil Kompile, lihat pada figure 1.31

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

## Package Plan ##

environment location: C:\Users\NS\Anaconda3

added / updated specs:
- scikit-learn

The following packages will be UPDATED:

  conda: 4.5.4-py36_0 --> 4.6.7-py36_0

Proceed ([y]/n)? y

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

Figure 1.28: conda install scikit-learn.

Dataset adalah objek seperti kamus yang menyimpan semua data dan berupa metadata tentang data. Data tersebut disimpan di .data anggota yang merupakan array. Misalnya dalam kasus dataset digit, memberikan akses ke fitur yang dapat digunakan untuk mengklarifikasi sempel digit, lihat pada figure 1.32.

### 1.6.3 Learning And Predicting

- Pada codingan di bawah ini yaitu terjadinya pengambilan package datasets dari Scikit-Learn.

```
from sklearn import datasets
```

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

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

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

Figure 1.29: Melihat Version.

```
C:\Users\NS>pip install -U scikit-learn
Collecting scikit-learn
  Using cached https://files.pythonhosted.org/packages/ee/c8/c89ebdc0d7dbba6e6fd222daabd257da3c28a967dd7c352d4272b2e1cef6/scikit_learn-0.20.2-cp36-cp36m-win32.whl
Requirement not upgraded as not directly required: numpy>=1.8.2 in c:\users\ns\anaconda3\lib\site-packages (from scikit-learn) (1.14.3)
Requirement not upgraded as not directly required: scipy>=0.13.3 in c:\users\ns\anaconda3\lib\site-packages (from scikit-learn) (1.1.0)
distributed 1.21.8 requires msgpack, which is not installed.
Installing collected packages: scikit-learn
  Found existing installation: scikit-learn 0.19.1
    Uninstalling scikit-learn-0.19.1:
      Successfully uninstalled scikit-learn-0.19.1
Successfully installed scikit-learn-0.20.2
You are using pip version 10.0.1, however version 19.0.3 is available.
You should consider upgrading via the 'python -m pip install --upgrade pip' command.
```

Figure 1.30: Install pip.

- Pada codingan di bawah menjelaskan bahwa akan mengambil data iris dari package datasets.

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

- Pada codingan di bawah menjelaskan bahwa akan mengambil data digits dari package datasets.

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

- Pada codingan di bawah menjelaskan bahwa akan melakukan import sebuah Support Vector Machine (SVM) yang merupakan algoritma classification yang akan diambil dari Scikit-Learn.

```
from sklearn import svm
```

- Gamma tersebut akan didefinisikan secara manual, maka codingan akan terlihat seperti berikut

```
clf = svm.SVC(gamma=0.0001, C=100.)
```

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

Figure 1.31: Hasil Kompile.

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

Figure 1.32: Dataset.

- Untuk melihat angka prediksi datu data digits maka codingannya seperti berikut

```
clf.fit(digits.data[:-1], digits.target[:-1])
clf.predict(digits.data[-1:])
```

#### 1.6.4 Model Persistence

- Pada codingan di bawah menjelaskan bahwa akan melakukan import sebuah Support Vector Machine (SVM) yang merupakan algoritma classification yang akan diambil dari Scikit-Learn.

```
from sklearn import svm
```

- Pada codingan di bawah menjelaskan bahwa akan melakukan import datasets yang akan diambil dari Scikit-Learn

```
from sklearn import datasets
```

- Codingan dibawah berfungsi untuk mendeklarasikan value pada clf yang berisi gamma

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

- Dimana codingan dibawah ini berfungsi untuk mengambil data iris dari datasets

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

- Codingan dibawah menjelaskan bahwa nilai X sebagai data iris dan nilai y sebagai target iris

```
X, y = iris.data, iris.target
```

- Codingan dibawah berfungsi untuk melakukan pengujian classifier

```
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)
```

- Pada codingan dibawah dijelaskan akan melakukan import pickle. Pickle digunakan sebagai serialisasi dan de-serialisasi struktur objek python. Pickle loads digunakan untuk memuat data pickle dari string byte.

```
import pickle
s = pickle.dumps(clf)
clf2 = pickle.loads(s)
```

- Codingan dibawah adalah pengujian dengan nilai baru dan target asumsinya adalah (0,1) dan hasilnya akan berbentuk array.

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

### 1.6.5 Conventions

- Melakukan import numpy sebagai np

```
import numpy as np
```

- Melakukan import project secara random dari Sciki-Learn

```
from sklearn import random_projection
```

- Digunakan untuk menginisialisasikan random number generator

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

- Melakukan random value antara 10 sampai 2000

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

- Mendeklarasikan X sebagai float32

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

```
X.dtype
```

```
dtype('float32')
```

- Melemparkan nilai X yaitu float32 ke float64 dengan menggunakan transform

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

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

```
X_new.dtype
```

```
dtype('float64')
```

- Melakukan import datasets dari Sciki-Learn

```
from sklearn import datasets
```

- Melakukan import SVC dari sciki-learn yang ada pada svm

```
from sklearn.svm import SVC
```

- Mengambil datasets iris

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

- Mendeklarasikan gamma dengan scale pada SVC dalam variabel clf

```
clf = SVC(gamma='scale')
```

- Membaca data iris dan target iris dari variabel clf

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

- Melakukan pengujian prediksi dengan asumsi (0,3)

```
list(clf.predict(iris.data[:3]))
[0, 0, 0]
```

- Membaca data iris dan nama target iris dari target iris pada variabel clf

```
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)
```

- Melakukan pengujian prediksi dengan asumsi (0,3)

```
list(clf.predict(iris.data[:3]))
['setosa', 'setosa', 'setosa']
```

### 1.6.6 Penanganan Error

- Dimana Erorr Tersebut dapat di lihat pada figure ??
- Codingan yang eror tersebut adalah

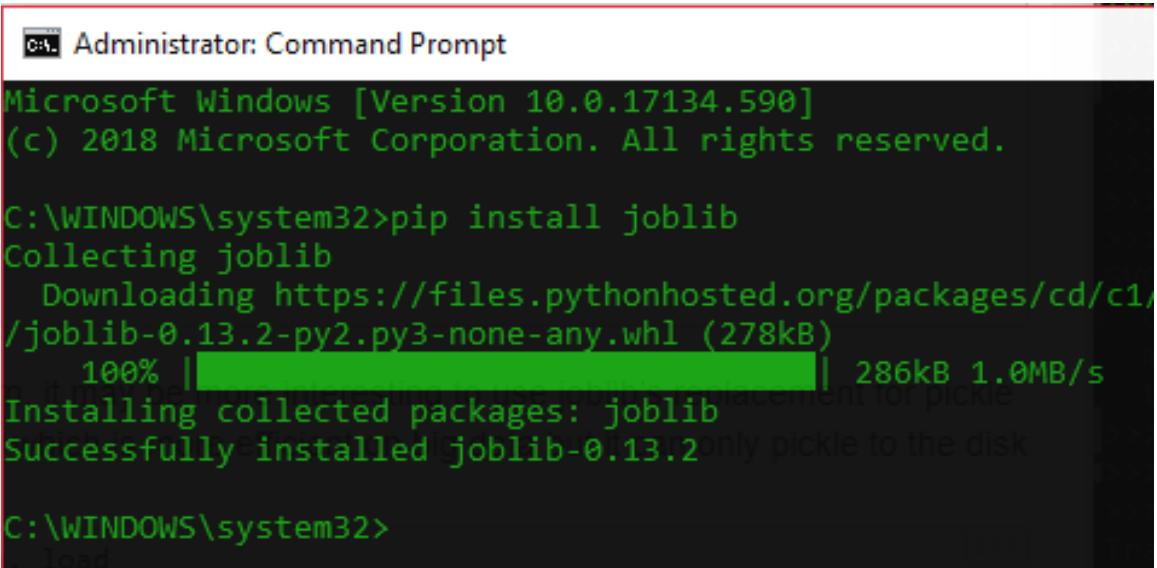
```
from joblib import dump, load
```

- Hal ini dikarenakan belum terinstallnya file joblib di dalam pc anda. Untuk menginstallnya cukup berikan perintah codingan dibawah, untuk lebih jelasnya liat pada figure 1.33

```
pip install joblib
```

- maka hasilnya akan nampak sepeti pada figure 1.34

=====



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

C:\WINDOWS\system32>pip install joblib
Collecting joblib
  Downloading https://files.pythonhosted.org/packages/cd/c1/joblib-0.13.2-py2.py3-none-any.whl (278kB)
    100% |████████████████████████████████| 286kB 1.0MB/s
Installing collected packages: joblib
Successfully installed joblib-0.13.2

C:\WINDOWS\system32>
```

Figure 1.33: Install Joblib.

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

Figure 1.34: Hasil Joblib.

## 1.7 Imron Sumadireja / 1164076

### 1.7.1 Teori

#### 1. Pengertian

Kecerdasan Buatan Artificial Intelligence merupakan salah satu bagian dari ilmu komputer yang mempelajari cara membuat mesin komputer dapat melakukan pekerjaan sebaik bahkan lebih baik dari yang dilakukan oleh manusia. Agar mesin dapat bekerja layaknya manusia maka perlu diberi bekal pengetahuan, sehingga mempunyai kemampuan untuk menalar. Menurut para ahli kecerdasan buatan seperti berikut:

- H. A. Simon: Kecerdasan buatan Artificial Intelligence merupakan kawasan

penelitian, aplikasi dan instruksi yang terkait dengan pemrograman komputer untuk melakukan sesuatu hal yang dalam pandangan manusia adalah cerdas.

- Rich and Knight: Kecerdasan buatan Artificial Intelligence merupakan sebuah studi tentang bagaimana membuat komputer melakukan hal-hal yang pada saat ini dapat dilakukan lebih baik oleh manusia.

## 2. Sejarah dan Perkembangan

Kata intelligence berasal dari bahasa latin intelligo yang memiliki arti saya paham. Arti dasar dari intelligence merupakan kemampuan untuk memahami dan melakukan aksi. Area Kecerdasan Buatan Artificial Intelligence, bermula pada saat kemunculan komputer sekitar tahun 1940-an, walaupun sejarah perkembangannya dapat dilacak sejak zaman Mesir kuno. Pada masa saat ini, perhatian difokuskan pada kemampuan komputer mengerjakan sesuatu yang dapat dilakukan oleh manusia. Dalam hal ini, komputer tersebut dapat meniru kemampuan kecerdasan dan perilaku manusia dengan akurasi yang cukup baik [5].

Pada akhir tahun 1955, Newell dan Simon mengembangkan The Logic Theorist, program AI pertama, program ini merepresentasikan masalah sebagai model pohon, lalu penyelesaiannya dengan memilih cabang yang akan menghasilkan kesimpulan yang paling benar. Pada tahun 1956 John McCarthy dari Massachusetts Institute of Technology dianggap sebagai bapak AI, menyelenggarakan konferensi untuk menarik para ahli komputer bertemu, dengan nama kegiatan The Dartmouth Summer Research Project on Artificial Intelligence. Konferensi Dartmouth itu mempertemukkan para pendiri AI, dan bertugas untuk meletakkan dasar bagi masa depan pengembangan dan penelitian AI. John McCarthy saat itu mengusulkan definisi AI adalah AI merupakan cabang dari ilmu komputer yang berfokus pada pengembangan komputer untuk dapat memiliki kemampuan dan berprilaku seperti manusia[2].

## 3. Supervised Learning dan Unsupervised Learning

Supervised Learning merupakan suatu 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. Sebagai contoh, ketika Anda memiliki sejumlah buku yang sudah dibeli dengan

beberapa kategori. Misalnya, kategori buku akademik, dan buku novel. Selanjutnya Anda membeli sejumlah buku baru, maka Anda harus mengidentifikasi buku tersebut, dan memasukannya dalam kategori yang sudah ada.

Unsupervised Learning merupakan suatu pendekatan namun tidak memiliki data yang dilatih, sehingga dari data yang ada, kita dapat mengelompokan data tersebut menjadi 2 bagian atau 3 bagian dan seterusnya. Sebagai contoh, Anda belum pernah membeli sejumlah buku, suatu hari Anda membeli sejumlah buku dan ingin membaginya kedalam beberapa kategori agar mudah dicari. Anda akan mengidentifikasi buku mana yang mirip. Dalam hal ini, kita memilih buku berdasarkan isinya.

#### 4. Klasifikasi dan Regresi

Klasifikasi merupakan penempatan objek-objek ke salah satu dari beberapa kategori yang telah ditentukan sebelumnya. Klasifikasi banyak digunakan untuk memprediksi kelas pada suatu label atau atribut tertentu, yaitu dengan mengklasifikasi data membangun model berdasarkan training set dan nilai-nilai dalam mengklasifikasikan data yang baru. Regresi dibedakan menjadi 2, diantaranya regresi linear dan regresi nonlinear.

- Regresi Linear Regresi Linear merupakan bentuk hubungan di mana variabel bebas x maupun variabel tergantung y sebagai faktor yang berpangkat satu.
- Regresi Nonlinear Regresi Nonlinear merupakan bentuk hubungan atau fungsi di mana variabel x dan variabel tidak bebas y dapat berfungsi sebagai faktor atau variabel dengan pangkat tertentu.

#### 5. Data set, Training set, dan Testing set

Untuk melakukan data set, training set, dan testing set diperlukan beberapa langkah, diantaranya:

- Membuat model atau mesin untuk memeriksa data,
- Membuat model atau mesin belajar dari kesalahannya,
- Membuat kesimpulan tentang seberapa baik kinerja model atau mesin tersebut.

(a) Data set

Data set ini mencakup sekumpulan contoh input yang modelnya akan cocok atau dilatih dengan menyesuaikan parameter.

(b) Training set

Training set diperlukan oleh model atau mesin agar dapat dilatih. Dengan menghitung kerugian tingkat kesalahan yang dilakukan model atau mesin menghasilkan pada set validasi pada titik tertentu, agar kita tahu seberapa akuratnya. Selanjutnya, model akan menyesuaikan parameteranya berdasarkan hasil evaluasi yang sering pada training set ini.

(c) Testing set

Testing set sangat penting untuk menguji generalisasi model atau mesin. Dengan testing set ini, kita bisa mendapatkan akurasi kinerja model atau mesin.

## 1.7.2 Instalasi

### 1.7.2.1 Proses Instalasi Anaconda dan Library Scikit

1. Pertama kita unduh terlebih dahulu aplikasi anaconda, seperti gambar berikut

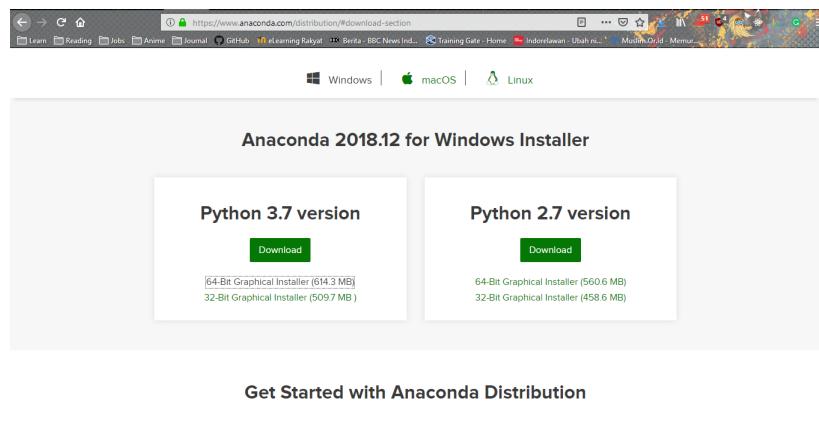


Figure 1.35: Download Aplikasi Anaconda

2. Setelah di unduh, selanjutnya buka aplikasi tersebut. Lalu klik next untuk melanjutkan.
3. Lalu klik I Agree untuk melanjutkan.

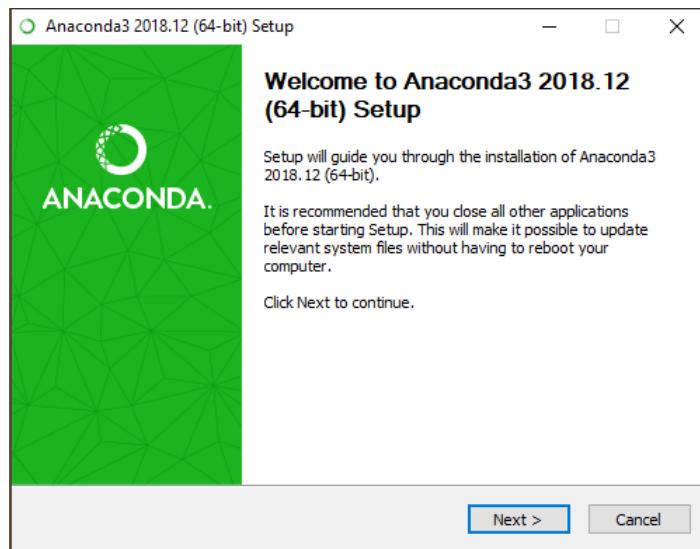


Figure 1.36: Proses Instalasi Aplikasi

4. Selanjutnya pilih Just me agar aplikasi tersebut hanya dapat digunakan oleh user yang login pada laptop tersebut.
5. Lalu tentukan direktori penyimpanan file tersebut
6. Selanjutnya akan muncul pop up box tentang advance installation options, ceklis keduanya.
7. Tunggu hingga proses install selesai
8. Setelah proses instalasi selesai, klik next
9. Pada bagian selanjutnya akan muncul box dengan memberikan pilihan untuk install VS Code, jika tidak klik skip.
10. Setelah selesai, klik finish
11. Setelah proses instalasi selesai, selanjutnya buka cmd dan ketikan seperti berikut.
12. Selanjutnya ketikan perintah berikut untuk mengunduh library scikit
13. Jika sudah berhasil selanjutnya, ketikan perintah seperti gambar berikut untuk malakukan cek versi conda dan python
14. Mencoba dan mengcompile source code, hasilnya seperti berikut

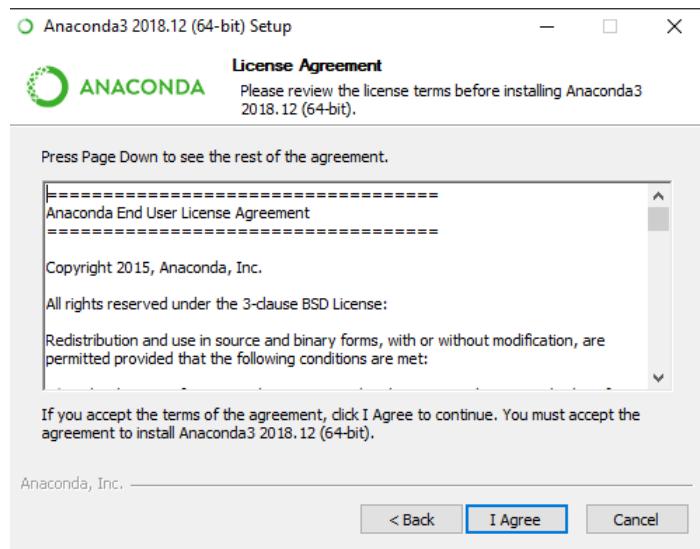


Figure 1.37: Proses Instalasi Aplikasi

### 1.7.3 Mencoba Loading Dataset

1. Berikut source code yang menjelaskan tentang loading dataset. Pada baris pertama code tersebut berfungsi untuk import library datasets dari sklearn. Baris kedua berfungsi untuk menampilkan data secara berurutan. Baris ketiga untuk menampilkan data tersebut berupa angka dan baris keempat untuk menampilkan data tersebut.

### 1.7.4 Learning and Predicting

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

- import datasets dari package sklearn
- loading dataset iris
- loading dataset digits
- menampilkan data dari loading dataset digits

```
>>> from sklearn import svm  
>>> clf = svm.SVC(gamma=0.001, C=100.)  
>>> clf.fit(digits.data[:-1], digits.target[:-1])
```

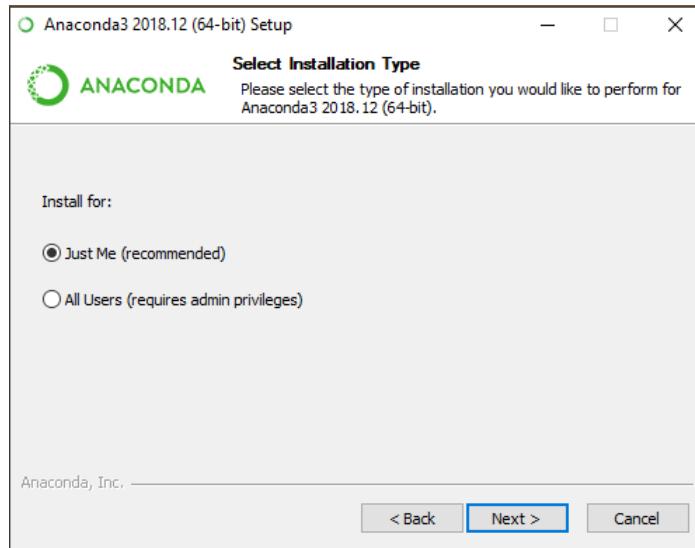


Figure 1.38: Proses Instalasi Aplikasi

- Baris tersebut menjelaskan bahwa dalam project ini kita menggunakan source dari sklearn dengan mengambil/import dari svm
- classifier svc dengan atribut gamma dan c
- classifier tersebut akan dijalankan dengan menggunakan metode fit

```
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)
```

- hasilnya seperti di atas

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

- classifier predict loading data digits

```
array([8])
```

- hasilnya seperti di atas

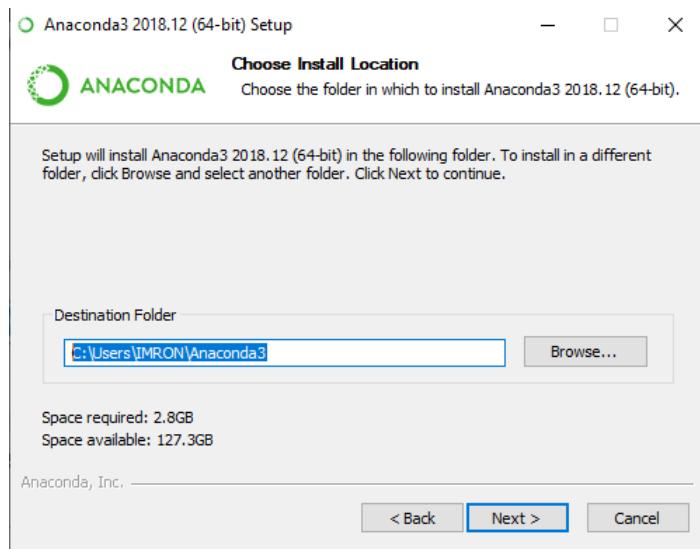


Figure 1.39: Proses Instalasi Aplikasi

### 1.7.5 Model Persistence

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

- import svm dari package sklearn
- importt datasets dari package sklearn
- classifier svc dengan atribut gamma
- loading dataset iris
- parameter x dan y dengan key iris data dan iris target
- classifier akan dijalankan menggunakan metode fit

```
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)
```

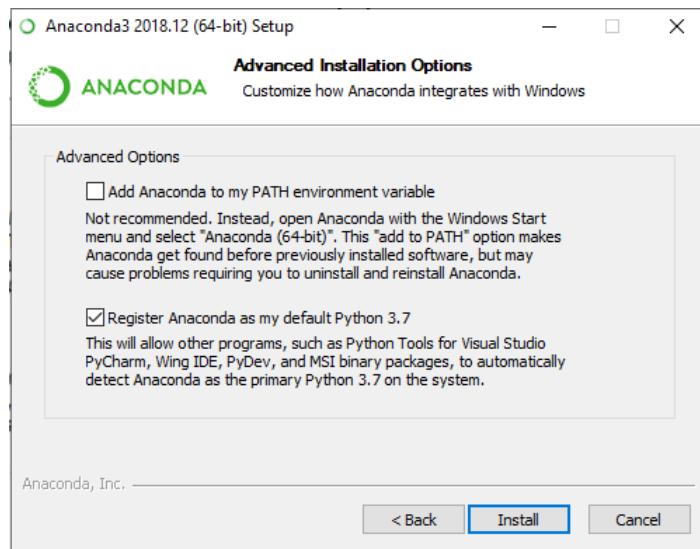


Figure 1.40: Proses Instalasi Aplikasi

- hasilnya seperti di atas

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

- import package pickle
- pickle akan melakukan dumps pada classifier
- classifier2 akan mengambil data pada classifier pertama
- classifier2 akan memprediksi hasilnya dengan menggunakan syntax python

```
array([0])
```

- hasilnya seperti diatas

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

- parameter y dengan atribut 0

0

- hasil seperti di atas

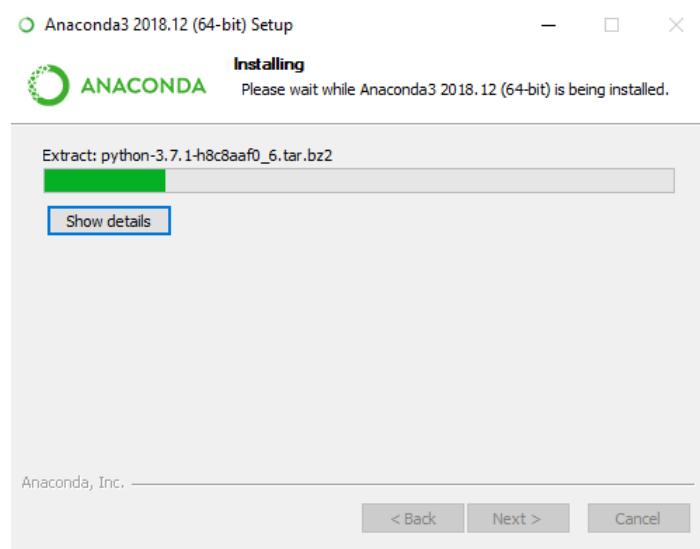


Figure 1.41: Proses Instalasi Aplikasi

### 1.7.6 Conventions

```
>>> 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
```

- import numpy dengan alias np
- import random projection pada package sklearn
- rng parameter dan akan melakukan proses random dalam menentukan hasil
- parameter x memiliki rand dengan nilai 10, 2000
- parameter x dengan numpy array akan memunculkan kata float32 pada hasil terakhir

```
dtype('float32')
```

- hasilnya seperti diatas

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

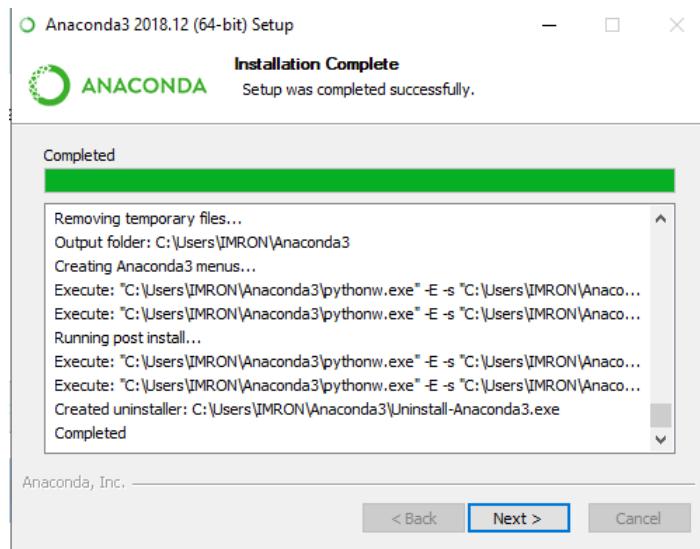


Figure 1.42: Proses Instalasi Aplikasi

- transformer parameter yang di gunakan untuk melakukan pencarian data dengan gaussianrandomprojection
- x new parameter dan akan dijalankan dengan menggunakan metode fit
- x dtype akan menampilkan hasilnya

```
dtype('float64')
```

- hasilnya seperti di atas

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

- import datasets pada package sklearn
- import svc pada package sklearn
- loading dataset iris
- classifier dengan atribut gamma
- classifier dengan metode fit pada key iris dan target.

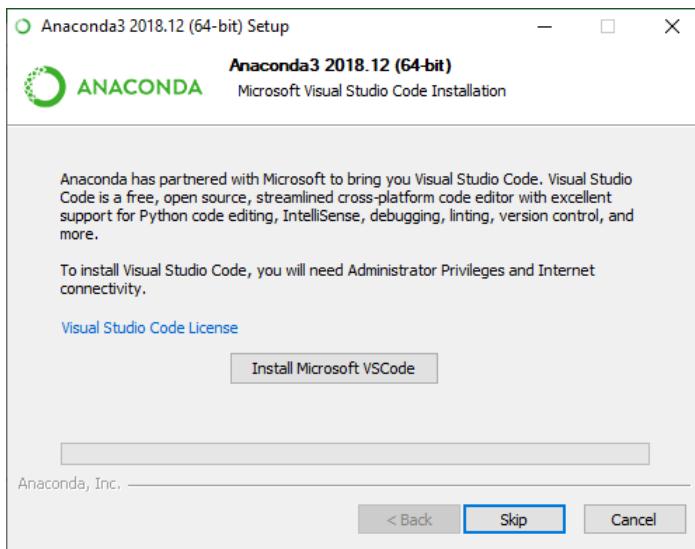


Figure 1.43: Proses Instalasi Aplikasi

```
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)
```

- hasilnya seperti di atas

```
>>> list(clf.predict(iris.data[:3]))
```

- untuk list classifier predict pada loading dataset iris

```
[0, 0, 0]
```

- hasilnya seperti diatas

```
>>> clf.fit(iris.data, iris.target_names[iris.target])
```

- classifier dengan menggunakan metode fit dan key data dan 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)
```

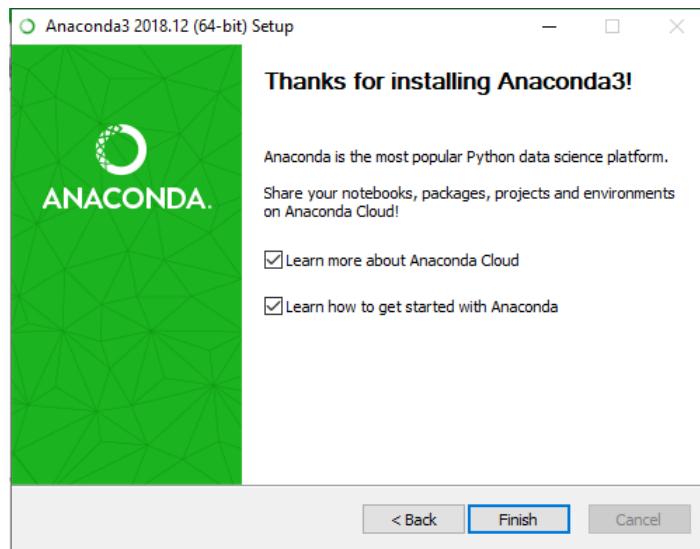


Figure 1.44: Proses Instalasi Aplikasi

```
Administrator: Command Prompt
(c) 2018 Microsoft Corporation. All rights reserved.

C:\WINDOWS\system32>pip install -U scikit-learn
Collecting scikit-learn
  Using cached https://files.pythonhosted.org/packages/ee/c8/c89ebdc0d7dbba6e6fd222daabd257da3c28a967dd7c352d4272b2
  e1cef0/scikit_learn-0.20.2-cp36-cp36m-win32.whl
Requirement not upgraded as not directly required: scipy>=0.13.3 in c:\programdata\anaconda3\lib\site-packages (from scikit-learn) (1.1.0)
Requirement not upgraded as not directly required: numpy>=1.8.2 in c:\programdata\anaconda3\lib\site-packages (from scikit-learn) (1.14.3)
distributed 1.21.8 requires msgpack, which is not installed.
Installing collected packages: scikit-learn
  Found existing installation: scikit-learn 0.19.1
    Uninstalling scikit-learn-0.19.1:
      Successfully uninstalled scikit-learn-0.19.1
Successfully installed scikit-learn-0.20.2
You are using pip version 10.0.1, however version 19.0.3 is available.
You should consider upgrading via the 'python -m pip install --upgrade pip' command.
```

Figure 1.45: Instalasi Library

- hasilnya seperti diatas

```
>>> list(clf.predict(iris.data[:3]))
```

- list untuk classifier pada predict loading data iris

```
['setosa', 'setosa', 'setosa']
```

- hasilnya seperti diatas

```
>>> import numpy as np
>>> from sklearn.svm import SVC
>>> rng = np.random.RandomState(0)
>>> X = rng.rand(100, 10)
```

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

## Package Plan ##

environment location: C:\ProgramData\Anaconda3

added / updated specs:
- scikit-learn

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

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

Proceed ([y]/n)? y

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

Figure 1.46: Instalasi Library

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

C:\WINDOWS\system32>conda --version
conda 4.6.7
```

Figure 1.47: Instalasi Library

```
>>> y = rng.binomial(1, 0.5, 100)
>>> X_test = rng.rand(5, 10)
>>> clf = SVC()
>>> clf.set_params(kernel='linear').fit(X, y)
```

- import numpy alias np
- import svc dari package sklearn
- rng parameter untuk mencari data pada atribut randomstate
- X memiliki jangakaun rand 100, 10
- y memiliki binominal 5,10
- x memiliki rand 5,10
- classifier dengan atribut svc
- classifier parameter dengan atribut linear menggunakan metode fit

```

Administrator: Command Prompt - python
C:\WINDOWS\system32>python
Python 3.6.5 |Anaconda, Inc.| (default, Mar 29 2018, 13:23:52) [MSC v.1900 32 bit (Intel)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>>
>>> iris = datasets.load_iris()
>>> digits = datasets.load_digits()
>>> print(digits.data)
[[ 0.  0.  5. ... 0.  0.  0.]
 [ 0.  0.  0. ... 10. 0.  0.]
 [ 0.  0.  0. ... 16. 9.  0.]
 ...
 [ 0.  0.  1. ... 6.  0.  0.]
 [ 0.  0.  2. ... 12. 0.  0.]
 [ 0.  0.  10. ... 12. 1.  0.]]
>>> digits.target
array([0, 1, 2, ..., 8, 9, 8])
>>> digits.images[0]
array([[ 0.,  0.,  5., 13.,  9.,  1.,  0.,  0.],
       [ 0.,  0., 13., 15., 10., 15.,  5.,  0.],
       [ 0.,  3., 15.,  2.,  0., 11.,  8.,  0.],
       [ 0.,  4., 12.,  0.,  0.,  8.,  8.,  0.],
       [ 0.,  5.,  8.,  0.,  0.,  9.,  8.,  0.],
       [ 0.,  4., 11.,  0.,  1., 12.,  7.,  0.],
       [ 0.,  2., 14.,  5., 10., 12.,  0.,  0.],
       [ 0.,  0., 13., 10.,  0.,  0.,  0.,  0.]])
>>> 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)

```

Figure 1.48: Instalasi Library

```

C:\WINDOWS\system32>python
Python 3.6.5 |Anaconda, Inc.| (default, Mar 29 2018, 13:23:52) [MSC v.1900 32 bit (Intel)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>>
>>> from sklearn import datasets
>>> iris = datasets.load_iris()
>>> digits = datasets.load_digits()
>>> print(digits.data)
[[ 0.  0.  5. ... 0.  0.  0.]
 [ 0.  0.  0. ... 10. 0.  0.]
 [ 0.  0.  0. ... 16. 9.  0.]
 ...
 [ 0.  0.  1. ... 6.  0.  0.]
 [ 0.  0.  2. ... 12. 0.  0.]
 [ 0.  0.  10. ... 12. 1.  0.]]
>>> digits.target
array([0, 1, 2, ..., 8, 9, 8])
>>>

```

Figure 1.49: Loading dataset

```

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)

```

- hasilnya seperti di atas

```
>>> clf.predict(X_test)
```

- classifier untuk memprediksi nilai x

```
array([1, 0, 1, 1, 0])
```

- hasilnya seperti di atas

```
>>> clf.set_params(kernel='rbf', gamma='scale').fit(X, y)
```

- classifier parameters dengan atribut gamma menggunakan metode fit

```
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)
```

- hasilnya seperti di atas

```
>>> clf.predict(X_test)
```

- classifier prediksi dari x

```
array([1, 0, 1, 1, 0])
```

- hasilnya seperti di atas

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

- import svc dari package sklearn
- import OneVsRentClassifier dari package sklearn
- import LabelBinarizer dari package sklearn
- atribut x
- atribut y
- classifier dengan atribut OneVsRestClassifier dan estimator svc

```
array([[1, 0, 0],  
       [1, 0, 0],  
       [0, 1, 0],  
       [0, 0, 0],  
       [0, 0, 0]])
```

- hasilnya seperti di atas

```
>>> 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)
```

- import MultiLabelBinarizer dari package sklearn
- atribut y
- atribut y akan dijankan dengan metode fit pada tampilan MultiLabelBinarizer
- classifier dengan metode fit pada x dan y untuk memprediksi

```
array([[1, 1, 0, 0, 0],  
       [1, 0, 1, 0, 0],  
       [0, 1, 0, 1, 0],  
       [1, 0, 1, 0, 0],  
       [1, 0, 1, 0, 0]])
```

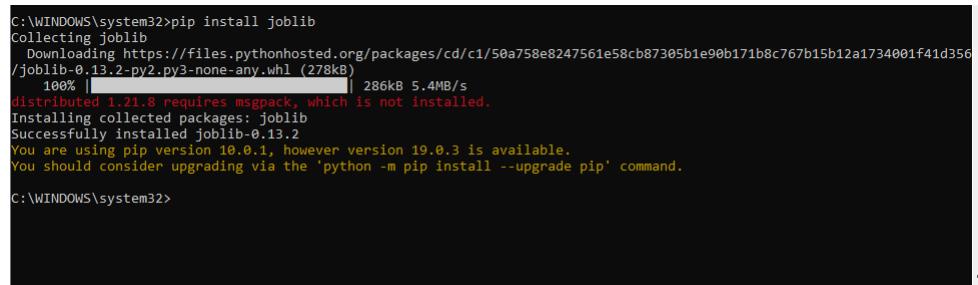
- hasilnya seperti di atas

### 1.7.7 Penanganan Error

Dari percobaan yang telah dilakukan terdapat error yang di dapatkan pada bagian joblib model persistence

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

Figure 1.50: Error



```
C:\WINDOWS\system32>pip install joblib
Collecting joblib
  Downloading https://files.pythonhosted.org/packages/cd/c1/50a758e8247561e58cb87305b1e90b171b8c767b15b12a1734001f41d356
/joblib-0.13.2-py2.py3-none-any.whl (278kB)
    100% |██████████| 286kB 5.4MB/s
distributed 1.21.8 requires msgpack, which is not installed.
Installing collected packages: joblib
Successfully installed joblib-0.13.2
You are using pip version 10.0.1, however version 19.0.3 is available.
You should consider upgrading via the 'python -m pip install --upgrade pip' command.

C:\WINDOWS\system32>
```

Figure 1.51: Instalasi

- Error tersebut dikarenakan saya belum install package joblib sehingga error pun terjadi dengan source code sebagai berikut: from joblib import dump, load
- Untuk itu solusinya saya install terlebih dahulu joblib agar error pun tidak terjadi kembali.

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

Figure 1.52: Hasil

# **Chapter 2**

## **Related Works**

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

### **2.1 Same Topics**

Cite every latest journal with same topic

#### **2.1.1 Topic 1**

cite for first topic

#### **2.1.2 Topic 2**

if you have two topics you can include here to

### **2.2 Same Method**

write and cite latest journal with same method

#### **2.2.1 Method 1**

cite and paraphrase method 1

#### **2.2.2 Method 2**

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

## 2.3 Yusniar Nur Syarif Sidiq/1164089

### 2.3.1 Binary Classification

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

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

Untuk contoh Binary Classification dapat dilihat pada gambar 2.1

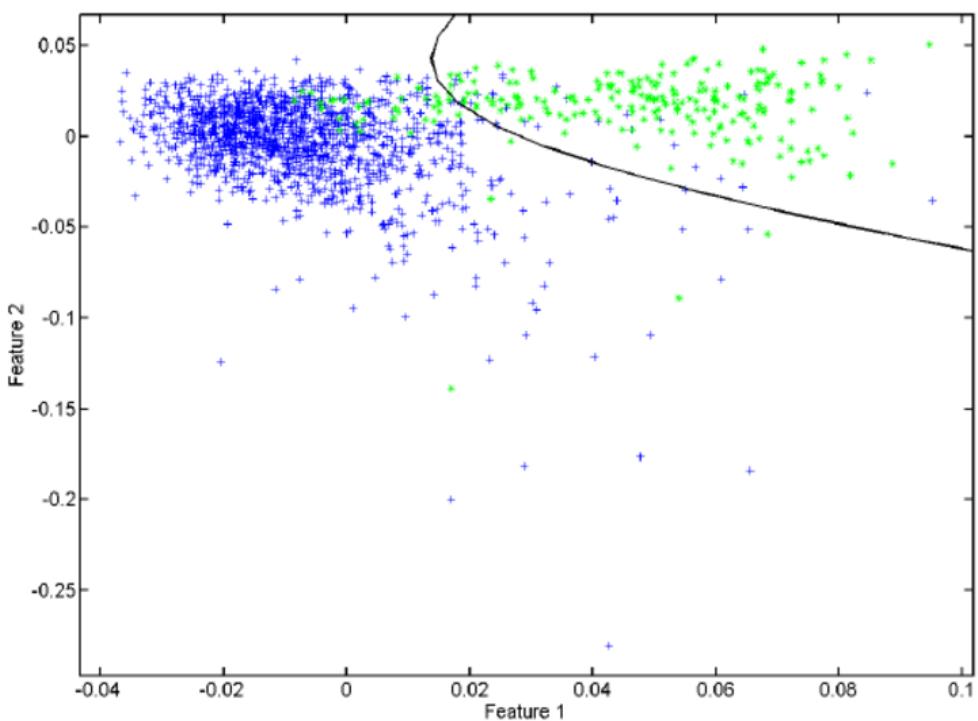


Figure 2.1: Binary Classification.

### 2.3.2 Supervised Learning, Unsupervised Learning, Dan Classtering

1. Supervised Learning merupakan sebuah pendekatan yang dimana sudah adanya sdata yang dilatih dan telah terdapat variabel yang telah ditargetkan sehingga bertujuan untuk mengelompokkan suatu data ke data yang sudah ada. Contoh dalam Supervised Learning yaitu ketika anda memiliki sejumlah buku yang yang telah dilabel dengan urutan kategori tertentu. Ketika anda akan membeli sebuah buku baru, maka harus di identifikasi isi dari buku tersebut dan memasukkannya kedalam kategori tertentu. Ketika anda membeli sebuah buku tersebut maka anda telah menerapkan sebuah logika fuzzy. Ilustrasi Supervised Learning dapat dilihat pada gambar 2.2.

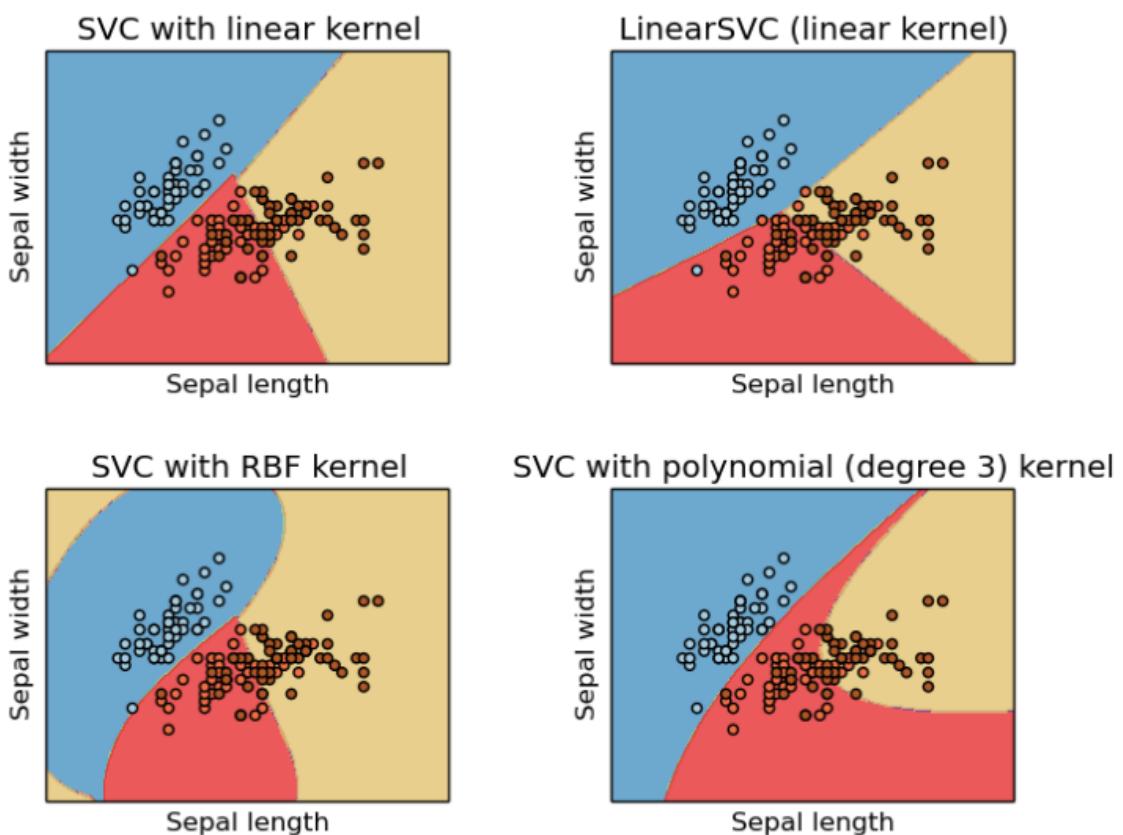


Figure 2.2: Supervised Learning.

2. Unsupervised Learning merupakan sebuah data yang belum ditentukan variabelnya jadi hanya berupa data saja. Dalam sebuah kasus Unsupervised Learning adalah aggap saja anda belum pernah membeli buku sama sekali dan pada

suatu hari anda telah membeli buku dengan sangat banyak dalam kategori yang berbeda. Sehingga buku tersebut belum dikategorikan dan hanya berupa data buku saja. Ilustrasi Unsupervised Learning dapat dilihat pada gambar 2.3.

## Unsupervised Learning

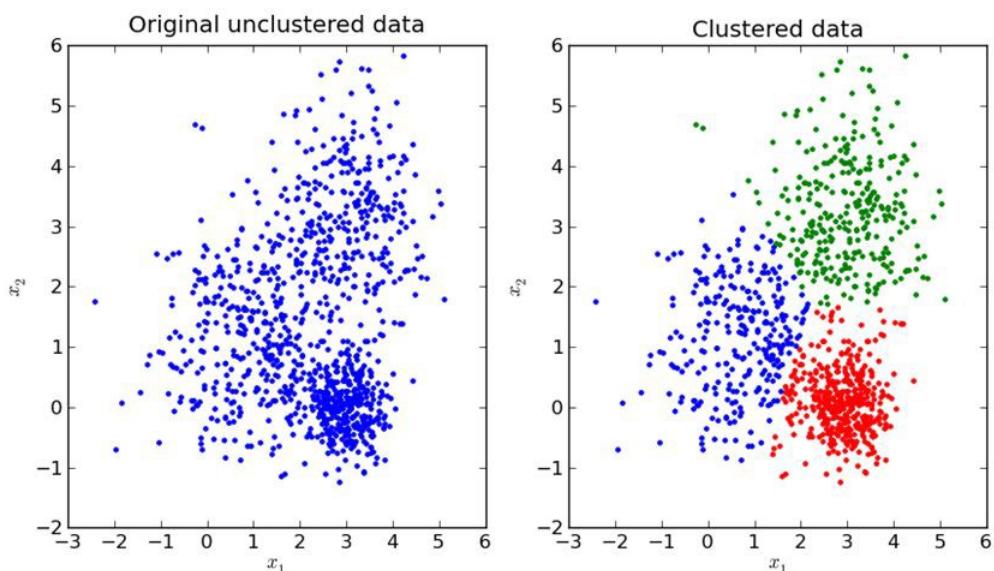


Figure 2.3: Unsupervised Learning.

3. Clustering merupakan sebuah proses untuk mengklasifikasikan sebuah data dalam satu parameter. Dalam kasus ini dapat dijelaskan ada beberapa orang yang memiliki kekuatan tubuh yang sehat dan kekuatan tubuh yang lemah. Parameter bagi orang yang memiliki tubuh yang kuat adalah orang yang terlihat bugar dan sehat maka dengan orang yang memiliki parameter adalah orang yang memiliki kekuatan tubuh yang kuat dan untuk kekuatan tubuh yang lemah adalah sebaliknya. Ilustrasi gambar dapat dilihat di gambar 2.4

### 2.3.3 Evaluasi Dan Akurasi

1. Evaluasi dan akurasi adalah bagaimana cara kita dapat mengevaluasi seberapa baik model melakukan pekerjaannya dengan cara mengukur akurasinya. Akurasi

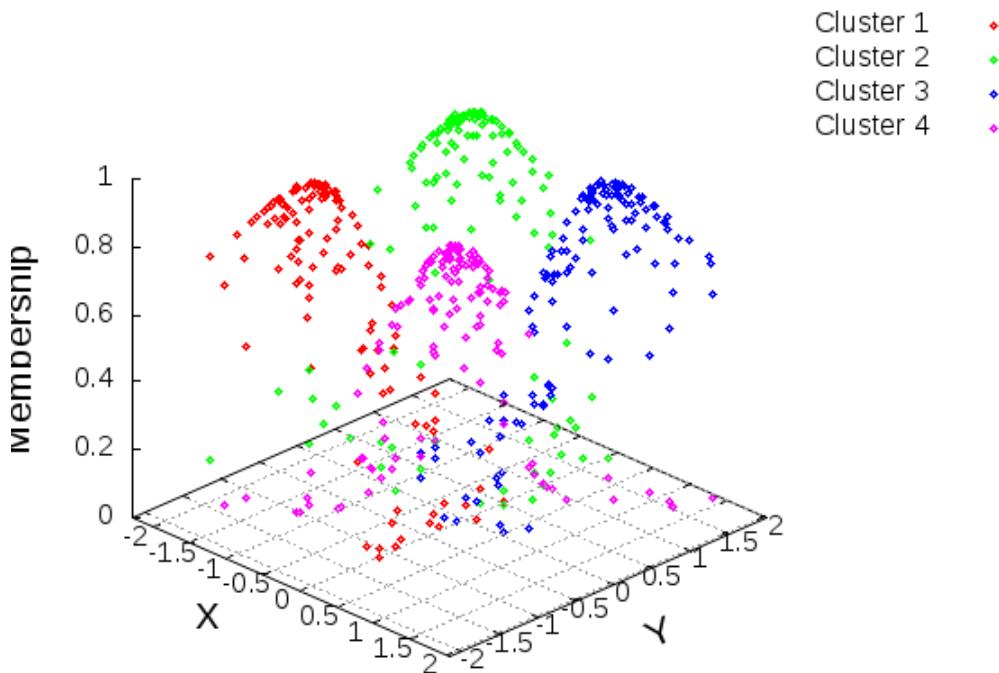


Figure 2.4: Clustering.

akan didefinisikan sebagai persentase kasus yang telah diklasifikasikan dengan benar. Kita dapat melakukan analisis kesalahan yang telah dibuat oleh model. Dalam tabel tersebut baris true mangga dan true anggur menunjukkan kasus apakah itu objek mangga atau anggur. Kolom telah diprediksi dan dibuat oleh model. Ada 20 mangga yang diprediksi benar dan ada 5 anggur yang diprediksi salah. Ilustrasi dapat dilihat pada gambar 2.5

### 2.3.4 Confusion Matrix

1. Ada beberapa cara untuk membuat dan membaca confusion matrix antara lain
  - Tentukan pokok permasalahan serta atributnya
  - Buat Decision Tree
  - Buat Data Testing
  - Mencari nilai variabelnya misal a,b,c, dan d
  - Mencari nilai recall, precision, accuracy, dan error rate

	<u>Predicted Mangga</u>	<u>Predicted Anggur</u>
<u>True Mangga</u>	20	5
<u>True Anggur</u>	3	22

Figure 2.5: Evaluasi Dan Akurasi.

Di bawah ini adalah contoh dari confusion matrix

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

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

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

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

### 2.3.5 Cara Kerja K-Fold Cross Validation

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

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

Untuk ilustrasi K-Fold Cross Validation data di lihat pada gambar 2.6

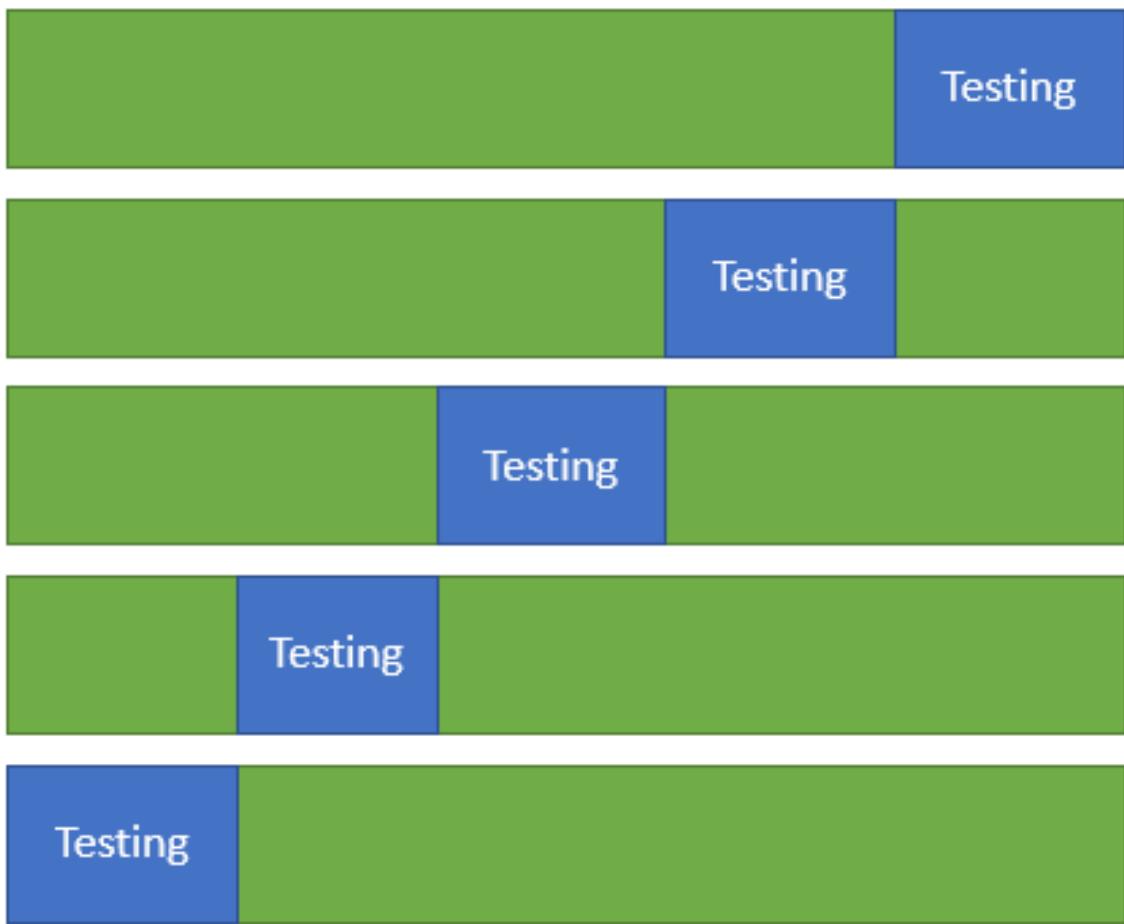


Figure 2.6: K-Fold Cross Validation.

### 2.3.6 Decision Tree

1. Decision Tree adalah sebuah metode pembelajaran yang digunakan untuk melakukan klarifikasi dan regresi. Decision Tree digunakan untuk membuat sebuah model yang dapat memprediksi sebuah nilai variabel target dengan cara mempelajari aturan keputusan dari fitur data. Contoh Decision Tree adalah untuk melakukan prediksi apakah Kuda termasuk hewan mamalia atau bukan, lihat pada gambar 2.7.

### 2.3.7 Gain Dan Entropi

1. Gain adalah pengurangan yang diharapkan dalam entropy. Dalam machine learning, gain dapat digunakan untuk menentukan sebuah urutan atribut atau memperkecil atribut yang telah dipilih. Urutan ini akan membentuk decision tree. atribut gain dipilih yang paling besar.

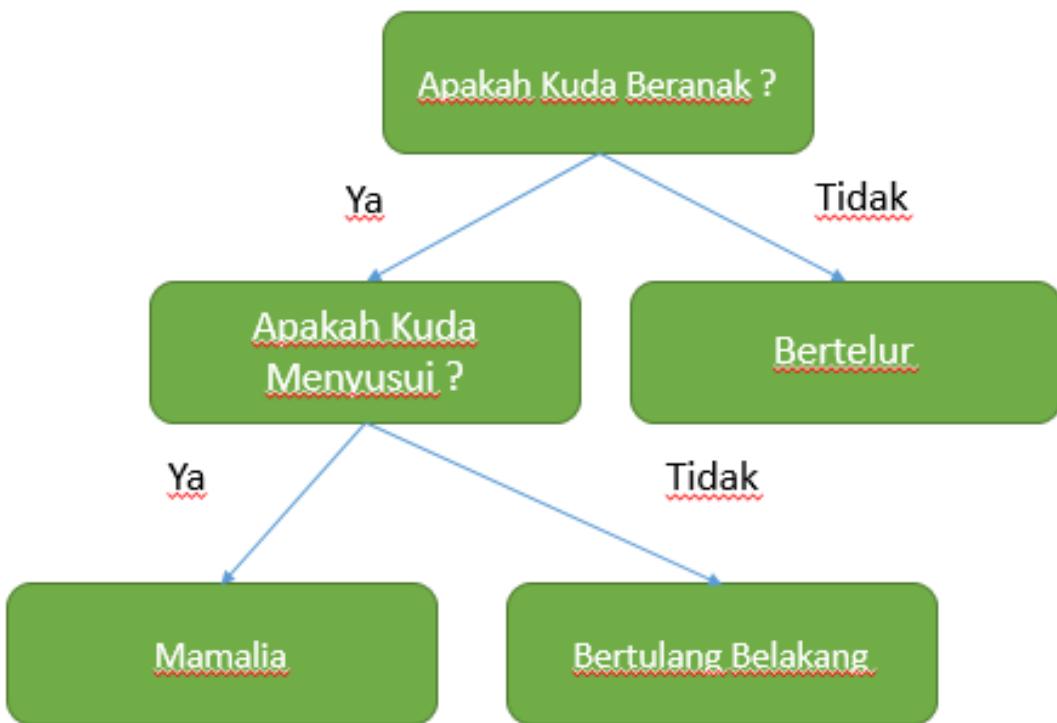


Figure 2.7: Decision Tree.

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

Ilustrasi dari gain dan entropi adalah bagaimana kita memprediksi jenis kelamin berdasarkan atributnya, perhatikan pada gambar 2.8

## 2.4 Scikit Learn

```

1. import pandas as salak
sawo = salak.read_csv('D:/Perkuliahinan/Semester 6/
Kecerdasan Buatan/BUKU AI/
Python-Artificial-Intelligence-Projects-for-Beginners/
Chapter01/dataset/student-por.csv', sep=';')
len(sawo)
  
```

Source Code tersebut digunakan untuk melakukan import pandas yang di rename dengan salak. Disini saya membuat variabel sawo yang dimana isinya fungsi dari read.csv lalu panggil file csv nya yaitu student-por.csv dan separatornya adalah ";". fungsi dari len sendiri yaitu hanya meng outputkan saja.

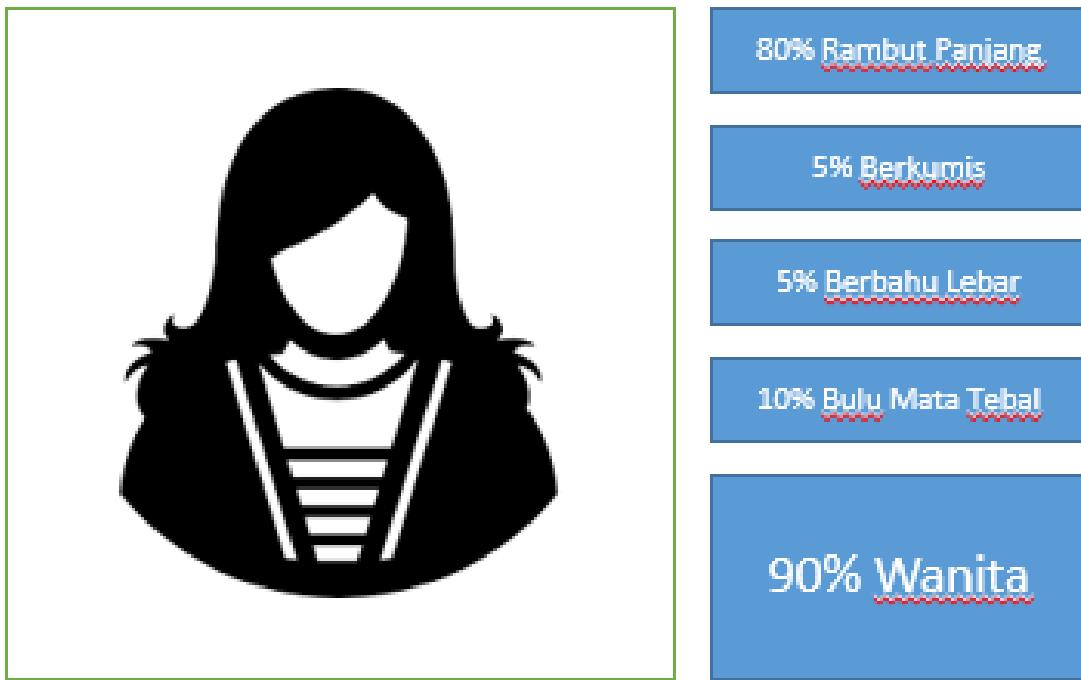


Figure 2.8: Gain Dan Entropi.

Hasil source code yang di running dalam spyder tersebut adalah gambar 2.9

```
In [14]: import pandas as salak
...: sawo = salak.read_csv('D:\Perkuliahan\Semester 6\Kecerdasan Buatan\BUKU AI
\Python-Artificial-Intelligence-Projects-for-Beginners\Chapter01\dataset\student-
por.csv', sep=';')
...: len(sawo)
Out[14]: 649
```

Figure 2.9: Output No 1.

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

Source Code tersebut akan menambahkan satu field yang diberi nama "pas", dimana lamda tersebut merupakan decision yang berada di dalam decision atau if else. Dimana if else tersebut dinilai apabila lebih dari 35 maka dinyatakan lulus. untuk axis sendiri yaitu apabila lulus maka akan di deklarasikan dengan angka 1 dan apabila tidak lulus akan di deklarasikan dengan angka 0. Selanjutnya akan di running kembali. Hasil running dari spyder dapat dilihat pada

gambar 2.10.

```
Out[15]:  
    school sex  age address famsize ...  Dalc  Walc  health absences pass  
0      GP   F    18      U    GT3 ...    1     1      3        4     0  
1      GP   F    17      U    GT3 ...    1     1      3        2     0  
2      GP   F    15      U    LE3 ...    2     3      3        6     1  
3      GP   F    15      U    GT3 ...    1     1      5        0     1  
4      GP   F    16      U    GT3 ...    1     2      5        0     1  
  
[5 rows x 31 columns]
```

Figure 2.10: Output No 2.

```
3. sawo = salak.get_dummies(sawo, columns=  
    ['sex', 'school', 'address', 'famsize', 'Pstatus', 'Mjob', 'Fjob',  
     'reason', 'guardian', 'schoolsup',  
     'famsup', 'paid', 'activities',  
     'nursery', 'higher', 'internet', 'romantic'])  
sawo.head()
```

Source Code tersebut hanya menambahkan field baru akan tetapi dengan fungsi Binary Classification. Hasil running dari spyder adalah yang ditunjukan pada gambar 2.11.

```
Out[16]:  
    age  Medu  Fedu  ...  internet_yes  romantic_no  romantic_yes  
0    18    4     4  ...          0            1            0  
1    17    1     1  ...          1            1            0  
2    15    1     1  ...          1            1            0  
3    15    4     2  ...          1            0            1  
4    16    3     3  ...          0            1            0  
  
[5 rows x 57 columns]
```

Figure 2.11: Output No 3.

```
4. sawo = sawo.sample(frac=1)  
  
sawo_train = sawo[:500]  
sawo_test = sawo[500:]  
  
sawo_train_att = sawo_train.drop(['pass'], axis=1)
```

```

sawo_train_pass = sawo_train['pass']

sawo_test_att = sawo_test.drop(['pass'], axis=1)
sawo_test_pass = sawo_test['pass']

sawo_att = sawo.drop(['pass'], axis=1)
sawo_pass = sawo['pass']

import numpy as kelapa
print("Passing: %d out of %d (%.2f%%)" % (kelapa.sum(sawo_pass),
                                             len(sawo_pass),
                                             100*float(kelapa.sum(sawo_pass)) / len(sawo_pass)))

```

Pada source code tersebut terdapat data train dan juga test yang dimana digunakan untuk membagi data training dan juga data test. Selanjutnya akan melakukan import numpy yang di rename dengan kelapa yang digunakan untuk mengembalikan suatu nilai passing dari keseluruhan datasets dengan cara melakukan print. Hasil running dalam spyder ditunjukkan pada gambar 2.12.

Name	Type	Size	Value
sawo	DataFrame	(649, 57)	Column names: age, Medu, Fedu, travelttime, studytime, failures, famrel ...
sawo_att	DataFrame	(649, 56)	Column names: age, Medu, Fedu, travelttime, studytime, failures, famrel ...
sawo_pass	Series	(649,)	Series object of pandas.core.series module
sawo_test	DataFrame	(149, 57)	Column names: age, Medu, Fedu, travelttime, studytime, failures, famrel ...
sawo_test_att	DataFrame	(149, 56)	Column names: age, Medu, Fedu, travelttime, studytime, failures, famrel ...
sawo_test_pass	Series	(149,)	Series object of pandas.core.series module
sawo_train	DataFrame	(500, 57)	Column names: age, Medu, Fedu, travelttime, studytime, failures, famrel ...
sawo_train_att	DataFrame	(500, 56)	Column names: age, Medu, Fedu, travelttime, studytime, failures, famrel ...
.	.	(...)	.

Figure 2.12: Output No 4.

```

5. from sklearn import tree
anggur = tree.DecisionTreeClassifier(criterion="entropy", max_depth=5)
anggur = anggur.fit(sawo_train_att, sawo_train_pass)

```

Dimana akan di importkan modul bernama tree dari library scikitlearn dan kemudian akan di definisikan variabelnya dengan anggur menggunakan Deci-

sionClassifier. Pada variabel anggur terdapat fungsi Criterion yang dapat mengukur kualitas split. Untuk menjalankan DecisionTreeClassifier dibutuhkan sebuah perintah fit. Hasil running dalam spyder dapat dilihat pada gambar 2.13

```
In [18]: from sklearn import tree
...: anggur = tree.DecisionTreeClassifier(criterion="entropy", max_depth=5)
...: anggur = anggur.fit(sawo_train_att, sawo_train_pass)
```

Figure 2.13: Output No 5.

```
6. import graphviz
dot_data = tree.export_graphviz(anggur, out_file=None,
label="all", impurity=False,
proportion=True,
feature_names=list(sawo_train_att),
class_names=["fail", "pass"],
filled=True, rounded=True)
graph = graphviz.Source(dot_data)
graph
```

Graphviz merupakan software visualisasi grafik yang open source. Dimana ada yang dinamakan fungsi Treexportgraphviz, dimana fungsi tersebut dapat menghasilkan representasi Graphviz yang di ambil dari Decision Tree dan ditulis kedalam outfile sehingga dapat memunculkan gambar diagram dengan grafik bercabang. Bila Source code tersebut di running dalam spyder maka hasilnya akan terlihat seperti pada gambar 2.14.

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

Treexportgraphviz, dimana fungsi tersebut dapat menghasilkan representasi Graphviz yang di ambil dari Decision Tree dan ditulis kedalam outfile. Dalam source code tersebut akan menyimpan classifier dan melakukan ekspor ke file student performance dan apabila salah akan mengembalikan sebuah nilai fail. Hasil running dalam spyder dapat dilihat pada gambar 2.15.

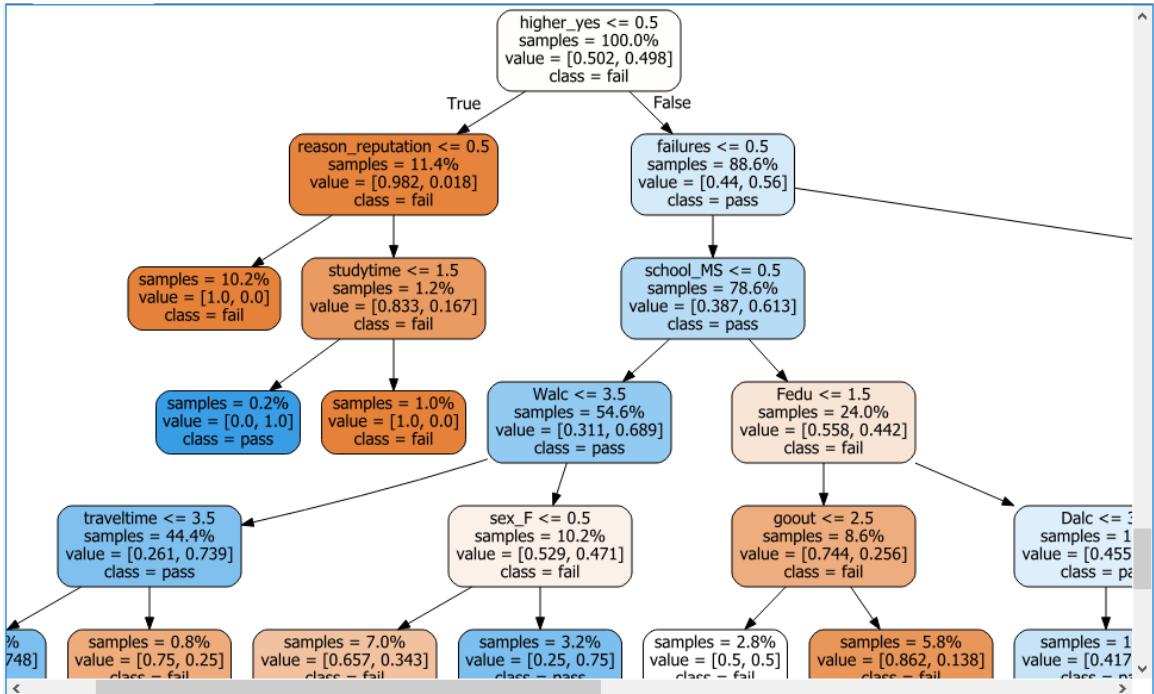


Figure 2.14: Output No 6.

```
In [20]: tree.export_graphviz(anggur, out_file="student-performance.dot", label="all", impurity=False,
proportion=True,
...: feature_names=list(sawo_train_att), class_names=["fail", "pass"],
...: filled=True, rounded=True)
```

Figure 2.15: Output No 7.

8. `anggur.score(sawo_test_att, sawo_test_pass)`

Dimana score adalah sebuah prediksi dan juga merupakan sebuah proses yang menghasilkan sebuah nilai berdasarkan model machine learning yang terlatih dan sudah diberikan beberapa data baru. Score yang telah dibuat dapat mewakili prediksi suatu nilai, dalam kasus ini variabel anggur akan melakukan prediksi nilai dari variabel sawo dari data testing att dan testing pass. Apabila source code tersebut di running dalam spyder maka hasilnya akan terlihat seperti pada gambar 2.16.

```
In [21]: anggur.score(sawo_test_att, sawo_test_pass)
Out[21]: 0.7248322147651006
```

Figure 2.16: Output No 8.

9. `from sklearn.model_selection import cross_val_score`

```

scores = cross_val_score(anggur, sawo_att, sawo_pass, cv=5)
print("Accuracy: %0.2f (+/- %0.2f)" % (scores.mean(), scores.std() * 2))

```

Source code tersebut akan melakukan evaluasi terhadap nilai score dengan validasi silang. Dimana variabel Scores berisikan cross\_val\_score yaitu sebuah fungsi bantu pada estimator dan juga dataset. Dari hasil tersebut akan ditunjukkan score rata-rata dan kurang-lebih dua standar deviasi yang telah mencangkup 95% score, data tersebut akan di print sehingga menunjukkan hasil seperti pada gambar 2.17.

```

In [22]: from sklearn.model_selection import cross_val_score
...: scores = cross_val_score(anggur, sawo_att, sawo_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: 0.67 (+/- 0.08)

```

Figure 2.17: Output No 9.

```

10. for max_depth in range(1, 20):
    t = tree.DecisionTreeClassifier(criterion="entropy",
                                    max_depth=max_depth)
    scores = cross_val_score(t, sawo_att, sawo_pass, cv=5)
    print("Max depth: %d, Accuracy: %0.2f (+/- %0.2f)" % (max_depth, scores.
    scores.std() * 2))

```

Dalam source code tersebut akan mendalami fungsi tree. Dimana semakin dalam tree maka akan semakin banyak perpecahan yang dimiliki dan dapat menangkap lebih banyak informasi. Dalam kasus ini variabel t akan melakukan pendekomposisi file tree yang kemudian variabel scores akan melakukan evaluasi score dengan validasi silang. Hasil running source code tersebut di dalam spyder akan di perlihatkan pada gambar 2.18.

```

11. depth_acc = kelapa.empty((19,3), float)
    i = 0
    for max_depth in range(1, 20):
        t = tree.DecisionTreeClassifier(criterion="entropy",
                                        max_depth=max_depth)
        scores = cross_val_score(t, sawo_att,
                                sawo_pass, cv=5)
        depth_acc[i,0] =
        max_depth

```

```

IPython console
Console 1/A
.... t = tree.DecisionTreeClassifier(criterion= entropy , max_depth=max_depth)
.... scores = cross_val_score(t, sawo_att, sawo_pass, cv=5)
.... print("Max depth: %d, Accuracy: %0.2f (+/- %0.2f)" % (max_depth, scores.mean(),
scores.std() * 2))
Max depth: 1, Accuracy: 0.62 (+/- 0.05)
Max depth: 2, Accuracy: 0.69 (+/- 0.01)
Max depth: 3, Accuracy: 0.69 (+/- 0.02)
Max depth: 4, Accuracy: 0.69 (+/- 0.06)
Max depth: 5, Accuracy: 0.68 (+/- 0.07)
Max depth: 6, Accuracy: 0.67 (+/- 0.07)
Max depth: 7, Accuracy: 0.69 (+/- 0.05)
Max depth: 8, Accuracy: 0.67 (+/- 0.08)
Max depth: 9, Accuracy: 0.68 (+/- 0.10)
Max depth: 10, Accuracy: 0.66 (+/- 0.09)
Max depth: 11, Accuracy: 0.65 (+/- 0.08)
Max depth: 12, Accuracy: 0.65 (+/- 0.10)
Max depth: 13, Accuracy: 0.64 (+/- 0.08)
Max depth: 14, Accuracy: 0.64 (+/- 0.08)
Max depth: 15, Accuracy: 0.64 (+/- 0.08)
Max depth: 16, Accuracy: 0.66 (+/- 0.10)
Max depth: 17, Accuracy: 0.66 (+/- 0.11)
Max depth: 18, Accuracy: 0.64 (+/- 0.09)
Max depth: 19, Accuracy: 0.64 (+/- 0.10)

In [24]: |

```

Figure 2.18: Output No 10.

```

depth_acc[i,1] =
scores.mean()
depth_acc[i,2] =
scores.std() * 2
i += 1

depth_acc

```

Depth acc akan membuat array kosong dengan mengembalikan array baru dengan bentuk dan tipe yang diberikan, tanpa menginisialisasi entri. Dimana 19 merupakan bentuk array kosong dan 3 adalah output data-type sedangkan float urutan kolom-utama dalam memori. Pada source code tersebut jika di running dalam spyder maka akan menunjukkan hasil seperti pada gambar 2.19.

```

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

```

Dimana source code tersebut akan melakukan import dari library matplotlib

```
....  
....:  
....: depth_acc  
Out[24]:  
array([[1.00000000e+00, 6.16234552e-01, 4.62227393e-02],  
       [2.00000000e+00, 6.87243396e-01, 1.23768569e-02],  
       [3.00000000e+00, 6.94924142e-01, 2.23150024e-02],  
       [4.00000000e+00, 6.82591687e-01, 6.18637635e-02],  
       [5.00000000e+00, 6.71834018e-01, 7.60033236e-02],  
       [6.00000000e+00, 6.74863966e-01, 6.76815174e-02],  
       [7.00000000e+00, 6.82580672e-01, 3.83801373e-02],  
       [8.00000000e+00, 6.68733789e-01, 7.48310958e-02],  
       [9.00000000e+00, 6.74982680e-01, 1.01101809e-01],  
       [1.00000000e+01, 6.54863966e-01, 7.64335218e-02],  
       [1.10000000e+01, 6.54863966e-01, 9.17627318e-02],  
       [1.20000000e+01, 6.39348528e-01, 1.07921469e-01],  
       [1.30000000e+01, 6.51727776e-01, 8.93160325e-02],  
       [1.40000000e+01, 6.51751811e-01, 1.11646902e-01],  
       [1.50000000e+01, 6.47112028e-01, 7.34148210e-02],  
       [1.60000000e+01, 6.39419902e-01, 9.05497305e-02],  
       [1.70000000e+01, 6.40910841e-01, 7.46834954e-02],  
       [1.80000000e+01, 6.42425269e-01, 7.07916986e-02],  
       [1.90000000e+01, 6.34780665e-01, 8.24965986e-02]])
```

In [25]:

Figure 2.19: Output No 11.

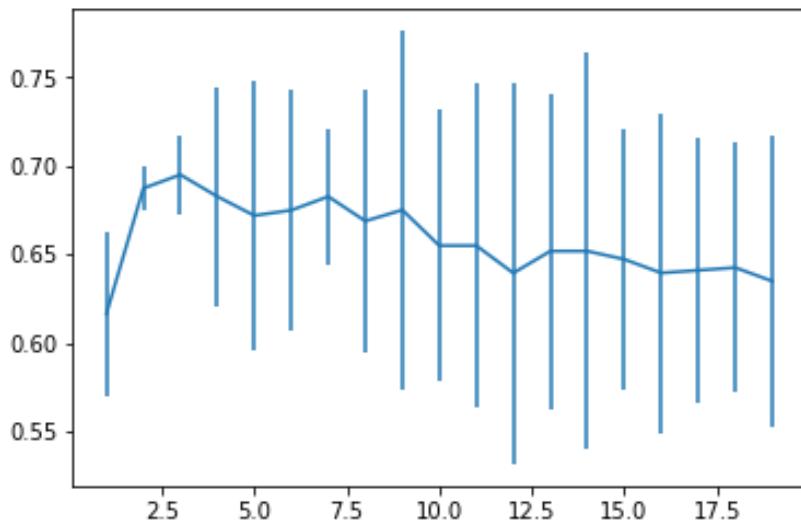
yang berupa pyplot dan di rename menjadi plt. fig dan juga ax akan menggunakan subplots guna membuat sebuah gambar dan satu set subplot. Fungsi ax.errorbar sendiri yaitu akan membuat error bar dan kemudian grafik akan ditampilkan menggunakan plt.show. Hasil source code tersebut jika di running dalam spyder ditunjukkan pada gambar 2.20.

## 2.5 Penangan Erorr

1. Untuk Screenshot yang erorr dapat dilihat pada gambar 2.21.
2. Untuk bagian erorrnya adalah sebagai berikut

ExecutableNotFoundError: failed to execute

```
In [25]: import matplotlib.pyplot as plt
....: fig, ax = plt.subplots()
....: ax.errorbar(depth_acc[:,0], depth_acc[:,1], yerr=depth_acc[:,2])
....: plt.show()
```



```
In [26]:
```

Figure 2.20: Output No 12.

```
['dot', '-Tsvg'], make sure the Graphviz
executables are on your system' PATH
```

3. Untuk penanganannya dengan cara lakukan installasi graphviz seperti yang diperlihatkan pada gambar 2.22 dan 2.23.

## 2.6 Andri Fajar Sunandhar/1164065

### 2.6.1 binary classification dilengkapi ilustrasi gambar

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

```

File "C:\Users\NS\Anaconda3\lib\site-packages\graphviz\backend.py", line 206, in pipe
    out, _ = run(cmd, input=data, capture_output=True, check=True, quiet=quiet)

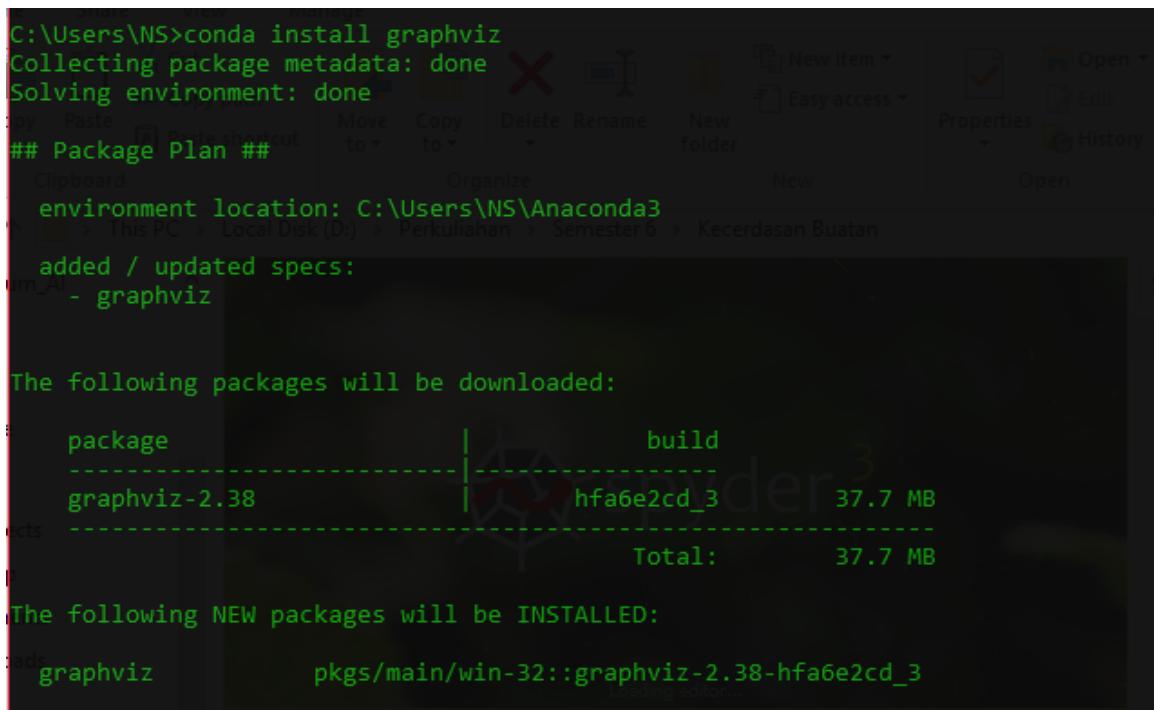
File "C:\Users\NS\Anaconda3\lib\site-packages\graphviz\backend.py", line 150, in run
    raise ExecutableNotFoundError(cmd)

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

Out[9]: <graphviz.files.Source at 0x9709fd0>

```

Figure 2.21: Graphviz Error.



The screenshot shows a Windows File Explorer window with a dark theme. The address bar indicates the path: C:\Users\NS>conda install graphviz. The main area displays the output of the command:

```

C:\Users\NS>conda install graphviz
Collecting package metadata: done
Solving environment: done
## Package Plan ##
environment location: C:\Users\NS\Anaconda3
added / updated specs:
  - graphviz

The following packages will be downloaded:
  package          | build
  graphviz-2.38   | hfa6e2cd_3
Total:           37.7 MB

The following NEW packages will be INSTALLED:
  graphviz         pkgs/main/win-32::graphviz-2.38-hfa6e2cd_3

```

Figure 2.22: Graphviz Install 1.

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

1. Supervised learning adalah tugas pembelajaran mesin untuk mempelajari suatu fungsi yang memetakan input ke output berdasarkan contoh pasangan input-output. Ini menyimpulkan fungsi dari data pelatihan berlabel yang terdiri dari serangkaian contoh pelatihan. Dalam pembelajaran yang diawasi, setiap contoh adalah pasangan yang terdiri dari objek input (biasanya vektor) dan nilai output yang diinginkan (juga disebut sinyal pengawas). Algoritma pembelajaran yang diawasi menganalisis data pelatihan dan menghasilkan fungsi yang disimpulkan, yang dapat digunakan untuk memetakan contoh-contoh baru. Ske-

```

Proceed ([y]/n)? y
[...]
IMPORTANT NOTE: It seems that you are using Spyder with Anaconda!
Downloading and Extracting Packages
graphviz-2.38 | 37.7 MB  | #####|#####|#####|#####|#####|#####|#####|#####|#####| 100%
Preparing transaction: done
Verifying transaction: done, please wait until new conda packages are available and use conda to
Executing transaction: done
[...]
C:\Users\NS>

```

Figure 2.23: Graphviz Install 2.

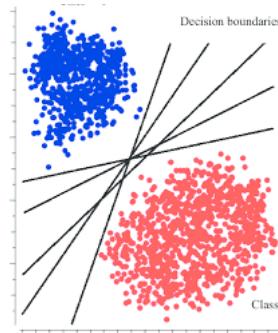


Figure 2.24: Binary Classification

nario optimal akan memungkinkan algoritma menentukan label kelas dengan benar untuk instance yang tidak terlihat. Ini membutuhkan algoritma pembelajaran untuk menggeneralisasi dari data pelatihan untuk situasi yang tidak terlihat dengan cara yang "masuk akal" (lihat bias induktif). Tugas paralel dalam psikologi manusia dan hewan sering disebut sebagai pembelajaran konsep. Contoh dibawah yaitu Supervised Learning dengan SVC.

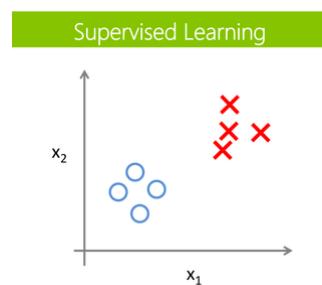


Figure 2.25: Supervised Learning

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

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

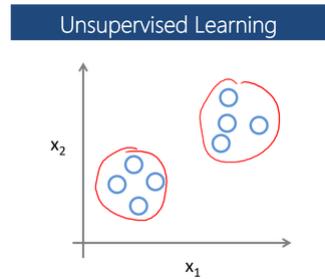


Figure 2.26: Unsupervised Learning

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

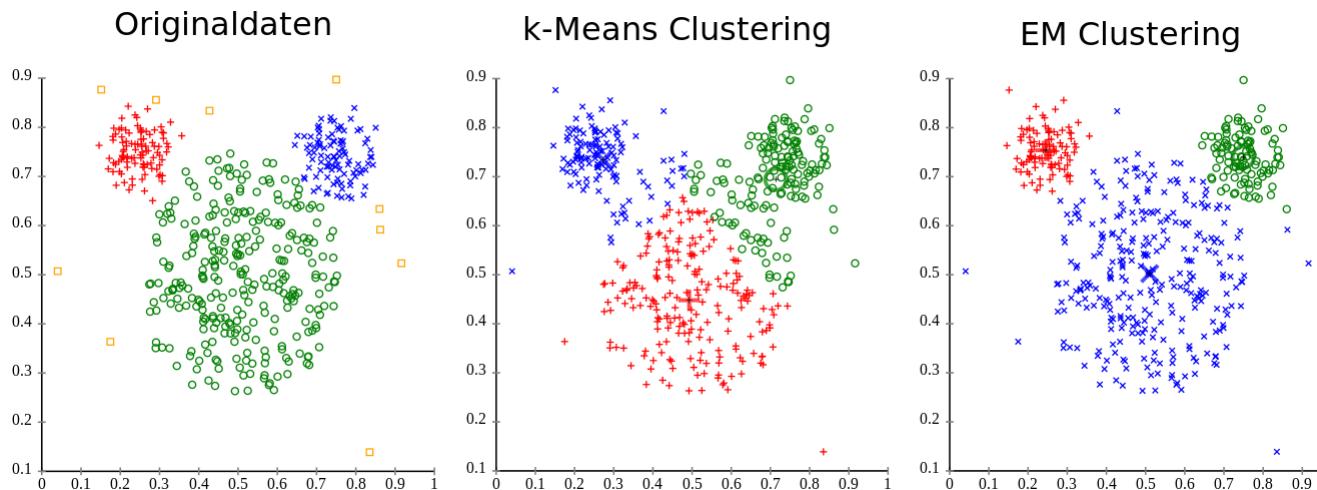


Figure 2.27: Cluster

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

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

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

Figure 2.28: Evaluasi dan Akurasi

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

1. Cara membuat dan membaca confusion matrix :

- 1) Tentukan pokok permasalahan dan atributanya, misal gaji dan listik.

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

### **2.6.5 bagaimana K-fold cross validation bekerja dengan gambar ilustrasi**

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

### **2.6.6 decision tree dengan gambar ilustrasi**

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

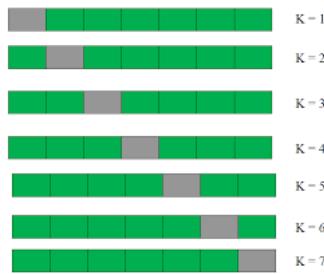


Figure 2.29: K-fold cross validation

sederhana yang disimpulkan dari fitur data.

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

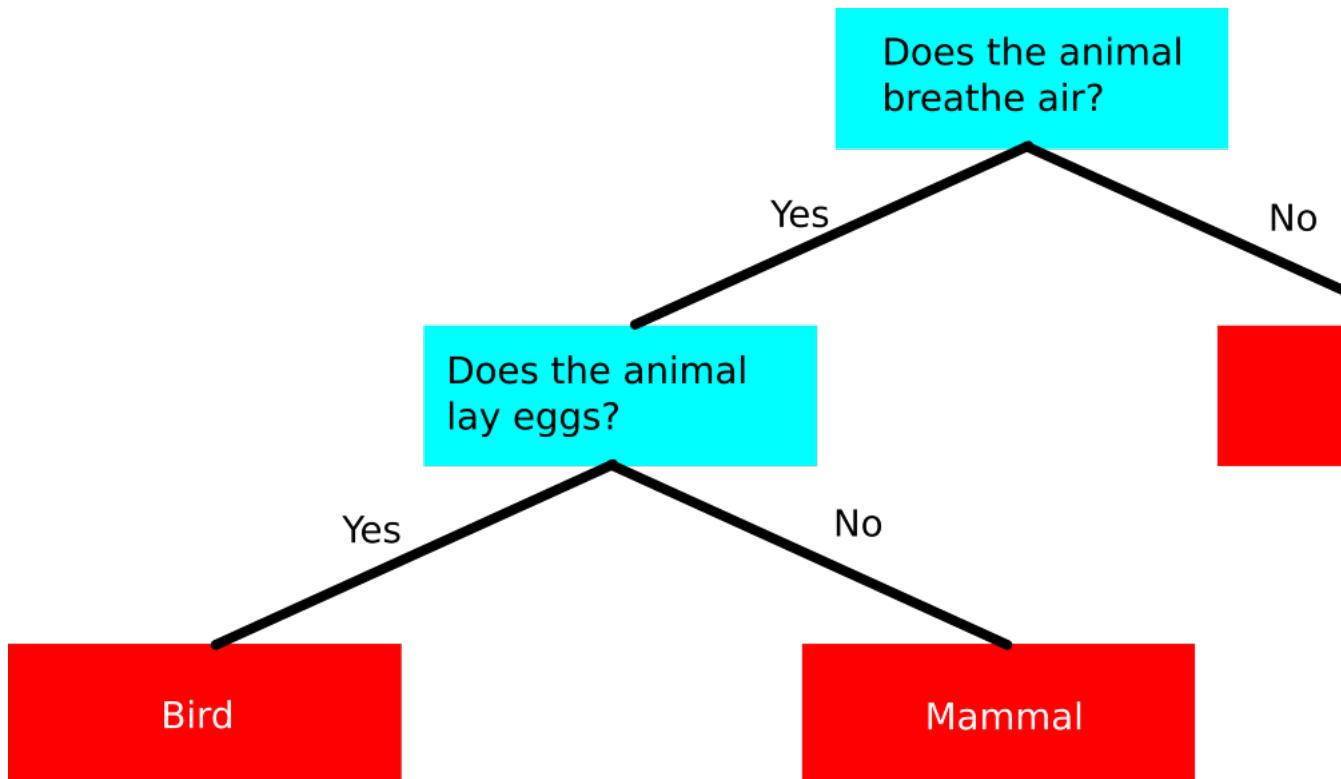


Figure 2.30: Decision Tree

### 2.6.7 Information Gain dan entropi dengan gambar ilustrasi

1. Information gain didasarkan pada penurunan entropi setelah dataset dibagi pada atribut. Membangun decision tree adalah semua tentang menemukan

atribut yang mengembalikan perolehan informasi tertinggi (mis., Cabang yang paling homogen).



Figure 2.31: Information gain

2. Entropi adalah ukuran keacakan dalam informasi yang sedang diproses. Semakin tinggi entropi, semakin sulit untuk menarik kesimpulan dari informasi itu. Membalik koin adalah contoh tindakan yang memberikan informasi yang acak. Untuk koin yang tidak memiliki afinitas untuk kepala atau ekor, hasil dari sejumlah lemparan sulit diprediksi. Mengapa? Karena tidak ada hubungan antara membalik dan hasilnya. Inilah inti dari entropi.

## 2.7 Imron Sumadireja / 1164076

### 2.7.1 Binary Classification

1. Binary classification merupakan suatu cara untuk mengklasifikasikan atau mengkategorikan objek set dengan atribut ke dalam ke dua kategori yang sudah ada atau biasa disebut dengan supervised. Binary classification dapat diterapkan dengan tujuan praktis, dalam banyak masalah binary classification. Untuk contoh binary classification dapat dilihat pada gambar 2.32

### 2.7.2 Supervised Learning, Unsupervised Learning, dan Clustering

1. Supervised learning merupakan suatu pembelajaran bagi mesin untuk mempelajari suatu fungsi yang memetakan input ke output berdasarkan data yang telah diberikan dan terdapat variable yang telah ditargetkan sehingga tujuan dari pembelajaran ini mesin dapat memetakan output dengan baik. Sehingga proses training yang dilakukan pada mesin dapat berjalan sesuai dengan target

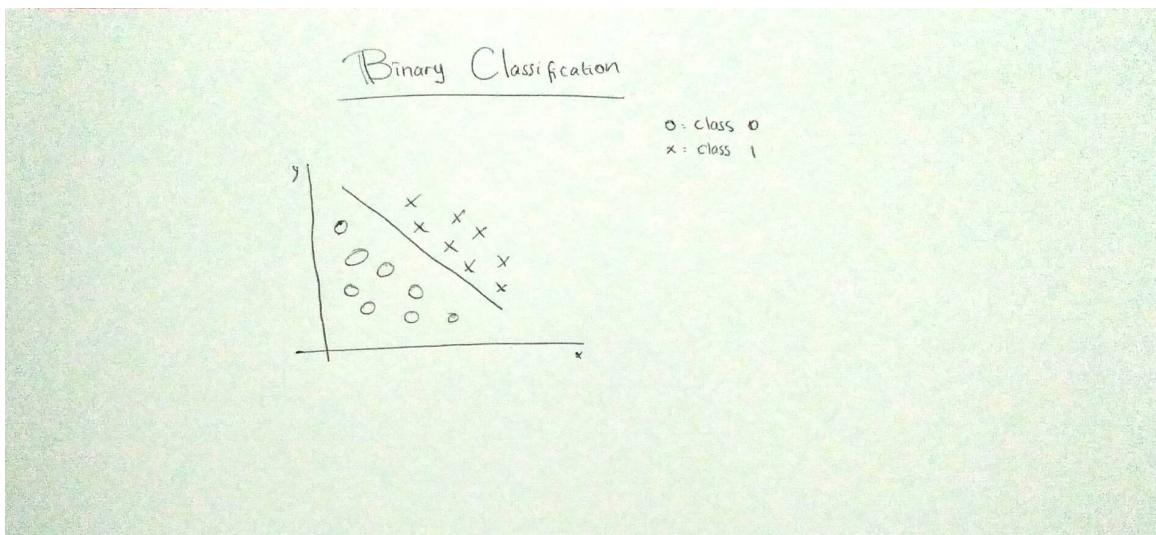


Figure 2.32: Binary Classification.

yang ditentukan dan hasil dari data training tersebut dapat digunakan untuk melakukan prediksi. Contoh supervised learning dapat dilihat pada gambar berikut 2.33

2. Unsupervised learning merupakan suatu pembelajaran bagi mesin, namun tidak memiliki data latih, atau data training. Unsupervised ini dapat mengklasifikasi suatu objek secara langsung dengan atribut seadanya pada data tersebut. Sebagai contoh, jika kita ingin mengelompokkan sekumpulan orang hanya diperlukan dari data yang ada misalnya dari jenis kelamin, pakaian yang digunakan, dan lain sebagainya. Oleh karena itu unsupervised learning ini tidak memiliki data training. Contoh supervised learning dapat dilihat pada gambar berikut 2.34
3. Clustering adalah metode pengelompokan data ke dalam beberapa cluster atau kelompok agar data dalam satu cluster tersebut memiliki tingkat kemiripan yang maksimum dan data dengan kemiripan yang minimum. Clustering merupakan proses satu set data ke dalam himpunan bagian atau kelompok yang disebut dengan cluster. Contoh clustering dapat dilihat pada gambar berikut 2.36

### 2.7.3 Evaluasi dan Akurasi

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

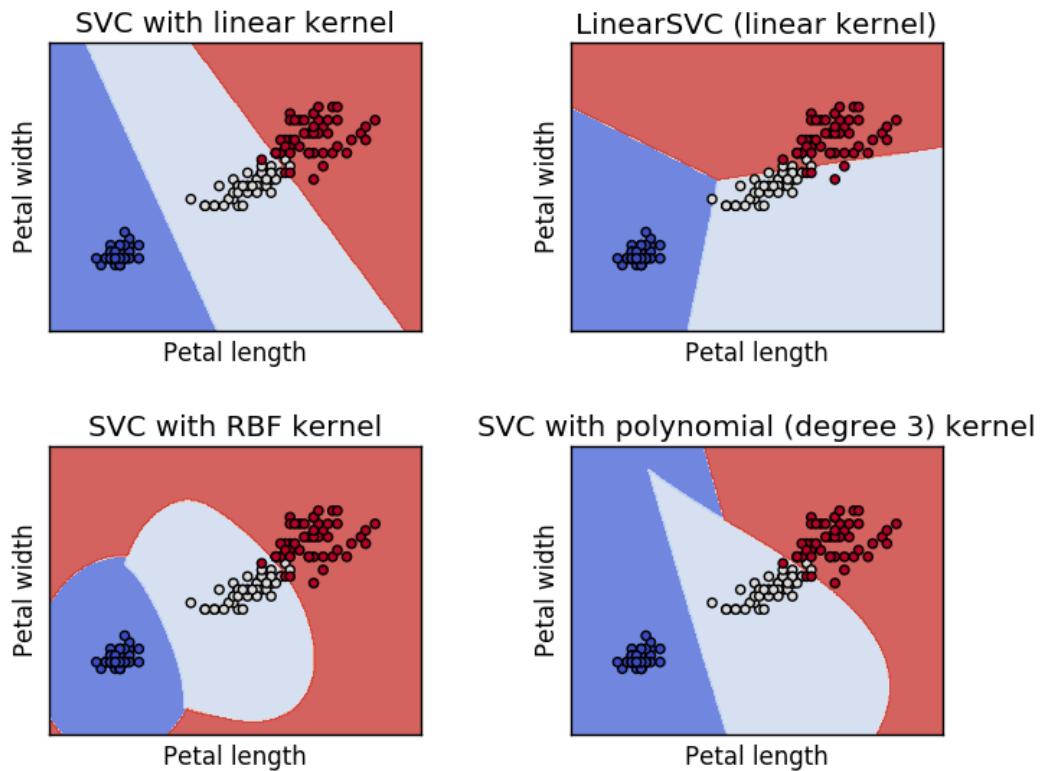


Figure 2.33: Supervised Learning.

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 mengacu pada kebingungan dalam model, tetapi matriks kebingungan ini bisa menjadi sedikit lebih sulit untuk dipahami ketika mereka menjadi sangat besar. Contohnya dapat dilihat pada gambar berikut ??

#### 2.7.4 Confusion Matrix

1. Terdapat beberapa cara untuk membuat dan membaca confusion matrix diantaranya, sebagai berikut
  - Tentukan pokok permasalahan dan atributnya, misal pendapatan dan pengeluaran.
  - Buat decission tree
  - Buat data testingnya

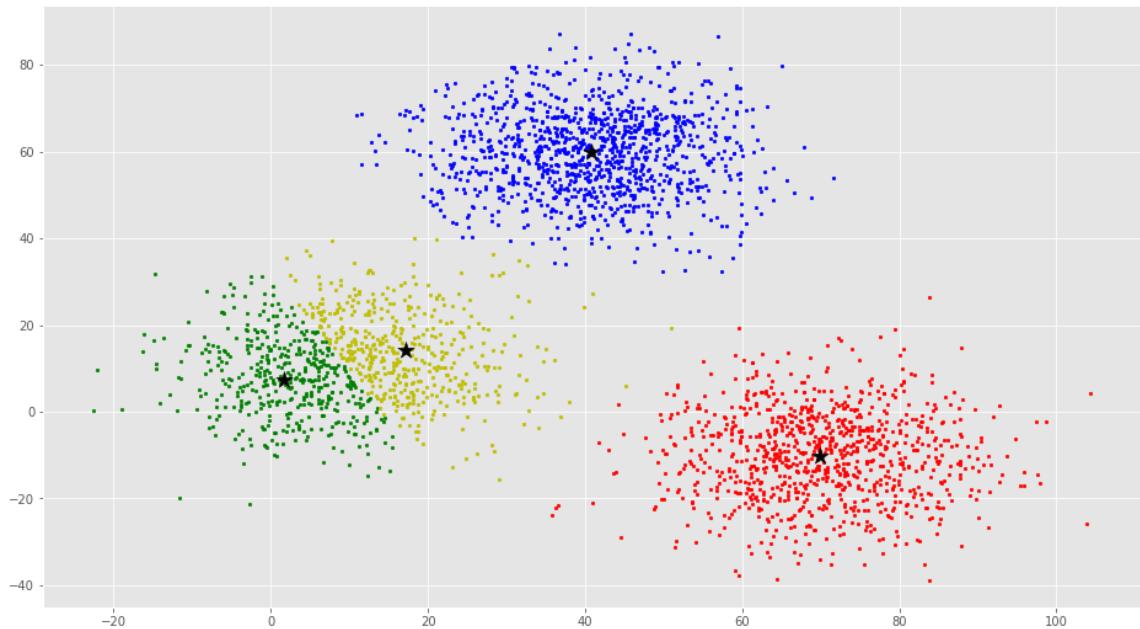


Figure 2.34: Unsupervised Learning.

- Lalu mencari nilai a, b ,c dan d. Misal a = 8, b = 2, c = 2, dan d = 6.
- Selanjutnya mencari nilai recall, precision, accuracy, dan error rate.

Berikut contoh dari confusion matrix

$$\text{Recall} = 6/(2+6) = 1,33$$

$$\text{Precision} = 6/(2+6) = 1,33$$

$$\text{Accuracy} = (8+6)/(8+2+2+6) = 0,8$$

$$\text{Error rate} = (2+2)/(8+2+2+6) = 0,22$$

### 2.7.5 Cara kerja K-Fold Cross Validation

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

- Total instance dibagi menjadi N bagian
- Fold yang pertama adalah bagian pertama menjadi data uji (testing) dan sisanya menjadi training data
- Lalu hitung akurasi berdasarkan porsi data tersebut dengan menggunakan persamaan
- Fold yang kedua adalah bagian ke dua menjadi data uji (testing) dan sisanya menjadi training data

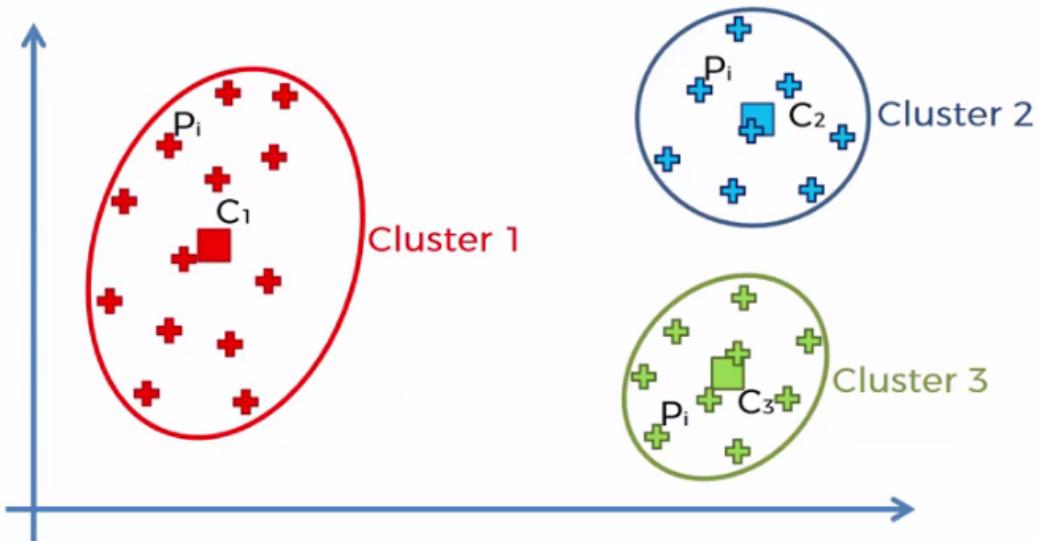


Figure 2.35: Clustering.

- Lalu hitung akurasi berdasarkan porsi data tersebut
- Dan selanjutnya hingga mencapai fold ke-4
- Terakhir hitung rata-rata akurasi K buah.

Ilustrasi dari K-Fold Cross Validation dapat dilihat pada gambar 2.37

### 2.7.6 Decision Tree

1. Decision tree adalah sebuah metode pembelajaran yang diawasi non-parametrik digunakan untuk klasifikasi dan regresi. Decision tree digunakan untuk membuat sebuah model yang dapat memprediksi variable dengan mempelajari aturan keputusan dengan ciri-ciri yang terdapat pada atribut tersebut. Sebagai contoh decision tree dapat melakukan prediksi apakah di bulan terdapat gravitasi atau bukan. Contohnya dapat dilihat pada gambar berikut 2.38

### 2.7.7 Information Gain dan Entropi

1. Information Gain adalah informasi atau kriteria dalam pembagian sebuah objek. Sebagai contoh misalnya information gain pada gambar laki-laki, atribut yang biasanya dimiliki pada gambar laki-laki diantaranya berambut pendek, berjakin, berjenggot, berkumis. Dalam beberapa hal terdapat perempuan yang memiliki rambut pendek, berkumis, dan berjenggot, namun dari parameter yang telah diidentifikasi bahwa gambar tersebut memiliki akurasi yang lebih tinggi

	Predict Alpukat	Predict Durian
True Alpukat	10	8
True Durian	2	15

Figure 2.36: Evaluasi dan Akurasi.

jadi dapat disimpulkan bahwa gambar tersebut adalah laki-laki. Untuk lebih jelasnya bisa dilihat dalam gambar 2.39

2. Entropi merupakan ukuran dari keacakan informasi, semakin tinggi entropi maka akan semakin sulit dalam menentukan suatu keputusan

## 2.8 Imron Sumadireja / 1164076

### 2.8.1 scikit-learn

Pada praktikum kali ini saya merubah beberapa variable yang terdapat pada source code dengan nama kota. Source code 1:

- Pada baris pertama dari source code tersebut menjelaskan bahwa kita akan import library pandas dengan merubah nama alias menjadi padalarang, seperti gambar berikut 2.40
- Pada baris kedua terdapat variable baru dengan nama dumai, dan akan membaca file dengan ekstensi .csv
- Berikut hasil yang di dapat dari source code berikut 2.41

Source code 2:

- Pada baris pertama menjelaskan bahwa kita akan menambahkan kolom lulus atau gagal. Data dari kolom tersebut akan berisi 1 dan 0. 1 Untuk mahasiswa

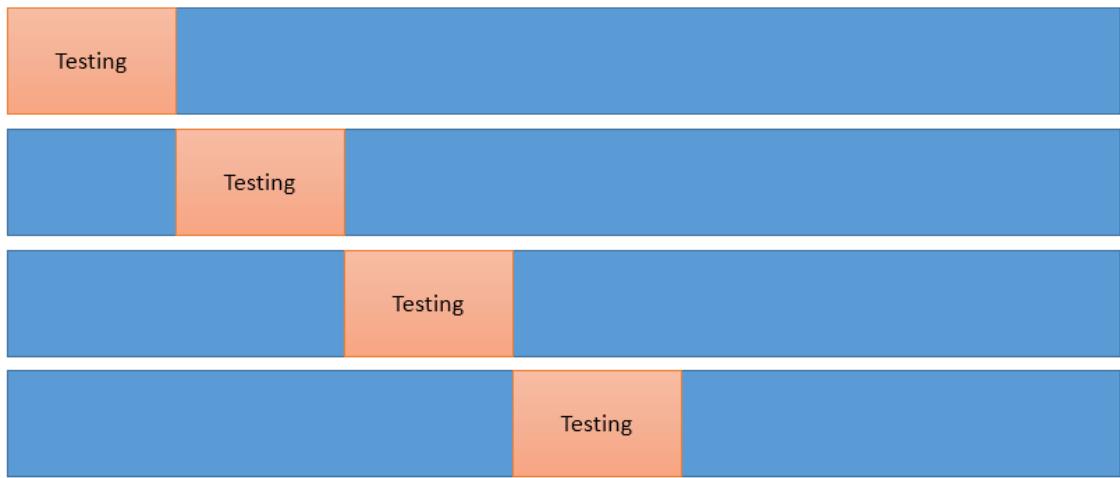


Figure 2.37: K-Fold Cross Validation.

yang dinyatakan lulus dan 0 untuk mahasiswa yang tidak lulus, seperti gambar berikut 2.42

- Pada baris kedua akan membuat data-data tersebut disusun secara berurutan sesuai atribut
- Pada baris ketiga berguna untuk menyinkronkan data yang terdapat pada source code pertama
- Berikut hasil yang di dapat dari source code berikut 2.43

Source code 3:

- Pada baris pertama menjelaskan bahwa dalam data tersebut akan memberikan tambahan kolom dengan atribut dengan isi 0 dan 1, seperti gambar berikut 2.44
- Pada baris kedua berguna untuk menyinkronkan data yang terdapat pada source code sebelumnya
- Berikut hasil yang di dapat dari source code berikut 2.45

Source code 4:

- Pada baris pertama menjelaskan bahwa variable dumai akan menjalankan fungsi sample dengan frac 1, seperti gambar berikut 2.46
- Pada baris kedua dan ketiga berguna untuk memberikan data training dan data testing dengan masing-masing nilai 500



Figure 2.38: Decision Tree.

- Pada baris keempat dan kelima berguna untuk melatih data training
- Pada baris keenam dan ketujuh berguna untuk melatih data testing
- Pada baris kedelapan dan kesembilan berguna untuk membuat sebuah keputusan dari hasil data training dan data testing
- Pada baris kesepuluh berguna untuk import library numpy
- Pada baris kesebelas berguna untuk menampilkan hasil data suatu keputusan tersebut
- Berikut hasil yang di dapat dari source code berikut 2.47

Source code 5:

- Pada baris pertama berguna untuk import library tree yang berguna untuk membuat keputusan dengan metode tree, seperti gambar berikut 2.48
- Pada baris kedua variable tangerang akan menjalankan fungsi tree decision
- Pada baris ketiga variable tangerang akan menjalankan fungsi tersebut menggunakan data training
- Berikut hasil yang di dapat dari source code berikut 2.49

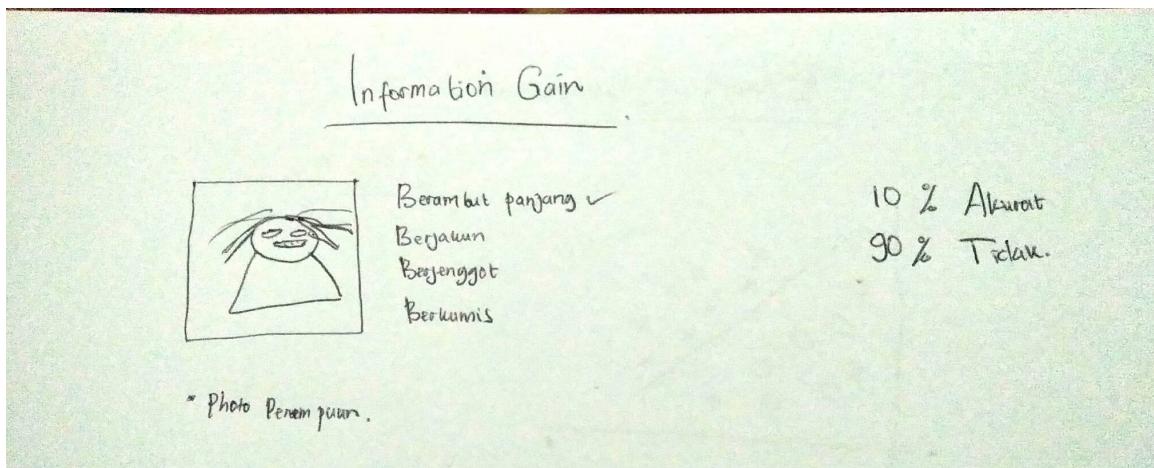


Figure 2.39: Information Gain dan Entropi.

```
# Load dataset (student Portuguese scores)
import pandas as padalarang
dumai = padalarang.read_csv('F:\Imron\Kuliah\Semester 6\Artificial Intelligence\praktikum\Python-Artificial-Intelligence-Projects-for-Beginners\Chapt
len(dumai)
```

Figure 2.40: Source Code.

Source code 6:

- Pada baris pertama berguna untuk import library graphviz, seperti pada gambar berikut 2.50
- Pada baris kedua berguna untuk membuat graphviz dari hasil data yang telah di latih pada source code sebelumnya
- Baris ketiga berguna untuk memanggil atribut dot untuk ditampilkan dalam bentuk graphic
- Berikut hasil yang di dapat dari source code berikut 2.51

Source code 7:

- Source code tersebut berguna untuk mengekspor representasi visual dalam bentuk PDF atau format lainnya, seperti gambar berikut 2.52
- Berikut hasil yang di dapat dari source code berikut 2.53

Source code 8:

- Source code berikut 2.54 berguna untuk memeriksa skor tree dengan menggunakan set pengujian yang telah di buat sebelumnya

```
In [2]: import pandas as padalarang
...: dumai = padalarang.read_csv('F:\Imron\Kuliah\Semester 6\Artificial
Intelligence\praktikum\Python-Artificial-Intelligence-Projects-for-Beginners
\Chapter01\dataset\student-por.csv', sep=';')
...: len(dumai)
Out[2]: 649
```

Figure 2.41: Source Code.

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

Figure 2.42: Source Code.

- Berikut hasil yang di dapat dari source code berikut 2.55

Source code 9:

- Pada baris pertama berguna untuk import library cross val score, seperti gambar berikut 2.56
- Baris kedua data yang telah dibuat sebelumnya akan kembali digunakan untuk memastikan rata-rata tersebut
- Pada baris ketiga akan menampilkan hasil rata-rata dari data tersebut
- Berikut hasil yang di dapat dari source code berikut 2.57

Source code 10:

- Source code berikut ini berfungsi untuk melakukan pengecekan lebih dalam lagi untuk menentukan keputusan yang lebih akurat dibandingkan dengan metode sebelumnya. Pada source code tersebut melakukan validasi silang, seperti gambar berikut 2.58
- Berikut hasil yang di dapat dari source code berikut 2.59

Source code 11:

- Source code tersebut menjelaskan bahwa untuk mendapatkan hasil keputusan yang akurat diperlukan training yang lebih banyak lagi dalam kasus ini depth acc memiliki nilai 19 dan 3. Proses yang dilakukan sama dengan proses sebelumnya yakni dengan menggunakan decision tree dan dari data hasil data training dan data testing, seperti gambar berikut 2.60

```

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

[5 rows x 31 columns]

```

Figure 2.43: Source Code.

```

# use one-hot encoding on categorical columns
dumai = padalarang.get_dummies(dumai, columns=['sex', 'school', 'address', 'famsize', 'Pstatus', 'Mjob', 'Fjob',
                                                'reason', 'guardian', 'schoolsup', 'famsup', 'paid', 'activities',
                                                'nursery', 'higher', 'internet', 'romantic'])
dumai.head()

```

Figure 2.44: Source Code.

- Berikut hasil yang di dapat dari source code berikut 2.61

Source code 12:

- Pada baris pertama berguna untuk import library matplotlib.pyplot, seperti gambar berikut 2.62
- Source code tersebut berguna untuk menampilkan diagram hasil keputusan pada pelatihan-pelatihan data sebelumnya
- Berikut hasil yang di dapat dari source code berikut 2.63

## 2.8.2 Penanganan Error

Dari percobaan yang telah dilakukan saya mengalami 2 kali error, berikut screenshot error serta penanganan yang saya dapat:

1. Screenshot error 2.64
2. Solusi dari permasalahan tersebut, kita tinggal memasukan direktori tempat file tersebut berada 2.65
3. Screenshot error 2.66

```

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

```

Figure 2.45: Source Code.

```

# shuffle rows
dumai = dumai.sample(frac=1)
# split training and testing data
dumai_train = dumai[:500]
dumai_test = dumai[500:]

dumai_train_att = dumai_train.drop(['pass'], axis=1)
dumai_train_pass = dumai_train['pass']

dumai_test_att = dumai_test.drop(['pass'], axis=1)
dumai_test_pass = dumai_test['pass']

dumai_att = dumai.drop(['pass'], axis=1)
dumai_pass = dumai['pass']

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

```

Figure 2.46: Source Code.

4. Solusi dari permasalahan tersebut, kita harus install library graphviz terlebih dahulu seperti gambar 2.67, berhubung saya sudah install maka gambarnya seperti itu.
5. Selanjutnya setalah install library graphviz selesai, kita masukkan path graphviz tersebut kedalam environment variables seperti gambar 2.68 agar dapat digunakan.
6. Setelah itu semua selesai, maka permasalahan pun sudah ditangani.

## 2.9 scikit-learn

HARI KEDUA ANDRI FAJAR SUNANDHAR 1164065

1. # load dataset (student Portuguese scores)

```

In [5]: dumai = dumai.sample(frac=1)
....: # split training and testing data
....: dumai_train = dumai[:500]
....: dumai_test = dumai[500:]
....:
....: dumai_train_att = dumai_train.drop(['pass'], axis=1)
....: dumai_train_pass = dumai_train['pass']
....:
....: dumai_test_att = dumai_test.drop(['pass'], axis=1)
....: dumai_test_pass = dumai_test['pass']
....:
....: dumai_att = dumai.drop(['pass'], axis=1)
....: dumai_pass = dumai['pass']
....:
....: # number of passing students in whole dataset:
....: import numpy as nabire
....: print("Passing: %d out of %d (%.2f%%)" % (nabire.sum(dumai_pass),
len(dumai_pass), 100*float(nabire.sum(dumai_pass)) / len(dumai_pass)))
Passing: 328 out of 649 (50.54%)

```

Figure 2.47: Source Code.

```

# fit a decision tree
from sklearn import tree
tangerang = tree.DecisionTreeClassifier(criterion="entropy", max_depth=5)
tangerang = tangerang.fit(dumai_train_att, dumai_train_pass)

```

Figure 2.48: Source Code.

```

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

```

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

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

```

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

```

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

```
In [6]: from sklearn import tree
.... tangerang = tree.DecisionTreeClassifier(criterion="entropy", max_depth=5)
.... tangerang = tangerang.fit(dumai_train_att, dumai_train_pass)
```

Figure 2.49: Source Code.

```
# visualize tree
import graphviz
dot_data = tree.export_graphviz(tangerang, out_file=None, label="all", impurity=False, proportion=True,
                                feature_names=list(dumai_train_att), class_names=["fail", "pass"],
                                filled=True, rounded=True)
graph = graphviz.Source(dot_data)
graph
```

Figure 2.50: Source Code.

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

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

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

### 4. # shuffle rows

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

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

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

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

```
# number of passing students in whole dataset:
import numpy as np
```

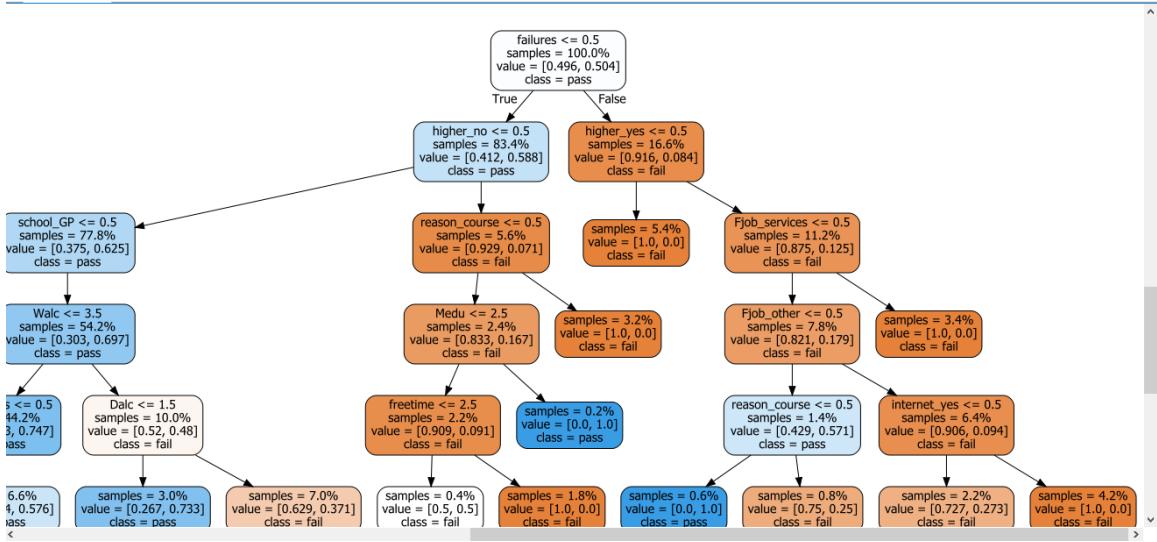


Figure 2.51: Source Code.

```
# save tree
tree.export_graphviz(tangerang, out_file="student-performance.dot", label="all", impurity=False, proportion=True,
                     feature_names=list(dumai_train_att), class_names=["fail", "pass"],
                     filled=True, rounded=True)
```

Figure 2.52: Source Code.

```
print("Passing: %d out of %d (%.2f%%)" % (np.sum(jeruk_pass), len(jeruk_pass), np.sum(jeruk_pass)/len(jeruk_pass)*100))
```

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

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

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

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

```
6. # visualize tree
import graphviz
```

```
In [8]: tree.export_graphviz(tangerang, out_file="student-performance.dot", label="all", impurity=False, proportion=True,
...:                                     feature_names=list(dumai_train_att),
...:                                     class_names=["fail", "pass"],
...:                                     filled=True, rounded=True)
```

Figure 2.53: Source Code.

```
tangerang.score(dumai_test_att, dumai_test_pass)
```

Figure 2.54: Source Code.

```
dot_data = tree.export_graphviz(semangka, out_file=None, label="all", impurity=False,
...:                                     feature_names=list(jeruk_train_att), class_names=["fail", "pass"],
...:                                     filled=True, rounded=True)
graph = graphviz.Source(dot_data)
graph
```

Mengimport Graphviz Sehingga akan muncul gambardiagram grafik bercabang.

#### 7. # save tree

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

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

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

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

#### 9. from sklearn.model\_selection import cross\_val\_score

```
scores = cross_val_score(semangka, jeruk_att, jeruk_pass, cv=5)
# show average score and +/- two standard deviations away (covering 95% of scores)
print("Accuracy: %0.2f (+/- %0.2f)" % (scores.mean(), scores.std() * 2))
```

```
In [9]: tangerang.score(dumai_test_att, dumai_test_pass)
Out[9]: 0.6778523489932886
```

Figure 2.55: Source Code.

```
from sklearn.model_selection import cross_val_score
scores = cross_val_score(tangerang, dumai_att, dumai_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))
```

Figure 2.56: Source Code.

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

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

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

```
11. depth_acc = np.empty((19,3), float)
i = 0
for max_depth in range(1, 20):
    semangka = tree.DecisionTreeClassifier(criterion="entropy", max_depth=max_
    scores = cross_val_score(semangka, jeruk_att, jeruk_pass, cv=5)
    depth_acc[i,0] = max_depth
    depth_acc[i,1] = scores.mean()
    depth_acc[i,2] = scores.std() * 2
    i += 1

depth_acc
```

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

```

In [10]: from sklearn.model_selection import cross_val_score
...: scores = cross_val_score(tangerang, dumai_att, dumai_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: 0.70 (+/- 0.06)

In [11]:

```

Figure 2.57: Source Code.

```

for max_depth in range(1, 20):
    tangerang = tree.DecisionTreeClassifier(criterion="entropy", max_depth=max_depth)
    scores = cross_val_score(tangerang, dumai_att, dumai_pass, cv=5)
    print("Max depth: %d, Accuracy: %0.2f (+/- %0.2f)" % (max_depth, scores.mean(), scores.std() * 2))

```

Figure 2.58: Source Code.

```

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

```

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

## 2.10 Penanganan Error

Hari Kedua Andri fajar Sunandhar 1164065

### 2.10.1 Error Graphviz

1. error yang didapatkan saat menjalankan Graphviz
2. Kode erornya adalah ModuleNotFoundError. Eror ini terjadi karena module named Graphviz nya tidak ada.

```
In [11]: for max_depth in range(1, 20):
...:     tangerang =
tree.DecisionTreeClassifier(criterion="entropy", max_depth=max_depth)
...:     scores = cross_val_score(tangerang, dumai_att,
dumai_pass, cv=5)
...:     print("Max depth: %d, Accuracy: %0.2f (+/- %0.2f)" %
(max_depth, scores.mean(), scores.std() * 2))
Max depth: 1, Accuracy: 0.64 (+/- 0.07)
Max depth: 2, Accuracy: 0.69 (+/- 0.08)
Max depth: 3, Accuracy: 0.70 (+/- 0.06)
Max depth: 4, Accuracy: 0.69 (+/- 0.04)
Max depth: 5, Accuracy: 0.70 (+/- 0.07)
Max depth: 6, Accuracy: 0.67 (+/- 0.06)
Max depth: 7, Accuracy: 0.67 (+/- 0.08)
Max depth: 8, Accuracy: 0.64 (+/- 0.06)
Max depth: 9, Accuracy: 0.67 (+/- 0.04)
Max depth: 10, Accuracy: 0.65 (+/- 0.05)
Max depth: 11, Accuracy: 0.65 (+/- 0.08)
Max depth: 12, Accuracy: 0.65 (+/- 0.08)
Max depth: 13, Accuracy: 0.62 (+/- 0.04)
Max depth: 14, Accuracy: 0.62 (+/- 0.05)
Max depth: 15, Accuracy: 0.63 (+/- 0.05)
Max depth: 16, Accuracy: 0.63 (+/- 0.03)
Max depth: 17, Accuracy: 0.63 (+/- 0.03)
Max depth: 18, Accuracy: 0.61 (+/- 0.04)
Max depth: 19, Accuracy: 0.63 (+/- 0.07)
```

Figure 2.59: Source Code.

3. Solusi yang bisa dilakukan untuk mengatasi eror tersebut adalah sebagai berikut :

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

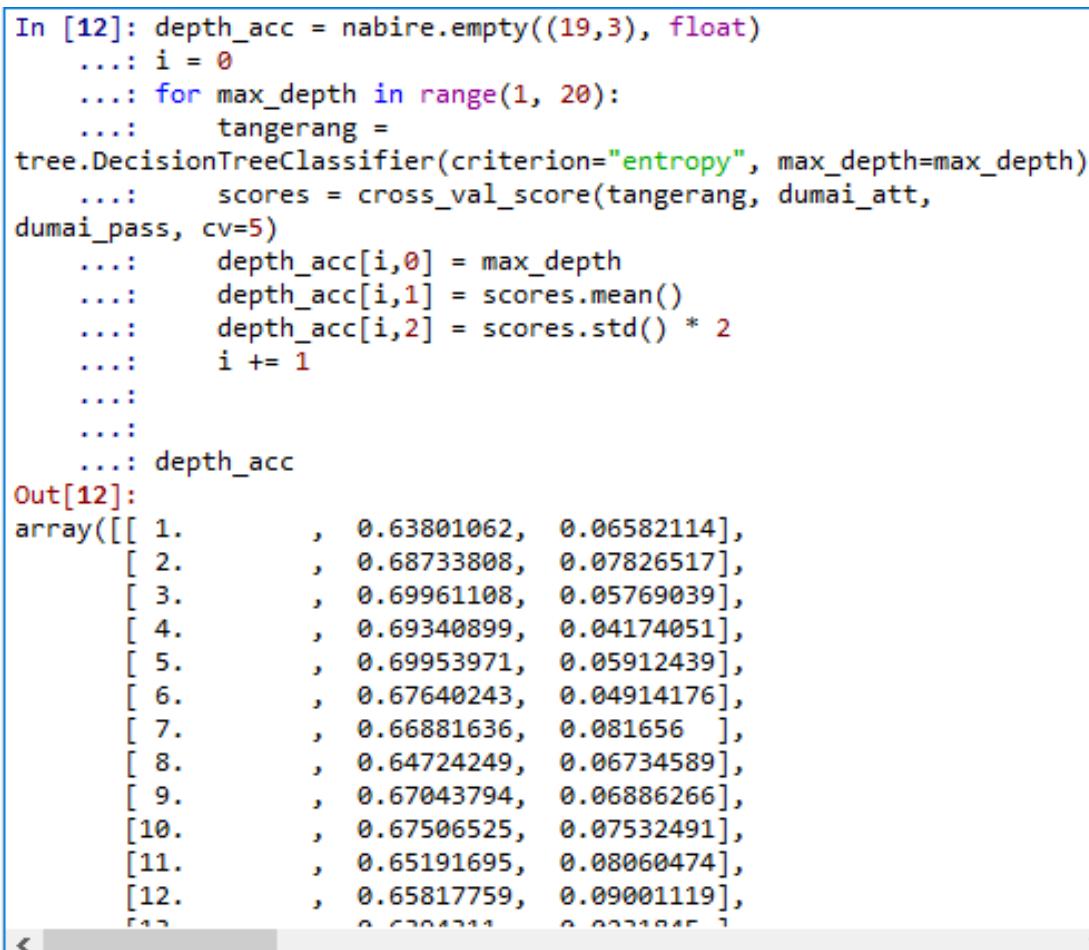
```

depth_acc = nabire.empty((19,3), float)
i = 0
for max_depth in range(1, 20):
    tangerang = tree.DecisionTreeClassifier(criterion="entropy", max_depth=max_depth)
    scores = cross_val_score(tangerang, dumai_att, dumai_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

```

Figure 2.60: Source Code.



```

In [12]: depth_acc = nabire.empty((19,3), float)
...: i = 0
...: for max_depth in range(1, 20):
...:     tangerang =
tree.DecisionTreeClassifier(criterion="entropy", max_depth=max_depth)
...:     scores = cross_val_score(tangerang, dumai_att,
dumai_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[12]:
array([[ 1.        ,  0.63801062,  0.06582114],
       [ 2.        ,  0.68733808,  0.07826517],
       [ 3.        ,  0.69961108,  0.05769039],
       [ 4.        ,  0.69340899,  0.04174051],
       [ 5.        ,  0.69953971,  0.05912439],
       [ 6.        ,  0.67640243,  0.04914176],
       [ 7.        ,  0.66881636,  0.081656  ],
       [ 8.        ,  0.64724249,  0.06734589],
       [ 9.        ,  0.67043794,  0.06886266],
      [10.        ,  0.67506525,  0.07532491],
      [11.        ,  0.65191695,  0.08060474],
      [12.        ,  0.65817759,  0.09001119],
      [13.        ,  0.67042111,  0.07718451]
      ])

```

Figure 2.61: Source Code.

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

Figure 2.62: Source Code.

```
In [13]: import matplotlib.pyplot as plt
...: fig, ax = plt.subplots()
...: ax.errorbar(depth_acc[:,0], depth_acc[:,1], yerr=depth_acc[:,2])
...: plt.show()
```

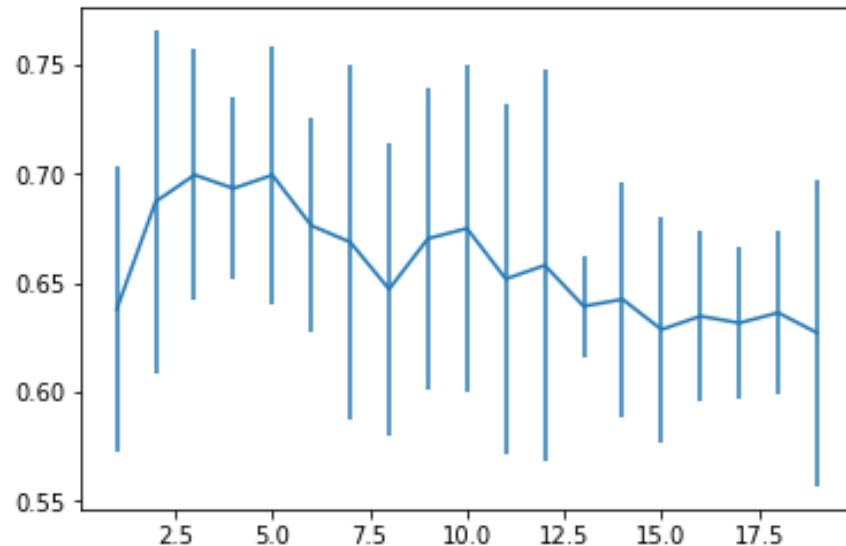


Figure 2.63: Source Code.

```
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
pandas._libs.parsers.TextReader._setup_parser_source

FileNotFoundException: File b'student-por.csv' does not exist

In [2]:
```

Figure 2.64: Error.

```
# Load dataset (student Portuguese scores)
import pandas as pd
dumai = pd.read_csv('F:\Imron\Kuliah\Semester 6\Artificial Intelligence\praktikum\Python-Artificial-Intelligence-Projects-for-Beginners\Chapter01\len(dumai)|
```

Figure 2.65: Resolve.

```
File "C:\Users\NS\Anaconda3\lib\site-packages\graphviz\backend.py", line 206, in pipe
    out, _ = run(cmd, input=data, capture_output=True, check=True, quiet=quiet)

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

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

Out[9]: <graphviz.files.Source at 0x9709fd0>
```

Figure 2.66: Error.

```
Administrator: Command Prompt
Microsoft Windows [Version 10.0.17763.316]
(c) 2018 Microsoft Corporation. All rights reserved.

C:\WINDOWS\system32>conda install graphviz
Collecting package metadata: done
Solving environment: done

# All requested packages already installed.

C:\WINDOWS\system32>
```

Figure 2.67: Resolve.

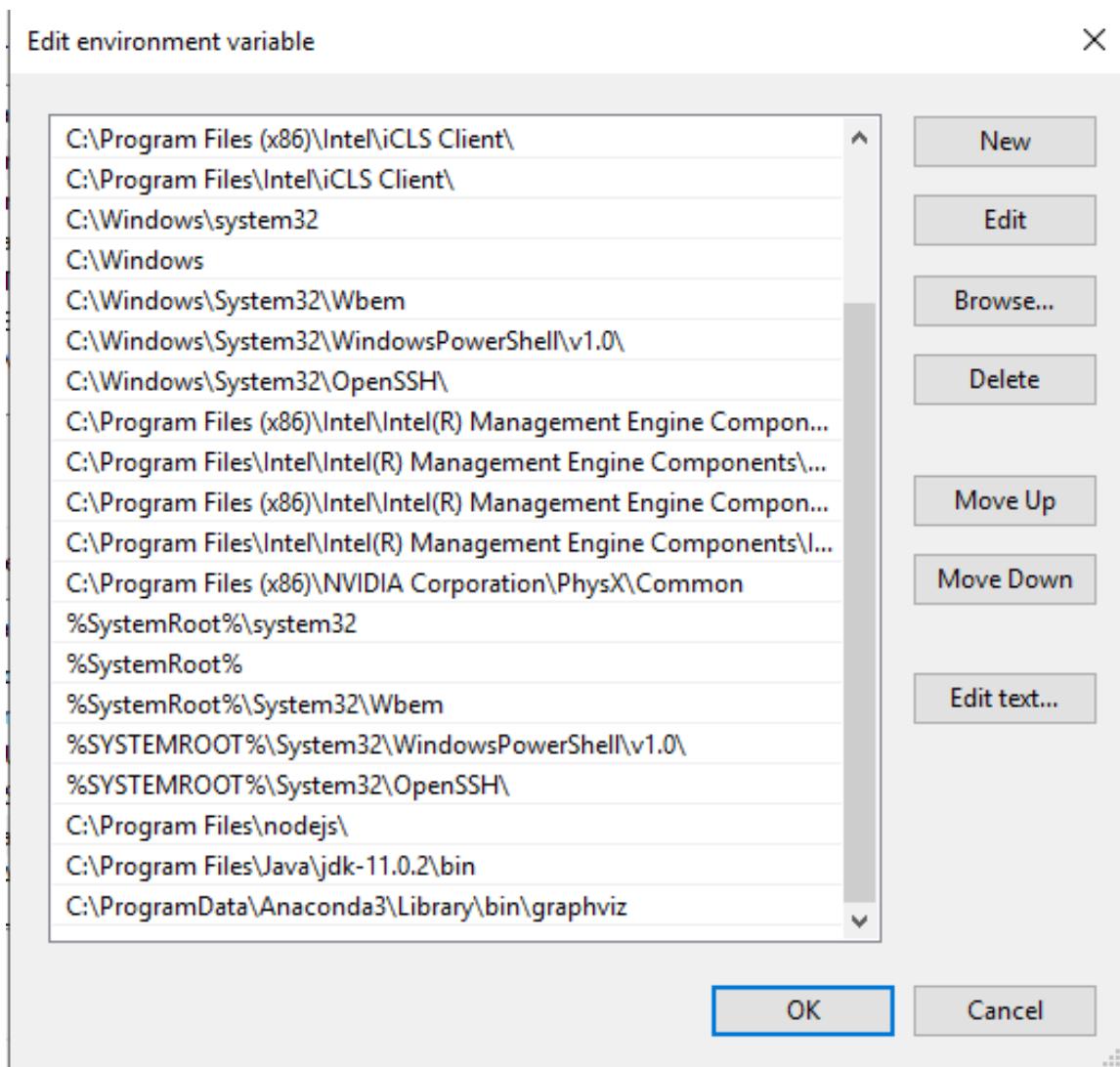


Figure 2.68: Resolve.

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

Figure 2.69: Loading Dataset

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

Figure 2.70: Generate Binary Label

```
In [9]: jeruk = ape1.get_dummies(jeruk, columns=['sex', 'school', 'address', 'famsize',
'pstatus', 'Mjob', 'Fjob',
..., 'paid', 'activities',
..., jeruk.head())
Out[9]:
   age  Medu  Fedu    ...  internet_yes  romantic_no  romantic_yes
0    18     4     4    ...          0            1            0
1    17     1     1    ...          1            1            0
2    15     1     1    ...          1            1            0
3    15     4     2    ...          1            0            1
4    16     3     3    ...          0            1            0
[5 rows x 57 columns]
```

Figure 2.71: One-hot Encoding

```
In [10]: jeruk = jeruk.sample(frac=1)
.... # split training and testing data
.... jeruk_train = jeruk[:500]
.... jeruk_test = jeruk[500:]
....
.... jeruk_train_att = jeruk_train.drop(['pass'], axis=1)
.... jeruk_train_pass = jeruk_train['pass']
....
.... jeruk_test_att = jeruk_test.drop(['pass'], axis=1)
.... jeruk_test_pass = jeruk_test['pass']
....
.... jeruk_att = jeruk.drop(['pass'], axis=1)
.... jeruk_pass = jeruk['pass']
....
.... # number of passing students in whole dataset:
.... import numpy as np
.... print("Passing: %d out of %d (%.2f%%)" % (np.sum(jeruk_pass), len(jeruk_pass),
100*float(np.sum(jeruk_pass)) / len(jeruk_pass)))
Passing: 160 out of 395 (42.03%)
```

Figure 2.72: Shuffle Rows

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

Figure 2.73: Fit Decision Tree

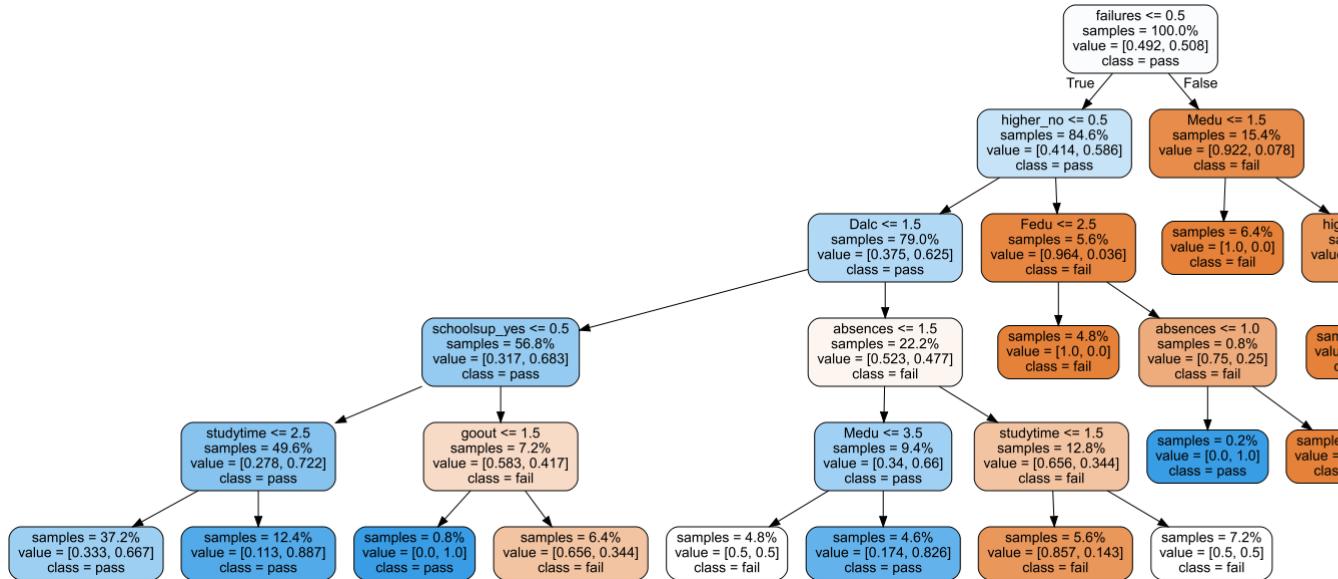


Figure 2.74: Fit Decision Tree

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

Figure 2.75: Fit Decision Tree

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

Figure 2.76: Score

```
In [15]: from sklearn.model_selection import cross_val_score
...: scores = cross_val_score(semangka, jeruk_att, jeruk_pass,
cv=5)
...: # show average score and +/- two standard deviations away
(covering 95% of scores)
...: print("Accuracy: %0.2f (+/- %0.2f)" % (scores.mean(),
scores.std() * 2))
Accuracy: 0.58 (+/- 0.04)
```

Figure 2.77: Cross Val Score

```
In [16]: for max_depth in range(1, 20):
...:     semangka = tree.DecisionTreeClassifier(criterion="entropy", max_depth=max_depth)
...:     scores = cross_val_score(semangka, jeruk_att, jeruk_pass, cv=5)
...:     print("Max depth: %d, Accuracy: %0.2f (+/- %0.2f)" % (max_depth, scores.mean(), scores.std() * 2))

Max depth: 1, Accuracy: 0.58 (+/- 0.01)
Max depth: 2, Accuracy: 0.58 (+/- 0.05)
Max depth: 3, Accuracy: 0.58 (+/- 0.02)
Max depth: 4, Accuracy: 0.59 (+/- 0.08)
Max depth: 5, Accuracy: 0.58 (+/- 0.03)
Max depth: 6, Accuracy: 0.58 (+/- 0.08)
Max depth: 7, Accuracy: 0.57 (+/- 0.10)
Max depth: 8, Accuracy: 0.60 (+/- 0.10)
Max depth: 9, Accuracy: 0.60 (+/- 0.09)
Max depth: 10, Accuracy: 0.62 (+/- 0.05)
Max depth: 11, Accuracy: 0.61 (+/- 0.08)
Max depth: 12, Accuracy: 0.63 (+/- 0.08)
Max depth: 13, Accuracy: 0.60 (+/- 0.12)
Max depth: 14, Accuracy: 0.59 (+/- 0.07)
Max depth: 15, Accuracy: 0.62 (+/- 0.07)
Max depth: 16, Accuracy: 0.59 (+/- 0.05)
Max depth: 17, Accuracy: 0.62 (+/- 0.05)
Max depth: 18, Accuracy: 0.62 (+/- 0.09)
Max depth: 19, Accuracy: 0.62 (+/- 0.07)
```

Figure 2.78: Max Depth

```
In [17]: depth_acc = np.empty((19,3), float)
...: i = 0
...: for max_depth in range(1, 20):
...:     semangka = tree.DecisionTreeClassifier(criterion="entropy", max_depth=max_depth)
...:     scores = cross_val_score(semangka, jeruk_att, jeruk_pass, cv=5)
...:     depth_acc[i,0] = max_depth
...:     depth_acc[i,1] = scores.mean()
...:     depth_acc[i,2] = scores.std() * 2
...:     i += 1
...:
...: depth_acc
Out[17]:
array([[1.00000000e+00, 5.79751704e-01, 6.30768599e-03],
[2.00000000e+00, 5.79817429e-01, 5.40865102e-02],
[3.00000000e+00, 5.82348264e-01, 2.2355177e-02],
[4.00000000e+00, 5.84752515e-01, 7.32490810e-02],
[5.00000000e+00, 5.84719247e-01, 4.05630974e-02],
[6.00000000e+00, 5.79462837e-01, 7.53158175e-02],
[7.00000000e+00, 5.71772152e-01, 9.89792479e-02],
[8.00000000e+00, 6.02088608e-01, 1.28171591e-01],
[9.00000000e+00, 6.02249270e-01, 9.28901055e-02],
[1.00000000e+01, 6.17666342e-01, 6.17102083e-02],
[1.10000000e+01, 6.02378286e-01, 5.13011736e-02],
[1.20000000e+01, 6.15198799e-01, 5.15762620e-02],
[1.30000000e+01, 6.05072217e-01, 5.81263071e-02],
[1.40000000e+01, 6.15038137e-01, 5.79996256e-02],
[1.50000000e+01, 5.97380721e-01, 6.43910242e-02],
[1.60000000e+01, 5.82190036e-01, 7.12462584e-02],
[1.70000000e+01, 5.97284972e-01, 8.33497031e-02],
[1.80000000e+01, 6.25166342e-01, 6.78027722e-02],
[1.90000000e+01, 5.97285784e-01, 1.01208800e-01]])
```

Figure 2.79: Depth in Range

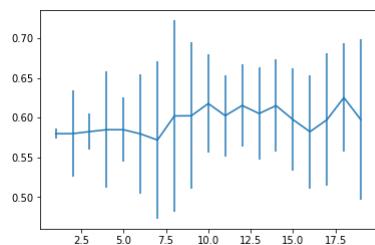


Figure 2.80: Matplotlib

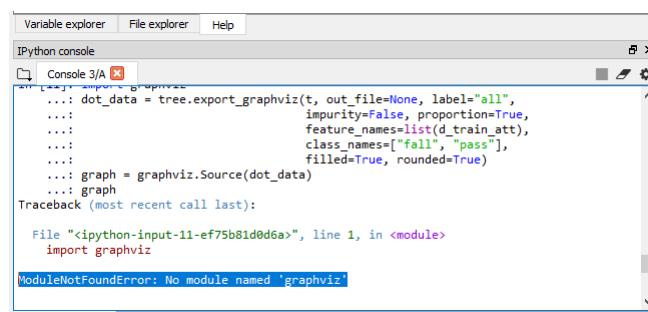


Figure 2.81: Error Graphviz

```

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

```

Figure 2.82: install Graphviz

```

C:\Users\ACER>conda install pip
Collecting package metadata: done
Solving environment: done
# All requested packages already installed.

```

Figure 2.83: Solving Environment

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

## Package Plan ##

environment location: C:\Users\ACER\Anaconda3

added / updated specs:
- python-graphviz

The following packages will be downloaded:
package          |      build
-----|-----
graphviz-2.38   | hfa6e2cd_3    37.7 MB
python-graphviz-0.8.4 | py36_1        28 KB
-----|-----
                           Total:   37.8 MB

The following NEW packages will be INSTALLED:
graphviz          pkgs/main/win-32::graphviz-2.38-hfa6e2cd_3
python-graphviz   pkgs/main/win-32::python-graphviz-0.8.4-py36_1

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

Figure 2.84: Evaluasi Eror

# **Chapter 3**

## **Methods**

### **3.1 The data**

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

### **3.2 Method 1**

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

### **3.3 Method 2**

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

### **3.4 Yusniar Nur Syarif Sidiq/1164089**

#### **3.4.1 Teori**

1. Random Forest merupakan algoritma yang digunakan terhadap klasifikasi data dalam jumlah yang besar. Klasifikasi pada random forest dilakukan dengan penggabungan dicision tree dengan melakuakn training terhadap sempel data yang dimiliki. Semakin banyak dicision tree maka data yang di dapat akan semakin akurat. Untuk gambar Random Forest dapat dilihat pada figure 3.1
2. Pertama download dataset terlebih dahulu lalu buka dengan menggunakan software spyder guna melihat isi dari dataset tersebut. Data tersebut memiliki extensi file bernama .txt dan didalamnya terdapat class dari field. Misalnya saja pada data jenis burung memiliki file index dan angka, dimana index berisi angka yang memiliki makna berupa jenis burung atau bahkan nama burung

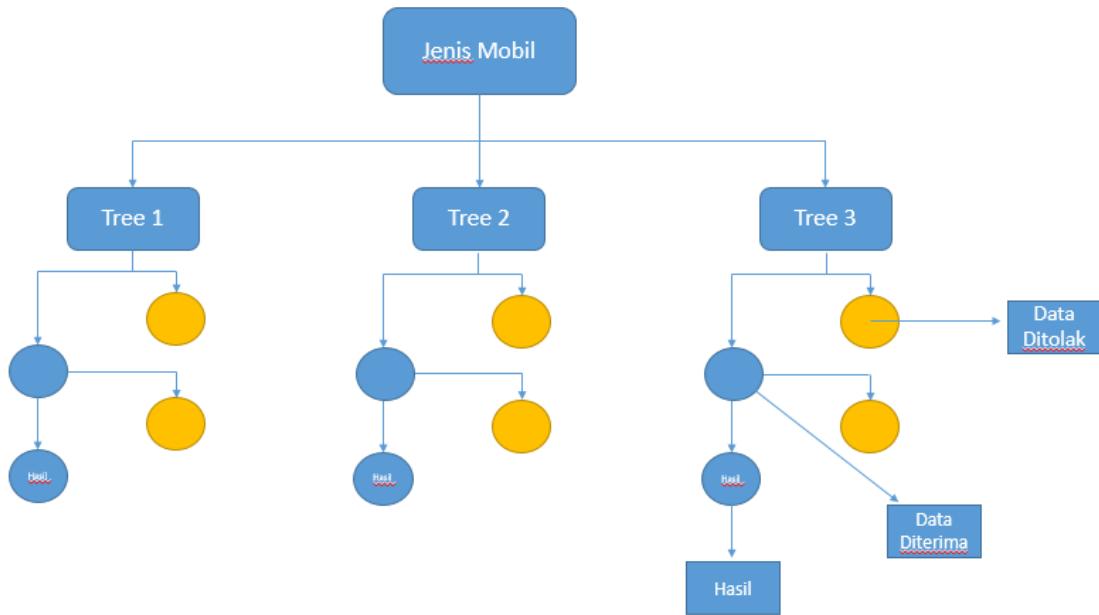


Figure 3.1: Random Forest.

sedangkan field memiliki isi nilai berupa 0 dan 1 yang dimana sifatnya boolean atau Ya dan Tidak. Hal ini dikarenakan komputer hanya dapat membaca bilangan biner maka dari itu field yang di isikan berupa angka. Artinya angka 0 berarti tidak dan angka 1 berarti Ya.

3. Cross Validation adalah sebuah teknik validasi model yang digunakan untuk menilai bagaimana hasil analisis statistik akan digeneralisasi ke kumpulan data independen. Cross validation digunakan dengan tujuan prediksi, dan bila kita ingin memperkirakan seberapa akurat model model prediksi yang dilakukan dalam sebuah praktek. Tujuan dari cross validation yaitu untuk mendefinisikan dataset guna menguji dalam fase pelatihan untuk membatasi masalah seperti overfitting dan underfitting serta mendapatkan wawasan tentang bagaimana model akan digeneralisasikan ke set data independen.
4. Dimana Score 44 % diperoleh dari hasil pengelahan dataset jenis burung. Di mana akan dilakukan proses pembagian data testing dan data training lalu diproses dan menghasilkan score sebanyak 44 % dimana menjelaskan bahwa score tersebut digunakan sebagai pembanding dalam tingkat keakuratannya. Pada dicision tree akan memperoleh data lebih kecil yaitu sebanyak 27 % hal ini dikarenakan data yang diolah menggunakan dicision tree dibagi menjadi beberapa tree dan lalu disimpulkan untuk mendapatkan data yang akurat. Pada

SVM akan memperoleh score sebanyak 29 % hal ini dikarenakan data yang dimiliki masih bernilai netral sehingga tingkat keakuratannya masih belum jelas.

5. Untuk membaca confusion matriks dapat menggunakan source code sebagai berikut,

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

Dimana numpy akan mengurus semua data yang berhubungan dengan matrix. Pada source code tersebut digunakan dalam melakukan read pada dataset burung dengan menggunakan metode confusion matrix. Dalam confusion matrix memiliki 4 istilah yaitu True Positive yang merupakan data positif yang terditeksi benar, True Negatif yang merupakan data negatif akan tetapi terditeksi benar, False Positif merupakan data negatif namun terditeksi sebagai data positif, False Negatif merupakan data positif namun terditeksi sebagai data negatif. Adapun contoh hasil read dataset menggunakan confusion matrix dapat dilihat pada figure 3.2

		True Values	
		True	False
Prediction	True	TP Correct result	FP Unexpected result
	False	FN Missing result	TN Correct absence of result

Figure 3.2: Confusion Matrix.

6. Voting merupakan proses pemilihan dari tree yang dimana akan dimunculkan hasilnya dan disimpulkan menjadi informasi yang pasti. Untuk kebih jelasnya saya akan memberikan sebuah contoh bagaimana voting berjalan.

Dimana ditunjukkan pada figure 3.3 terdapat 3 tree. Dalam tree tersebut akan dilakukan proses voting. Saya akan memberikan contoh kasus, dimana akan

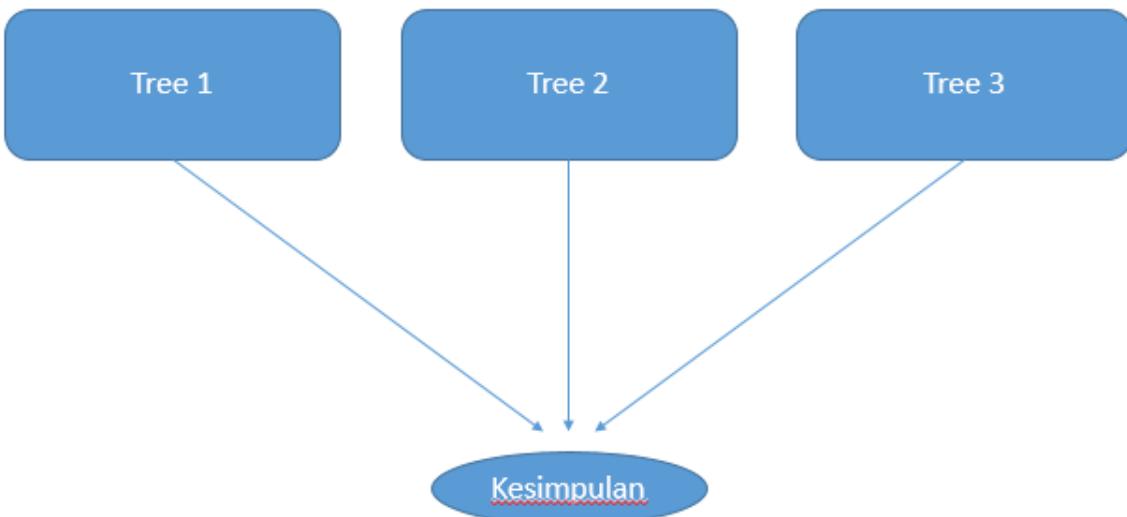


Figure 3.3: Voting.

diadakan voting untuk menentukan sebuah mobil. Dalam tree akan diberikan sejumlah data misalnya saja data tersebut berupa gambar, yang dimana data tersebut akan dipilih dengan cara voting. Hasil voting akhir dari setiap tree menunjukkan mobil jazz, yang berarti kesimpulan dari data yang telah diberikan menyatakan gambar tersebut adalah mobil jazz. Bagaimana apabila terjadi perbedaan data misalnya saja pada tree 1 dan 2 menyatakan mobil jazz sedangkan pada tree 3 menyatakan mobil yaris, maka kesimpulan yang di ambil adalah mobil jazz dikarenakan hasil voting terbanyak adalah mobil jazz.

### 3.4.2 Praktek Program/Yusniar Nur Syarif Sidiq/1164089

#### 1. Aplikasi Sederhana Dengan Menggunakan Pandas

Perhatikan source code berikut ini,

```
import pandas as sidiq
```

Untuk source code diatas menjelaskan bahwa akan melakukan import pada library pandas dan di rename menjadi sidiq. Selanjutnya kita akan membuat varibel yang bernama mobil, dan mengisikan variabel tersebut dengan nama-nama mobil, perhatikan source code dibawah.

```
mobil = {"Nama Mobil" : ['Ferari','Lamorgini','Civic','Mobilio']}
```

Selanjutnya kita akan membuat satu lagi variabel bernama yn, dimana variabel tersebut akan membuat DataFrame dari library pandas dan akan memanggil variabel mobil. Perhatikan source code dibawah.

```
yn = sidiq.DataFrame(mobil)
```

Selanjutnya kita panggil variabel yn untuk memunculkan data dari variabel mobil dengan cara print. Perhatikan source code dibawah.

```
print ('Yusniar Punya Mobil ' + yn)
```

Jalankan source code tersebut dengan menggunakan spyder, sehingga hasilnya akan nampak seperti pada figure 3.4

Name	Type	Size	Value
mobil	dict	1	{'Nama Mobil':['Ferari', 'Lamorgini', 'Civic', 'Mobilio']}
yn	DataFrame	(4, 1)	Column names: Nama Mobil

yn - DataFrame

Index	Nama Mobil
0	Ferari
1	Lamorgini
2	Civic
3	Mobilio

	Nama Mobil
0	Yusniar Punya Mobil Ferari
1	Yusniar Punya Mobil Lamorgini
2	Yusniar Punya Mobil Civic
3	Yusniar Punya Mobil Mobilio

Figure 3.4: Aplikasi Sederhana Dengan Pandas.

## 2. Aplikasi Sederhana dengan menggunakan numpy

Perhatikan source code dibawah ini,

```
import numpy as sidiq
```

Source Code diatas menjeaskan akan melakukan import pada library numpy dan di rename dengan sidiq. Selanjutnya kita akan membuat matrix dengan numpy dengan menggunakan fungsi eye. Perhatikan source code dibawah,

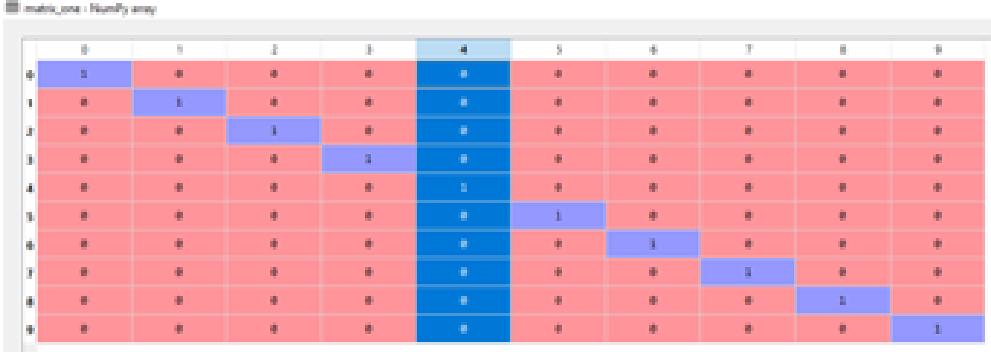
```
matrix_one = sidiq.eye(10)
matrix_one
```

Selanjutnya kita akan memanggil matrix identitas 10x10.

```
print (matrix_one)
```

Fungsi eye disini berguna untuk memanggil matrix identitas dengan jumlah kolom dan baris sesuai yang ditentukan. Disini saya telah menentukan 10 kolom dan baris maka dari itu hasilnya akan memunculkan matrix identitas 10x10, untuk lebih jelasnya dapat dilihat pada figure 3.5

Name	Type	Size	Value
matrix_one	float64	(10, 10)	<pre>[[1.  0.  0.  ...  0.  0.  0.]  [0.  1.  0.  ...  0.  0.  0.]  [0.  0.  1.  ...  0.  0.  0.]  [0.  0.  0.  1.  ...  0.  0.  0.]  [0.  0.  0.  0.  1.  ...  0.  0.  0.]  [0.  0.  0.  0.  0.  1.  ...  0.  0.  0.]  [0.  0.  0.  0.  0.  0.  1.  ...  0.  0.  0.]  [0.  0.  0.  0.  0.  0.  0.  1.  ...  0.  0.  0.]  [0.  0.  0.  0.  0.  0.  0.  0.  1.  ...  0.  0.  0.]  [0.  0.  0.  0.  0.  0.  0.  0.  0.  1.  ...  0.  0.  0.]</pre>

```
[ [1.  0.  0.  0.  0.  0.  0.  0.  0.  0.]
 [0.  1.  0.  0.  0.  0.  0.  0.  0.  0.]
 [0.  0.  1.  0.  0.  0.  0.  0.  0.  0.]
 [0.  0.  0.  1.  0.  0.  0.  0.  0.  0.]
 [0.  0.  0.  0.  1.  0.  0.  0.  0.  0.]
 [0.  0.  0.  0.  0.  1.  0.  0.  0.  0.]
 [0.  0.  0.  0.  0.  0.  1.  0.  0.  0.]
 [0.  0.  0.  0.  0.  0.  0.  1.  0.  0.]
 [0.  0.  0.  0.  0.  0.  0.  0.  1.  0.]
 [0.  0.  0.  0.  0.  0.  0.  0.  0.  1.] ]
```

Figure 3.5: Aplikasi Sederhana Dengan Numpy.

### 3. Aplikasi sederhana dengan menggunakan Matplotlib

Perhatikan source code dibawah ini,

```
import matplotlib.pyplot as sidiq

sidiq.plot([1,4,2,4,5,2,1])
sidiq.ylabel('YusniarNS 1164089')
sidiq.show()
```

- Pada baris pertama akan melakukan import library Matplotlib dan di rename menjadi sidiq
- Pada baris kedua kita akan memberikan nilai plot atau grafik pada library sidiq
- Pada baris ketiga kita akan memberikan nama atau label pada grafik tersebut
- Pada baris terakhir kita akan melakukan show sehingga grafik dapat kita lihat

Jalankan source code diatas dengan spyder sehingga hasilnya akan nampak seperti pada figure 3.6

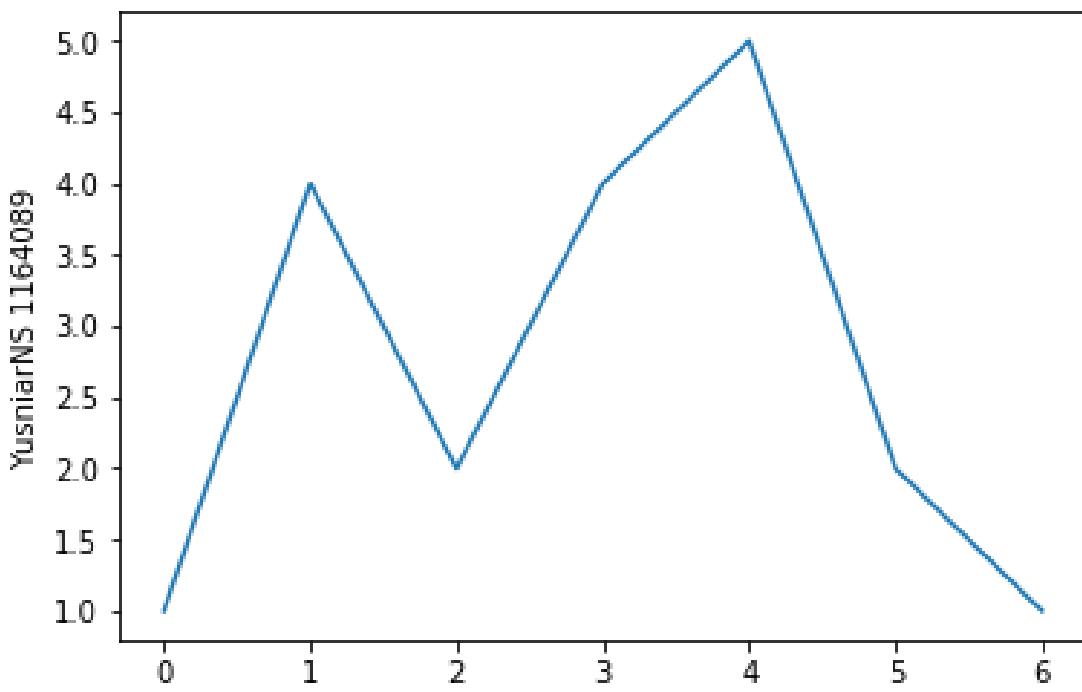


Figure 3.6: Aplikasi Sederhana Dengan Matplotlib Grafik.

Selanjutnya perhatikan source code berikut ini,

```

import matplotlib.pyplot as sidiq

y = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
n = [10, 9, 8, 7, 6, 5, 4, 3, 2, 1]
sidiq.bar(y,n)
sidiq.ylabel('YusniarNS 1164089')
sidiq.show()

```

Pada baris pertama akan melakukan import library Matplotlib dan di rename menjadi sidiq.Pada baris kedua dan ketiga kita akan menentukan nilai pada variabel y dan n.Pada baris kempat kita akan membuat diagram batang dengan fungsi bar.Pada baris kelima kita akan memberikan nama atau label pada diagram batang tersebut.Pada baris terakhir kita akan melakukan show sehingga kita dapat melihat hasilnya. Jalankan source code tersebut di dalam spyder maka hasilnya akan nampak seperti pada figure 3.7

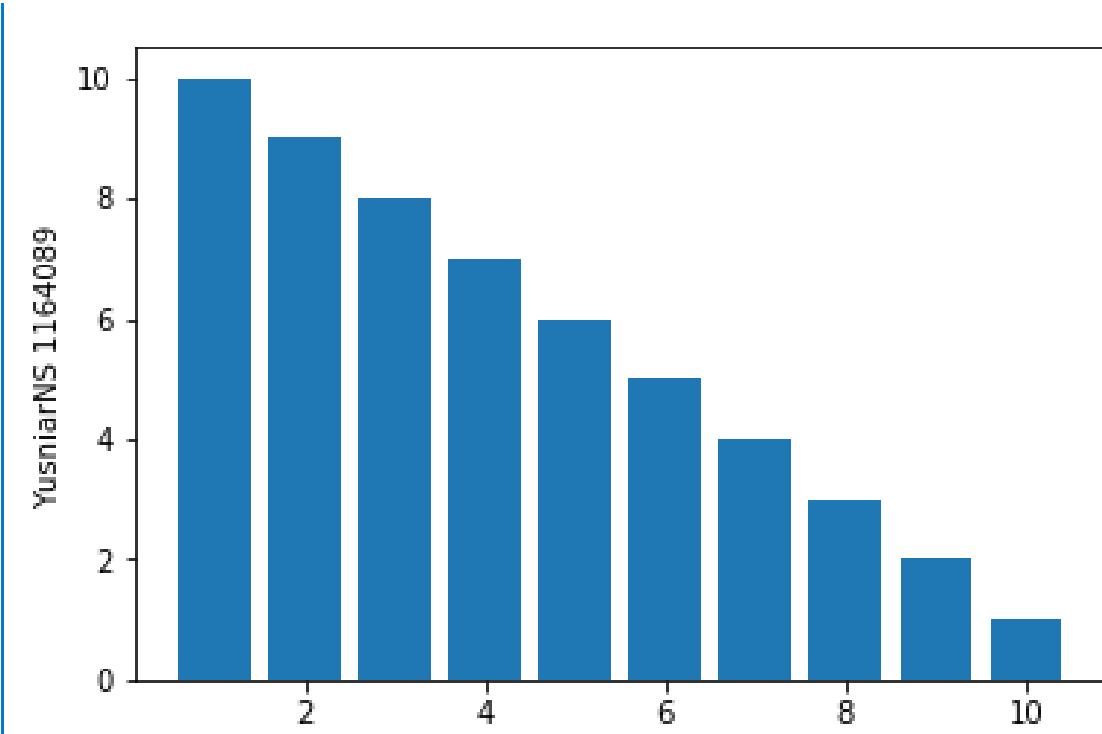


Figure 3.7: Aplikasi Sederhana Dengan Matplotlib Diagram Batang.

#### 4. Klasifikasi Random Forest

- Source Code yang dihasilkan pada figure 3.8 berfungsi untuk membaca dataset yang memiliki format text file dengan mendefinisikan variabel yang bernama imgatt. Variabel tersebut berisi value untuk membaca data.

The screenshot shows a Jupyter Notebook interface. At the top, there's a 'Variable explorer' tab, followed by 'File explorer' and 'Help'. Below that is the 'IPython console' tab, which is active. The console window displays the following Python session:

```

IPython 6.4.0 -- An enhanced Interactive Python.

In [1]: import pandas as pd
...:
...: # some Lines have too many fields (?), so skip bad lines
...: imgatt = pd.read_csv("D:/Perkuliahan/Semester 6/Kecerdasan Buatan/BUKU AI/CUB_200_2011/
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'])
...:

```

Figure 3.8: Random Forest.

- Pada source code berikutnya akan menghasilkan output seperti pada figure 3.9, dimana akan mengembalikan baris teratas dari DataFrame variabel imgatt.
- Pada output berikutnya akan menampilkan jumlah kolom dan baris dari DataFrame imgatt seperti yang dihasilkan pada figure 3.10
- Pada output yang ditampilkan dapat dilihat pada figure 3.11 bahwa variabel imgatt2 telah menggunakan function yang bernama pivot guna mengubah kolom jadi baris dan sebaliknya dari DataFrame sebelumnya
- Pada output yang ditunjukkan pada figure 3.12, dimana imgatt2 head berfungsi untuk mengembalikan value teratas pada DataFrame imgatt2
- Pada output yang dikeluarkan pada figure 3.13 menandakan jumlah kolom dan baris pada DataFrame imgatt2
- Pada output yang ditampilkan pada figure 3.14 menunjukkan dalam melakukan pivot yang mana imgid menjadi sebuah index yang unik

- Pada output yang ditampilkan pada figure 3.15 akan melakukan load jawabannya yang berisi apakah burung tersebut termasuk spesies yang mana. Kolom tersebut yaitu imgid dan label
- Pada output yang diberikan pada figure 3.16 menunjukkan bahwa jumlah baris sebanyak 11788 dan kolom 1 yang dimana kolom tersebut adalah jenis spesies pada burung
- Pada output yang ditunjukkan pada figure 3.17 akan melakukan join antara imgatt2 dengan imglabels dikarenakan memiliki isi yang sama sehingga akan mendapatkan sebuah data ciri-ciri dan data jawaban sehingga bisa dikategorikan sebagai supervised learning.
- Pada output yang ditunjukkan pada figure 3.18 akan melakukan drop pada label yang ada didepan dan akan menggunakan label yang baru di joinkan.
- Pada output yang ditunjukkan pada figure 3.19 akan mengecek isi 5 data teratas pada df att
- Pada output yang ditunjukkan pada figure 3.20 akan mengecek isi data teratas dari df label
- Pada output yang ditunjukkan pada figure 3.21 akan membagi 8000 row pertama menjadi data training dan sisanya adalah data testing
- Pada output yang ditunjukkan pada figure 3.22 dilakukan pemanggilan class RandomForestClassifier. Dimana artinya menunjukkan banyak kolom pada setiap tree adalah 50
- Pada output yang ditunjukkan pada figure 3.23 menunjukkan hasil prediksi dari Random Forest
- Pada output yang ditunjukkan pada figure 3.24 akan menampilkan besaran akurasi dari prediksi pada Random Forest yang merupakan score perolehan klarifikasi

## 5. Menjalankan Program Confusion Matrix

- Pada output yang ditunjukkan pada figure 3.25 akan melakukan import confusion matrix pada library sklearn matrix
- Pada output yang ditunjukkan oleh figure 3.26 bahwa akan menampilkan isi cm dalam bentuk matrix yang berupa array

- Pada output yang ditunjukkan pada figure 3.27 akan melakukan plotting atau merencanakan confusion matrix dengan matplotlib
- Pada output yang ditunjukkan pada figure 3.28 akan melakukan set plot dengan nama datanta dan akan membaca file classes.txt
- Pada output yang ditunjukkan ada figure 3.29 merupakan hasil perubahan pada label

## 6. Klarifikasi SVM dan Decission

- Pada figure 3.30 menunjukkan klarifikasi dengan decission tree menggunakan dataset yang sama dan akan memunculkan akurasi prediksi
- Pada figure 3.31 menunjukkan klarifikasi dengan SVM menggunakan dataset yang sama dan akan memunculkan akurasi prediksi

## 7. Program Cross Validation

- Pada figure 3.32 menunjukkan hasil cross validation pada Random Forest akan tetapi masih terdapat warning dikarenakan data numpy yang dikeluarkan oleh laptop saya tidak cukup memorynya
- Pada figure 3.33 menunjukkan hasil cross validation pada Decission Tree akan tetapi masih terdapat warning dikarenakan hal yang sama terjadi pada cross validation Random Forest
- Pada figure 3.34 menunjukkan hasil cross validation pada SVM akan tetapi masih terdapat warning dikarenakan hal yang sama terjadi pada cross validation Random Forest

## 8. Program Komponen Informasi

- Pada figure 3.35 menunjukkan informasi-informasi tree yang dibuat, atribut yang dipakai serta informasi lainnya akan tetapi terdapat warning dikarenakan data numpy pada komputer saya tidak cukup memorynya
- Pada figure 3.36 menunjukkan hasil dari plotting komponen informasi sehingga dapat kita baca

### **3.4.3 Penangan Erorr/Yusniar Nur Syarif Sidiq/1164089**

1. Script erorr yang saya dapat ditunjukkan pada figure 3.37
2. Erorrnya adalah

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

dikarenakan ekstensi file tersebut tidak terbaca cara penanganannya bisa dilihat pada figure 3.38

3. Solusinya jadikan file python dan .txt nya menjadi satu folder saja sehingga file .txt tersebut dapat terbaca

## **3.5 Imron Sumadireja/1164076**

### **3.5.1 Teori**

1. Random Forest Beserta Ilustrasinya

Random Forest adalah salah satu algoritma yang digunakan pada klasifikasi data dalam jumlah yang besar. Klasifikasi random forest ini dilakukan melalui penggabungan decision tree dengan melakukan training pada sampel data yang dimiliki atau biasa disebut dengan supervised learning. Semakin banyak menggunakan decision tree maka akan mempengaruhi akurasi yang didapatkan menjadi lebih baik. Setiap decision tree memiliki atribut yang berbeda, serta decision tree tersebut spesifik terhadap atributnya yang merupakan bagian kecil dari keseluruhan atribut pada data set. Contoh sederhananya bisa dilihat pada gambar berikut 3.39

2. Membaca Dataset, Makna Setiap File Serta Field Masing-Masing File

Pertama download terlebih dahulu datasetnya kemudian buka menggunakan spyder untuk mengetahui isi dari dataset tersebut. Untuk menjalankan code tersebut tinggal blok bagian yang akan di jalankan. Dataset tersebut di dalamnya terdapat class dari field atau data. Sebagai contoh pada data burung terdapat field index dan angka, untuk index biasanya berupa angka, angka tersebut memiliki makna sebagai pengganti nama atau jenis burung. Sedangkan field berisi nilai 0 dan 1 maknanya untuk memberikan penilaian ya atau tidak pada

setiap suatu data namun pada kasus ini field di ganti dari ya atau tidak menjadi 0 dan 1 karena komputer kesulitan membaca ya atau tidak dan hanya bisa membaca dengan 0 dan 1 saja.

### 3. Cross Validation

Cross validation adalah metode statistik yang dapat digunakan untuk mengevaluasi kinerja model atau algoritma dengan data dipisahkan menjadi dua subset yaitu data testing dan data training. Selain itu cross validation digunakan untuk memperkirakan seberapa akurat sebuah model prediktif ketika dijalankan. Untuk melakukan proses cross validation ini dibutuhkan sebuah data. Cross validation mengambil data dari output yang telah di eksekusi oleh algoritma sebelumnya. Hasil tersebut akan dipisahkan menjadi dua subset berdasarkan ukuran dataset. Selanjutnya dataset tersebut akan di test secara bergantian hingga seluruh bagian terpenuhi.

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

#### (a) Arti Score 44% Pada Random Forest

Score tersebut merupakan hasil prediksi dari data yang telah dieksekusi sebelumnya dengan algoritma random forest, score tersebut menandakan bahwa akurasi yang didapatkan tidak terlalu baik karena data yang diujinya cukup banyak. Tetapi itu jauh lebih baik daripada menebak secara acak.

#### (b) Arti Score 27% Pada Decision Tree

Score tersebut merupakan hasil prediksi dari data yang dieksekusi sebelumnya dengan algoritma decision tree, selain itu pada decision tree menggunakan library sklearn sebagai acuan untuk melakukan prediksi. Untuk decision tree ini hasil yang didapatkan ialah 27%. Hasil tersebut lebih buruk dibandingkan dengan menggunakan algoritma random forest.

#### (c) Arti Score 29% Pada SVM

Score tersebut merupakan hasil prediksi dari data yang dieksekusi sebelumnya dengan algoritma Support Vector Machine, score tersebut lebih baik daripada hasil yang di prediksi oleh decision tree namun score yang dimiliki oleh SVM tidak lebih baik dari hasil random forest.

## 5. Cara Membaca Confusion Matriks Beserta Ilustrasinya

Cara untuk membaca confusion matriks yakni dengan cara memasukan parameter nilai yang tersedia pada datasets. Data tersebut akan menghasilkan 0.5, 0.2 dan lain seterusnya sampai mendekati angka 1 atau akurasi yang sempurna. Pada confusion matriks terdapat 4 istilah sebagai representasi hasil proses klasifikasi, seperti gambar berikut 3.40

## 6. Jelaskan Voting Pada Random Forest Beserta Ilustrasinya

Voting pada random forest berguna untuk mengambil nilai pada masing-masing tree yang akan digunakan untuk menentukan hasil final dengan akurasi yang lebih baik. Untuk ilustrasi sederhananya sebagai berikut 3.41

### 3.5.2 Praktik Program / Imron Sumadireja / 1164076

#### 1. Aplikasi Sederhana Menggunakan Pandas

```
import pandas as pd
ron = {'Nama' : ['Arya', 'Razan', 'Bagja', 'MZ'], 'Umur' : [19,22,21,23],
       'NPM' : [1145032,1145031,1145065,1145098]}
df = pd.DataFrame(ron)
print (df)
```

- Pada baris pertama menjelaskan bahwa code tersebut mengimport library pandas
- Pada baris kedua itu merupakan sekumpulan data yang hasilnya akan membentuk seperti ndarray
- Pada baris ketiga itu merupakan dataframe atau kerangka data yang berisi variable
- Pada baris keempat itu untuk melihat hasil dari code tersebut.

Untuk hasilnya bisa dilihat seperti gambar berikut 3.42

#### 2. Aplikasi Sederhana Menggunakan Numpy

```
import numpy as np
a = np.arange(24)
a.ndim
b = a.reshape(2,3,4)
print (b)
```

- Pada baris pertama untuk import library numpy
- Pada baris kedua untuk menampilkan angka sebanyak 24 dimulai dari 0
- Pada baris ketiga merupakan nomor dari array dimensi
- Pada baris keempat tersebut akan merubah tampilannya menjadi 2 bagian dengan 4 kolom dan 3 baris
- Pada baris kelima untuk melihat hasil dari code tersebut.

Untuk hasilnya bisa dilihat seperti gambar berikut 3.43

### 3. Aplikasi Sederhana Menggunakan Matplotlib

```
import pandas as pd
import matplotlib.pyplot as plot
data = pd.read_csv("F:/Imron/Kuliah/Semester 6/Artificial Intelligence/penjualan_kelompok_tujuh.csv")
data.plot()
data.show()
```

- Pada baris pertama untuk import library pandas
- Pada baris kedua untuk import library matplotlib dengan inisialiasasi plt
- Pada baris ketiga untuk membaca file .csv pada direktori tersebut
- Pada baris keempat untuk membaca file .csv dengan library matplotlib
- Pada baris kelima untuk menampilkan grafik dari hasil data pada .csv

Untuk hasilnya bisa dilihat seperti gambar berikut 3.44

### 4. Menjalankan Program Klasifikasi Random Forest

Berikut ini adalah keluaran dari percobaan Klasifikasi Random Forest

- Hasil pada gambar 3.45 tersebut menampilkan data dari file image attribute label. File tersebut berisi kategori, attribut pada setiap gambarnya dengan jumlah data sekitar 3 juta dan dibagi menjadi 3 kolom, pada code tersebut digunakan syntax error bad lines itu berfungsi untuk melewatkkan data yang mengandung bad lines agar tidak terjadi errpr pada saat pem-bacaan file.
- Hasil pada gambar 3.46 tersebut menampilkan 5 data teratas pada dataframe secara default.

- Hasil pada gambar 3.47 menampilkan jumlah dari baris dan kolom pada file image attribute label atau dataframe
- Hasil pada gambar 3.48 merubah kolom menjadi baris, dan baris menjadi kolom dengan menggunakan fungsi dari pivot pada file sebelumnya
- Hasil pada gambar 3.49 tersebut menampilkan 5 data teratas secara default pada dataframe imgatt2
- Hasil pada gambar 3.50 menampilkan jumlah kolom dan baris pada dataframe imgatt2
- Hasil pada gambar 3.51 mengganti imgid menjadi index yang artinya unik untuk setiap datanya
- Hasil pada gambar 3.52 menampilkan 5 data teratas yang berisi apakah burung itu termasuk pada spesies yang mana. Kolom imgid ialah jenis burungnya dan kolom label itu spesies burungnya.
- Hasil pada gambar 3.53 menampilkan 11788 baris dan 1 kolom, dimana kolom tersebut merupakan spesies burungnya
- Hasil pada gambar 3.54 melakukan join antara imgatt2 dengan imglabels yang sebelumnya memiliki 312 kolom kini menjadi 313 kolom. Penggabungan ini termasuk ke dalam supervised learning karena kategorinya sudah tersedia.
- Hasil pada gambar 3.55 akan menghilangkan kolom pertama pada dataframe sebelumnya dan di rubah dengan kolom yang baru di join pada step sebelumnya
- Hasil pada gambar 3.56 menampilkan 5 data teratas dari dataframe att
- Hasil pada gambar 3.57 menampilkan 5 data teratas dari dataframe label
- Hasil pada gambar 3.58 membagi data menjadi 4 bagian, 8000 row pertama untuk data training atribut, 8000 row kedua untuk data training label, 8000 row ketiga untuk data testing atribut, dan 8000 row keempat untuk data testing label
- Hasil pada gambar 3.59 mengimport library sklearn ensemble untuk memanggil RandomForestClassifier, max features itu sebagai tanda ada beberapa kolom untuk setiap tree nya pada kali ini setiap tree memiliki 50 kolom dengan estimasi 100 tree.

- Hasil pada gambar 3.60 hasil dari fit untuk membuat random forest dengan kategori yang sudah ditentukan dengan maksimum fitur sebanyak 50 kolom untuk setiap tree nya dengan estimasi 100 tree
- Hasil pada gambar 3.61 menampilkan hasil prediksi pada step sebelumnya pada random forest
- Hasil pada gambar 3.62 menampilkan hasil presentasi akurasi dengan menggunakan algoritma random forest

## 5. Menjalankan Program Confusion Matrix

Berikut ini adalah keluaran dari hasil percobaan Confusion Matrix

- Hasil pada gambar 3.63 untuk memetakan data dari random forest ke dalam confusion matrix
- Hasil pada gambar 3.64 untuk menampilkan beberapa hasil dari data sebelumnya
- Hasil pada gambar 3.65 untuk merencanakan confusoin matrix dengan matplotlib sebelum di normalisasikan
- Hasil pada gambar 3.66 menampilkan file classes yang berisi nama-nama spesies burung
- Hasil pada gambar 3.67 merupakan dari proses normalisasi yang pada step sebelumnya sudah direncanakan

## 6. Menjalankan Klasifikasi SVM dan Decision Tree

Berikut ini adalah hasil dari percobaan yang telah dilakukan

- Hasil pada gambar 3.68 presentase prediksi yang dilakukan dengan menggunakan klasifikasi decision tree, dan hasilnya lebih buruk dibandingkan dengan random forest sebelumnya.
- Hasil pada gambar 3.69 presentase prediksi yang dilakukan dengan menggunakan klasifikasi SVM, dan hasilnya lebih baik di bandingkan dengan decision tree dan lebih buruk di bandingkan random forest

## 7. Menjalankan Program Cross Validation

Berikut ini adalah hasil keluaran dari percobaan yang telah dilakukan

- Hasil pada gambar 3.70 merupakan akurasi yang di dapatkan pada cross validation untuk random forest. Hasil tersebut masih warning dikarenakan mesinnya tidak kuat untuk melakukan prediksi secara menyeluruh
- Hasil pada gambar 3.71 merupakan akurasi yang didapatkan pada cross validation untuk decision tree. Hasil tersebut sama seperti step sebelumnya masih memiliki warning.
- Hasil pada gambar 3.72 merupakan akurasi yang didapatkan pada cross validation untuk SVM. Hasil tersebut masih sama seperti step sebelumnya masih memiliki tanda warning.

## 8. Menjalankan Program Pengamatan Komponen Informasi

Berikut ini adalah keluaran dari hasil percobaan yang telah saya lakukan

- Hasil pada gambar 3.73 seharusnya mengeluarkan beberapa informasi mengenai banyaknya tree dan atribut lainnya. Namun yang terjadi pada percobaan saya hanya mengeluarkan akurasi saja. Dikarenakan masih terdapat warning
- Hasil pada gambar 3.74 ini merupakan hasil dari plotting komponen informasi, namun dikarenakan pada step sebelumnya terdapat warning jadi data-data yang terdapat pada gambar tersebut, terlihat sedikit acak

### **3.5.3 Penanganan Error / Imron Sumadireja / 1164076**

Dari percobaan yang telah saya lakukan, saya menemukan beberapa error, diantaranya sebagai berikut

- Screenshot error 3.75
- Code error 3.76
- Solusi Pemecahan Masalah 3.77 saya coba rubah data training dan data testingnya menjadi 1000 dan hasilnya teratas dari pada yang sebelumnya. Error tersebut dikarenakan memorinya tidak muat untuk melakukan running dengan data yang begitu banyak. Bahkan laptop saya pun sudah coba di restart dan hasilnya tetap sama.

## **3.6 Andri Fajar Sunandhar/1164065**

### **3.6.1 Teori**

#### **1. Apa itu Random Forest Serta Gambar Ilustrasinya**

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

#### **2. Cara Membaca Dataset**

- (a) Buka Anaconda Navigator.
- (b) Jalankan Spyder
- (c) Import libraries yang dibutuhkan
- (d) Masukan kode berikut untuk membaca file Data.csv.
- (e) Jalankan kode tersebut, maka di windows console akan muncul pesan :
- (f) Klik variable explorer, maka akan terlihat dataset yang baru ter-import.
- (g) Kemudian double klik pada dataset cell, maka akan muncul pop-up windows seperti berikut:
- (h) Seperti yang terlihat pada gambar tersebut dataset ini memiliki Kolom Country, Age, dan Salary sebagai independent variable-nya dan kolom Purchased sebagai dependent variable-nya.

#### **3. Cross Validation**

Cross validation adalah metode statistik yang digunakan untuk memperkirakan keterampilan model pembelajaran mesin. Ini biasanya digunakan dalam pembelajaran mesin yang diterapkan untuk membandingkan dan memilih model untuk masalah pemodelan prediktif yang diberikan karena mudah dipahami, mudah

diimplementasikan, dan menghasilkan estimasi keterampilan yang umumnya memiliki bias lebih rendah daripada metode lainnya.

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

(a) Arti Score 44%

Pada Random Forest, Score tersebut merupakan hasil dari akurasi.

(b) Arti Score 27%

Pada decision tree adalah presentasi hasil dari perhitungan dataset.

(c) Arti Score 29% Pada SVM

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

(a) Confusion Matrix Dan Ilustrasinya

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

Kemudian data testingnya adalah

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

a= 5

b= 1

c= 1

d= 3

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

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

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

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

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

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

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

Setelah pohon terbentuk, maka akan dilakukan voting pada setiap kelas dari data sampel. Kemudian, mengkombinasikan vote dari setiap kelas kemudian diambil vote yang paling banyak. Dengan menggunakan random forest pada klasifikasi data maka, akan menghasilkan vote yang paling baik. 3.85

#### 3.6.2 Praktek Program

##### (a) Aplikasi Sederhana Menggunakan Pandas

Penjelasan kodingan :

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

Sehingga menghasilkan :

#### 6. Aplikasi Sederhana Menggunakan Numpy

Penjelasan kodingan :

- (a) Memanggil library numpy
- (b) Membuat variable dengan value eye dengan size10
- (c) Menampilkan hasil value

Sehingga menghasilkan :

#### 7. Aplikasi Sederhana Menggunakan Matplotlib

Penjelasan kodingan :

- (a) Memanggil library matplotlib.pyplot
- (b) Membuat variable yang berisi 10,20,30,40,50,60,70

- (c) Membuat garis koordinat
- (d) Menampilkan hasil plt

Sehingga menghasilkan :

8. Program Klasifikasi Random Forest :

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

9. Code Random Forest 2 :

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

10. Code Random Forest 3 :

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

11. Code Random Forest 4 :

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

12. Code Random Forest 5 :

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

13. Code Random Forest 6 :

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

14. Code Random Forest 7 :

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

15. Code Random Forest 8 :

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

16. Code Random Forest 9 :

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

17. Code Random Forest 10 :

- 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.

18. Code Random Forest 11 :

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

19. Code Random Forest 12 :

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

20. Code Random Forest 13 :

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

21. Code Random Forest 14 :

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

22. Code Random Forest 15 :

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

23. Code Random Forest 16 :

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

24. Code Random Forest 17 :

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

25. Code Random Forest 18 :

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

26. Program Klasifikasi Confusion Matrix

- Setelah melakukan random forest kemudian dipetakan ke dalam confusion matrix.
- Lalu melihat hasilnya.
- Kemudian dilakukan perintah plot.
- Selanjutnya nama data akan di set agar plot sumbunya sesuai.
- Setelah label berubah, maka dilakukan perintah plot.

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

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

28. Code Decision Tree :

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

29. Program Cross Validation

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

30. Program Pengamatan Komponen Informasi

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

### **3.6.3 Penanganan Eror**

Penyelesaian Tugas Harian ( Penanganan Error )

1. Menyelesaikan dan Membahas Penanganan Error :

- Skrinsut Error
- Kode Error: file b'data/CUB\_200\_2011/attributes/image\_attributes\_labels.txt'
- Solusi Pemecahan Error : Hapus Direktori data pada kode pastikan satu folder.

Console 1/A 

```
...: #
...: # <image_id> <attribute>
...: #
...: # where <image_id>, <attribute>
...: # in images.txt, attribute
...: # respectively. <is_present>
...: # present). <time> de
```

In [2]: `imgatt.head()`

Out[2]:

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

In [3]:

Figure 3.9: Random Forest.

```
Console 1/A X
....: #
....: # where <image_id>, <ati
....: # in images.txt, attribu
....: # respectively. <is_p
....: # present). <time> deno
```

In [2]: `imgatt.head()`

Out[2]:

	<code>imgid</code>	<code>attid</code>	<code>present</code>
0	1	1	0
1	1	2	0
2	1	3	0
3	1	4	0
4	1	5	1

In [3]: `imgatt.shape`

Out[3]: `(3677856, 3)`

In [4]:

Figure 3.10: Random Forest.

The screenshot shows the Jupyter Notebook interface. At the top is the "Variable explorer" pane, which lists two DataFrame objects: "imgatt" and "imgatt2". "imgatt" has 3677856 rows and 3 columns, with column names imgid, attid, and present. "imgatt2" has 11788 rows and 312 columns, with column names from 1 to 312. Below the variable explorer is the "IPython console" pane, which contains the following code and output:

```

.... # in images.txt, attributes/attributes.txt, and attributes/certainties.txt
.... # respectively. <is_present> is 0 or 1 (1 denotes that the attribute is
.... # present). <time> denotes the time spent by the MTurker in seconds.

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

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

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

```

Figure 3.11: Random Forest.

The screenshot shows the Jupyter Notebook "IPython console" pane with the following code and output:

```

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

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

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

[5 rows x 312 columns]

In [6]:

```

Figure 3.12: Random Forest.

The screenshot shows a Jupyter Notebook interface with two tabs: 'Variable explorer' and 'File explorer' at the top. Below is the 'IPython console' tab with a sub-tab 'Console 1/A'. The code in cell [4] is `imgatt2 = imgatt.pivot(index='imgid', columns='attid', values='present')`. The output of cell [5] is the head of the DataFrame:

```

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

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

```

[5 rows x 312 columns]

In cell [6], the command `imgatt2.shape` is run, resulting in the output `(11788, 312)`.

Figure 3.13: Random Forest.

The screenshot shows the Jupyter Notebook interface with the 'Variable explorer' tab active. It displays three variables:

Name	Type	Size	Value
imgatt	DataFrame	(3677856, 3)	Column names: imgid, attid, present
imgatt2	DataFrame	(11788, 312)	Column names: 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 1 ...
imglabels	DataFrame	(11788, 1)	Column names: label

Below the Variable explorer is the 'IPython console' tab with a sub-tab 'Console 1/A'. The code in cell [9] is:

```

In [9]: imglabels = pd.read_csv("D:/Perkuliahinan/Semester 6/Kecerdasan Buatan/BUKU AI/CUB_200_2011/
CUB_200_2011/image_class_labels.txt",
...:                         sep=' ', header=None, names=['imgid', 'label'])
...:
...: imglabels = imglabels.set_index('imgid')

```

Figure 3.14: Random Forest.

The screenshot shows a Jupyter Notebook interface. At the top, there are tabs for "Variable explorer", "File explorer", and "Help". Below the tabs, the title bar says "IPython console". A sub-header "Console 1/A" is followed by a red close button. The main area contains Python code and its execution results.

```
...: # The ground truth class
...: # in the file image_class
...: #
...: # <image_id> <class_id>
...: #
...: # where <image_id> and <
...: # respectively.
```

In [10]: `imglabels.head()`

Out[10]:

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

Figure 3.15: Random Forest.

IPython console

Console 1/A X

```
...: # <image_id> <class_
...: #
...: # where <image_id> a
...: # respectively.
```

In [10]: imglabels.head()

Out[10]:

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

In [11]: imglabels.shape

Out[11]: (11788, 1)

The screenshot shows a Jupyter Notebook interface. The Variable explorer pane at the top displays four DataFrame objects: df, imgatt, imgatt2, and imglabels. The IPython console pane below shows the execution of code to join and sample the data.

Name	Type	Size	Value
df	DataFrame	(11788, 313)	Column names: 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 1 ...
imgatt	DataFrame	(3677856, 3)	Column names: imgid, attid, present
imgatt2	DataFrame	(11788, 312)	Column names: 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 1 ...
imglabels	DataFrame	(11788, 1)	Column names: label

```
In [12]: df = imgatt2.join(imglabels)
...: df = df.sample(frac=1)
```

Figure 3.17: Random Forest.

The screenshot shows a Jupyter Notebook interface. The Variable explorer pane at the top displays six DataFrame objects: df, df\_att, df\_label, imgatt, imgatt2, and imglabels. The IPython console pane below shows the execution of code to join and sample the data, followed by selecting specific columns.

Name	Type	Size	Value
df	DataFrame	(11788, 313)	Column names: 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 1 ...
df_att	DataFrame	(11788, 312)	Column names: 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 1 ...
df_label	DataFrame	(11788, 1)	Column names: label
imgatt	DataFrame	(3677856, 3)	Column names: imgid, attid, present
imgatt2	DataFrame	(11788, 312)	Column names: 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 1 ...
imglabels	DataFrame	(11788, 1)	Column names: label

```
In [12]: df = imgatt2.join(imglabels)
...: df = df.sample(frac=1)

In [13]: df_att = df.iloc[:, :312]
...: df_label = df.iloc[:, 312:]
```

Figure 3.18: Random Forest.

```
In [14]: df_att.head()
Out[14]:
   1    2    3    4    5    6    7    ...    306   307   308   309   310   311   312
      ...
7809    0    0    0    0    1    0    0    ...    0    0    0    0    0    0    1
9555    0    0    0    0    0    0    1    ...    0    0    0    0    0    1    0
11676   0    0    0    0    0    0    1    ...    0    0    0    0    0    0    1
2054    0    0    0    0    0    0    1    ...    0    0    0    0    0    0    1
10229   0    0    0    0    0    0    1    ...    0    0    0    0    0    1    0

[5 rows x 312 columns]
```

Figure 3.19: Random Forest.

```
In [15]: df_label.head()
Out[15]:
          label

7809      134
9555      163
11676     199
2054      37
10229     174
```

Figure 3.20: Random Forest.

The screenshot shows a Jupyter Notebook interface. The Variable explorer pane at the top lists several data frames and series:

Name	Type	Size	Value
df_att	DataFrame	(11788, 312)	Column names: 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 1 ...
df_label	DataFrame	(11788, 1)	Column names: label
df_test_att	DataFrame	(3788, 312)	Column names: 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 1 ...
df_test_label	Series	(3788,)	Series object of pandas.core.series module
df_train_att	DataFrame	(8000, 312)	Column names: 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 1 ...
df_train_label	Series	(8000,)	Series object of pandas.core.series module
imgatt	DataFrame	(3677856, 3)	Column names: imgid, attid, present

The IPython console pane below contains the following code:

```
In [16]: 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 3.21: Random Forest.

```
In [17]: from sklearn.ensemble import RandomForestClassifier
....: clf = RandomForestClassifier(max_features=50, random_state=0, n_estimators=100)
```

Figure 3.22: Random Forest.

```
In [19]: print(clf.predict(df_train_att.head()))
[134 163 199 37 174]
```

Figure 3.23: Random Forest.

```
In [20]: clf.score(df_test_att, df_test_label)
Out[20]: 0.44376979936642025
```

Figure 3.24: Random Forest.

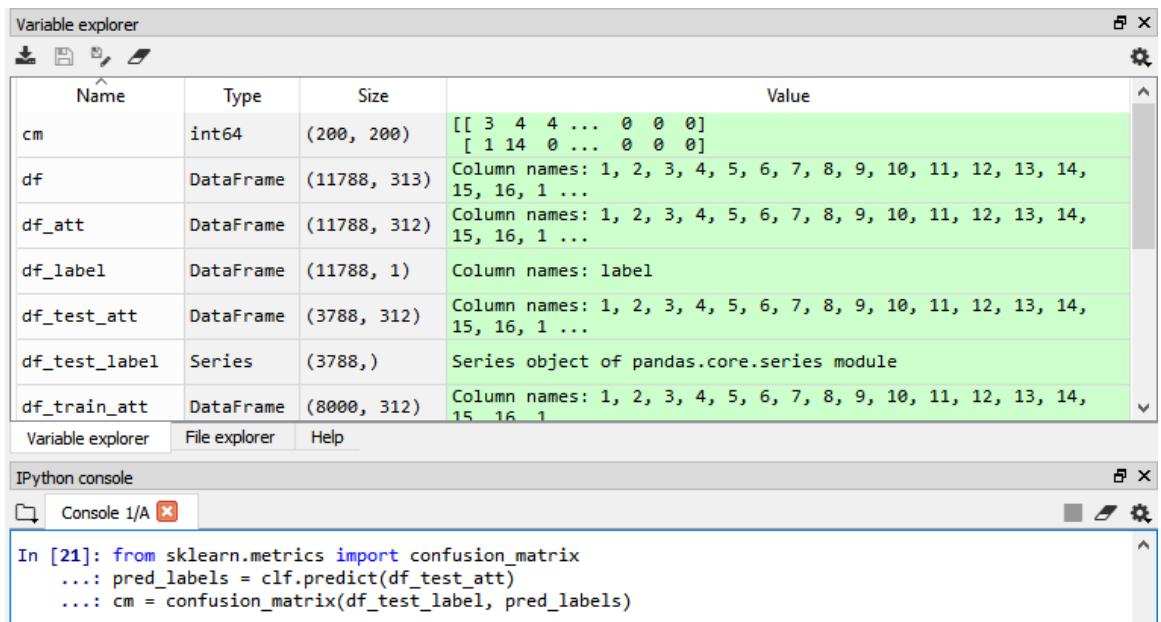


Figure 3.25: Confusion Matrix.

```
In [22]: cm
Out[22]:
array([[ 3,  4,  4, ...,  0,  0,  0],
       [ 1, 14,  0, ...,  0,  0,  0],
       [ 1,  0,  9, ...,  0,  0,  0],
       ...,
       [ 0,  0,  0, ...,  6,  0,  0],
       [ 0,  0,  0, ...,  1,  8,  0],
       [ 0,  0,  0, ...,  0,  0, 14]], dtype=int64)
```

Figure 3.26: Confusion Matrix.

```
In [23]: 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)
....:
```

Figure 3.27: Confusion Matrix.

```
In [25]: birds = pd.read_csv("D:/Perkuliahan/Semester 6/Kecerdasan Buatan/BUKU AI/CUB_200_2011/
CUB_200_2011/classes.txt",
    ...:                     sep='\s+', header=None, usecols=[1], names=['birdname'])
....: birds = birds['birdname']
....: birds
Out[25]:
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
```

Figure 3.28: Confusion Matrix.

```

In [26]: 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
[[0.12 0.15 0.15 ... 0. 0. 0.]
 [0.04 0.61 0. ... 0. 0. 0.]
 [0.07 0. 0.6 ... 0. 0. 0.]
 ...
 [0. 0. 0. ... 0.26 0. 0.]
 [0. 0. 0. ... 0.07 0.53 0.]
 [0. 0. 0. ... 0. 0. 0.61]]
Traceback (most recent call last):

```

Figure 3.29: Confusion Matrix.

```

In [27]: from sklearn import tree
....: clftree = tree.DecisionTreeClassifier()
....: clftree.fit(df_train_att, df_train_label)
....: clftree.score(df_test_att, df_test_label)
Out[27]: 0.2679514255543823

```

Figure 3.30: Klarifikasi SVM Dan Decission Tree.

```

In [28]: 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\NS\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[28]: 0.2824709609292503

```

Figure 3.31: Klarifikasi SVM Dan Decission Tree.

```

In [14]: from sklearn.model_selection import cross_val_score
....: scores = cross_val_score(clf, df_train_att, df_train_label, cv=5)
....: # show average score and +/- two standard deviations away (covering 95% of scores)
....: print("Accuracy: %0.2f (+/- %0.2f)" % (scores.mean(), scores.std() * 2))
C:\Users\NS\Anaconda3\lib\site-packages\sklearn\model_selection\_split.py:652: Warning: The least
populated class in y has only 1 members, which is too few. The minimum number of members in any class
cannot be less than n_splits=5.
  % (min_groups, self.n_splits)), Warning)
Accuracy: 0.28 (+/- 0.05)

In [15]:

```

Figure 3.32: Program Cross Validation.

```
In [18]: scorestree = cross_val_score(clftree, df_train_att, df_train_label, cv=5)
...: print("Accuracy: %0.2f (+/- %0.2f)" % (scorestree.mean(), scorestree.std() * 2))
C:\Users\NS\Anaconda3\lib\site-packages\sklearn\model_selection\_split.py:652: Warning: The least
populated class in y has only 1 members, which is too few. The minimum number of members in any class
cannot be less than n_splits=5.
% (min_groups, self.n_splits)), Warning)
Accuracy: 0.15 (+/- 0.03)
```

Figure 3.33: Program Cross Validation.

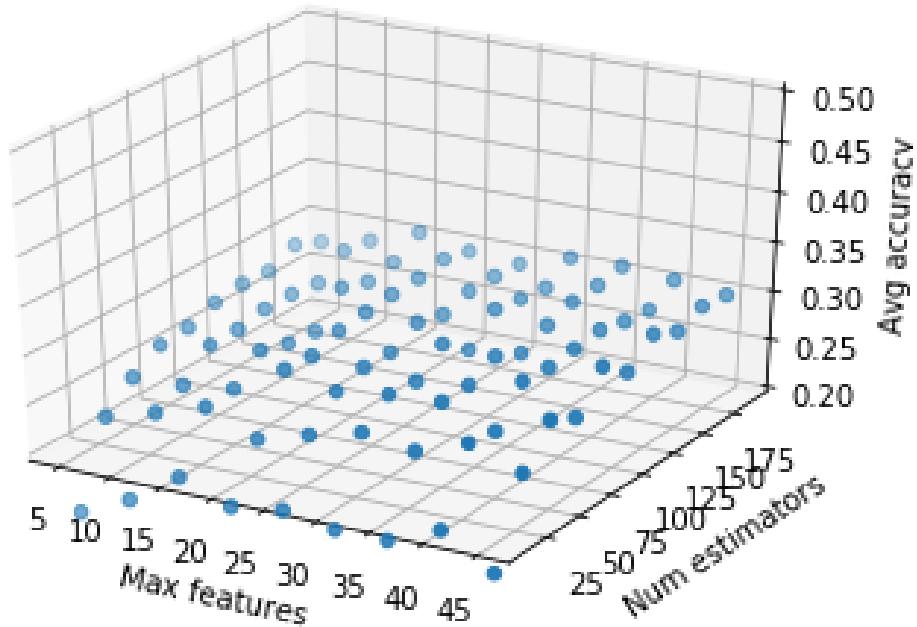
```
"avoid this warning.", FutureWarning)
C:\Users\NS\Anaconda3\lib\site-packages\sklearn\svm\base.py:196: FutureWarning: The default value of
gamma will change from 'auto' to 'scale' in version 0.22 to account better for unscaled features. Set
gamma explicitly to 'auto' or 'scale' to avoid this warning.
"avoid this warning.", FutureWarning)
Accuracy: 0.04 (+/- 0.03)
```

In [20]:

Figure 3.34: Program Cross Validation.

```
C:\Users\NS\Anaconda3\lib\site-packages\sklearn\model_selection\_split.py:652: Warning: The least
populated class in y has only 1 members, which is too few. The minimum number of members in any class
cannot be less than n_splits=5.
% (min_groups, self.n_splits)), Warning)
Max features: 45, num estimators: 190, accuracy: 0.30 (+/- 0.08)
```

Figure 3.35: Komponen Informasi.



In [27]:

---

Figure 3.36: Komponen Informasi.

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

Figure 3.37: Penanganan Erorr.

```
import pandas as pd

# some lines have too many fields (?), so skip bad lines
imgatt = pd.read_csv("data/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'])
```

Figure 3.38: Penanganan Erorr.

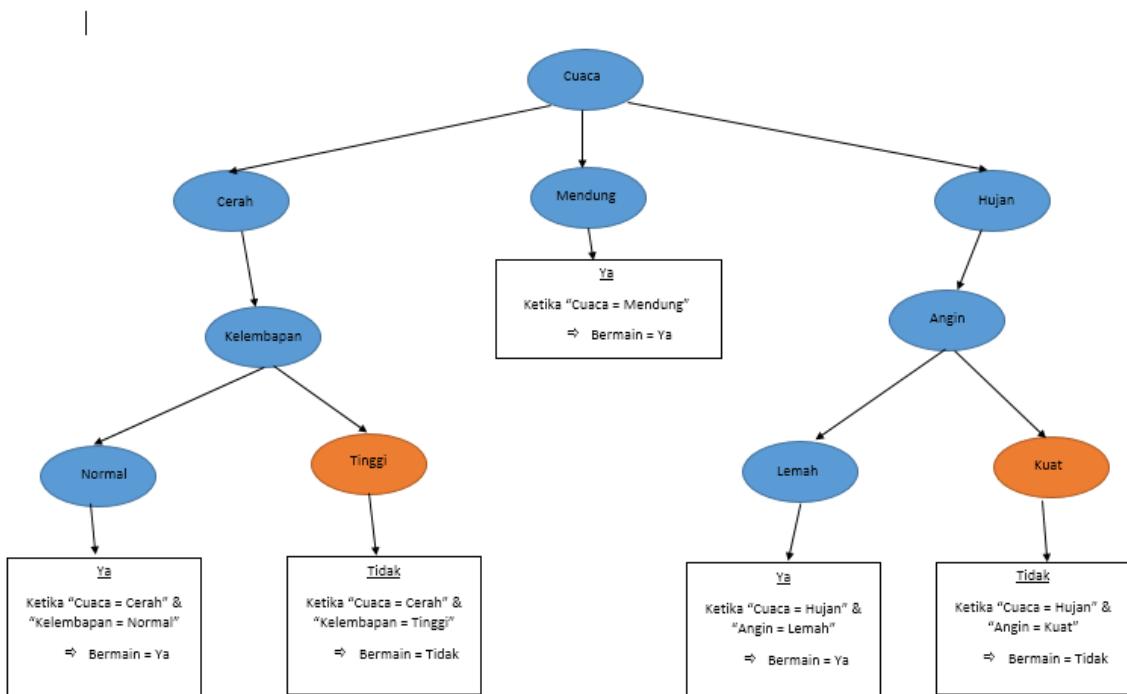


Figure 3.39: Random Forest.

		True Class	
		Positive	Negative
Predicted Class	Positive	true positives count (TP)	false negatives count (FP)
	Negative	false positives count (FN)	true negatives count (TN)

Figure 3.40: Confusion Matrix.

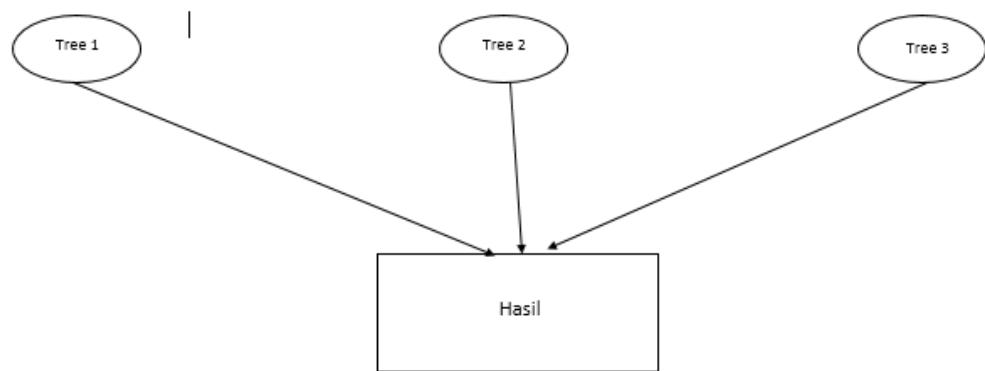


Figure 3.41: Voting.

```
In [25]: import pandas as pd
....:
....: ron = {'Nama' : ['Arya', 'Razan', 'Bagja', 'MZ'], 'Umur' : [19,22,21,23], 'NPM'
: [1145032,1145031,1145065,1145098]}
....: df = pd.DataFrame(ron)
....: print (df)
   Nama  Umur      NPM
0  Arya    19  1145032
1  Razan    22  1145031
2  Bagja    21  1145065
3     MZ    23  1145098
```

Figure 3.42: Hasil dari pandas.

```
In [20]: import numpy as np
....: a = np.arange(24)
....: a.ndim
....: b = a.reshape(2,3,4)
....: print (b)
[[[ 0  1  2  3]
 [ 4  5  6  7]
 [ 8  9 10 11]]
 [[12 13 14 15]
 [16 17 18 19]
 [20 21 22 23]]]
```

Figure 3.43: Hasil dari numpy.

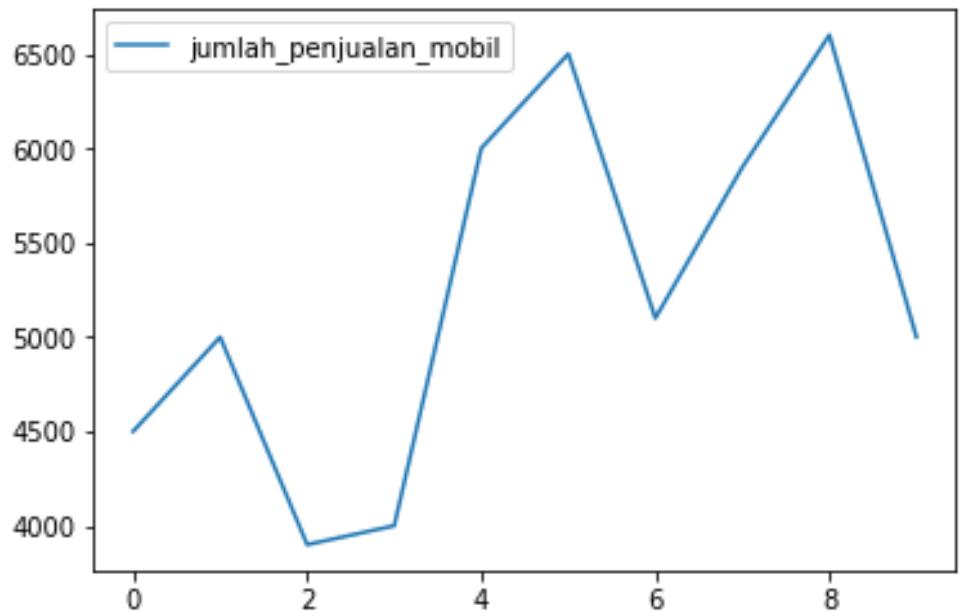


Figure 3.44: Hasil dari matplotlib.

imgatt - DataFrame

Index	imgid	attid	present
0	1	1	0
1	1	2	0
2	1	3	0
3	1	4	0
4	1	5	1
5	1	6	0
6	1	7	0
7	1	8	0
8	1	9	0
9	1	10	0
10	1	11	0
11	1	12	0
12	1	13	0
13	1	14	0

Format    Resize     Background color     Column min/max    OK    Cancel

Figure 3.45: Klasifikasi Random Forest1.

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

Figure 3.46: Klasifikasi Random Forest2.

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

Figure 3.47: Klasifikasi Random Forest3.

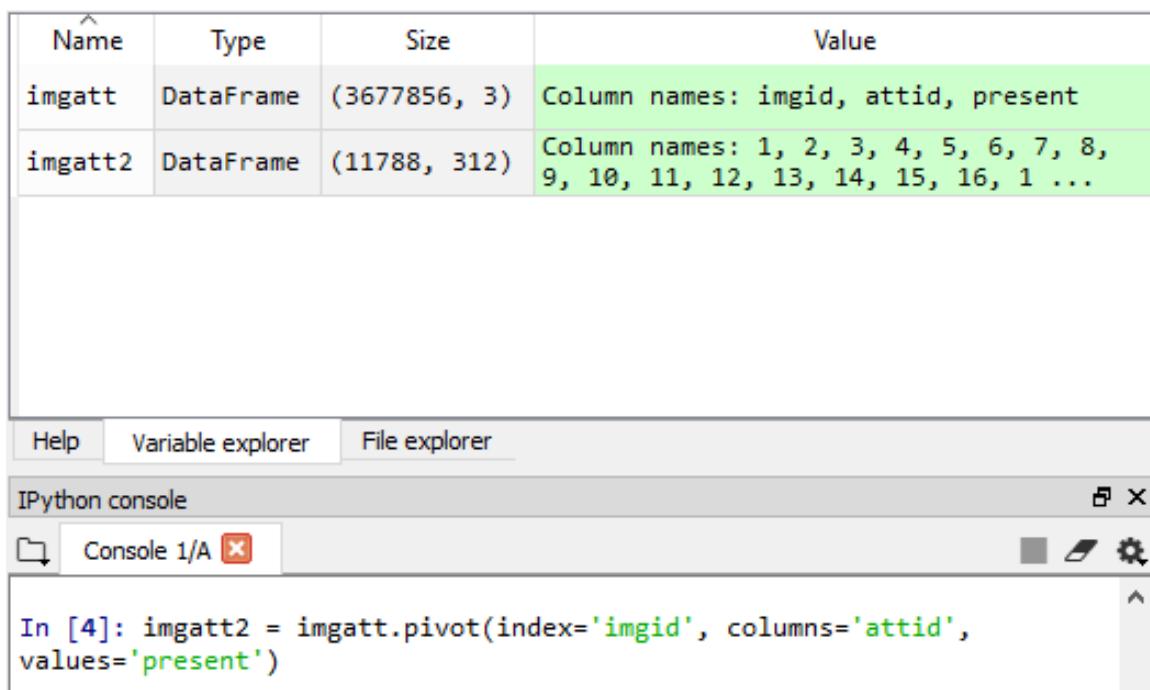


Figure 3.48: Klasifikasi Random Forest4.

```
In [6]: imgatt2.head()
Out[6]:
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
2      0    0    0    0    0    0    0    ...
0      0
3      0    0    0    0    1    0    0    ...
1      0
4      0    0    0    0    1    0    0    ...
0      0
5      0    0    0    0    1    0    0    ...
0      0
[5 rows x 312 columns]
```

Figure 3.49: Klasifikasi Random Forest5.

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

Figure 3.50: Klasifikasi Random Forest6.

Index	label
1	1
2	1
3	1
4	1
5	1
6	1
7	1
8	1
9	1
10	1
11	1
12	1
13	1
14	1

Figure 3.51: Klasifikasi Random Forest7.

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

Figure 3.52: Klasifikasi Random Forest8.

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

Figure 3.53: Klasifikasi Random Forest9.

df - DataFrame

Index	309	310	311	312	label
8279	0	0	0	141	
10604	0	1	0	181	
3145	0	0	1	55	
2373	0	0	0	42	
2586	0	1	0	45	
7719	0	1	0	132	
7790	0	1	0	133	
10938	0	0	1	186	
1389	0	0	0	25	
1360	0	0	0	25	
4774	0	1	0	82	
3612	0	0	0	63	
6824	0	0	0	117	
5710	0	0	1	98	

Background color  Column min/max

Figure 3.54: Klasifikasi Random Forest10.

df\_att - DataFrame

Index	1	2	3	4	5
8279	0	0	0	0	0
10604	0	0	0	0	0
3145	0	0	0	0	0
2373	0	0	0	0	0
2586	0	0	0	0	1
7719	0	0	0	0	0
7790	0	1	0	0	0
10938	0	0	0	0	0
1389	0	0	0	0	1
1360	0	0	0	0	1
4774	0	1	0	0	0
3612	0	0	0	0	1
6824	0	0	0	0	0
5710	0	1	0	0	0

Format Resize  Background color  Column min/max OK Cancel

Figure 3.55: Klasifikasi Random Forest11.

```

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

In [13]: df_att.head()
Out[13]:
   1    2    3    4    5    6    7    ...    306   307   308   309   310
311  312
imgid
8279    0    0    0    0    0    0    1    ...
0      0
10604   0    0    0    0    0    0    0    ...
1      0
3145    0    0    0    0    0    0    0    ...
0      1
2373    0    0    0    0    0    0    0    ...
0      0
2586   0    0    0    0    1    0    0    ...
1      0

[5 rows x 312 columns]

```

Figure 3.56: Klasifikasi Random Forest12.

```
In [14]: df_label.head()
Out[14]:
      label
imgid
8279      141
10604     181
3145       55
2373       42
2586       45
```

Figure 3.57: Klasifikasi Random Forest13.

Name	Type	Size	Value
df_test_att	DataFrame	(3788, 312)	Column names: 1, 2, 3, 4, 5,...
df_test_label	Series	(3788,)	Series object of pandas.core.series module
df_train_att	DataFrame	(8000, 312)	Column names: 1, 2, 3, 4, 5,...
df_train_label	Series	(8000,)	Series object of pandas.core.series module

Figure 3.58: Klasifikasi Random Forest14.

```
In [16]: from sklearn.ensemble import RandomForestClassifier
...: clf = RandomForestClassifier(max_features=50, random_state=0,
n_estimators=100)
```

Figure 3.59: Klasifikasi Random Forest15.

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

Figure 3.60: Klasifikasi Random Forest16.

```
In [18]: print(clf.predict(df_train_att.head()))
[141 181 55 42 45]
```

Figure 3.61: Klasifikasi Random Forest17.

```
In [19]: clf.score(df_test_att, df_test_label)
Out[19]: 0.44852164730728616
```

Figure 3.62: Klasifikasi Random Forest18.

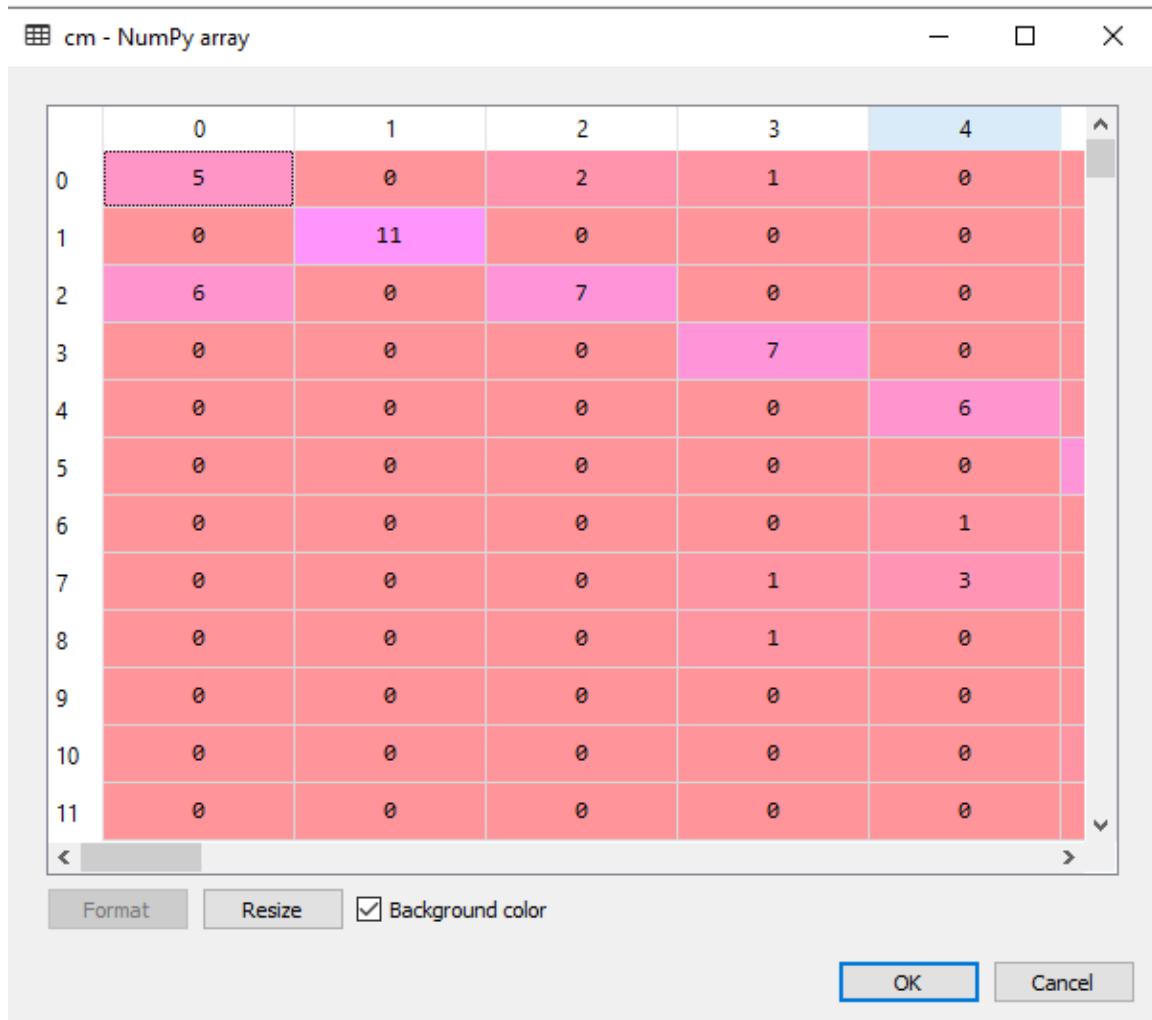


Figure 3.63: Confusion Matrix1.

```
In [21]: cm
Out[21]:
array([[ 5,  0,  2, ...,  0,  0,  0],
       [ 0, 11,  0, ...,  0,  0,  0],
       [ 6,  0,  7, ...,  0,  0,  0],
       ...,
       [ 1,  0,  0, ...,  5,  0,  0],
       [ 0,  0,  0, ...,  0,  8,  0],
       [ 0,  0,  0, ...,  0,  0, 14]], dtype=int64)
```

Figure 3.64: Confusion Matrix2.

```
In [22]: import matplotlib.pyplot as plt
...: import itertools
...: def plot_confusion_matrix(cm, classes,
...:                         normalize=False,
...:                         title='Confusion matrix',
...:                         cmap=plt.cm.Blues):
...:
...:     """
...:     This function prints and plots the confusion matrix.
...:     Normalization can be applied by setting `normalize=True`.
...:
...:     if normalize:
...:         cm = cm.astype('float') / cm.sum(axis=1)[:, np.newaxis]
...:         print("Normalized confusion matrix")
...:     else:
...:         print('Confusion matrix, without normalization')
...:
...:     print(cm)
...:
...:     plt.imshow(cm, interpolation='nearest', cmap=cmap)
```

Figure 3.65: Confusion Matrix3.

```
In [23]: birds = pd.read_csv("F:\Imron\Kuliah\Semester 6\Artificial
Intelligence\praktikum\Python-Artificial-Intelligence-Projects-for-
Beginners\Chapter02\CUB_200_2011\classes.txt",
...:                         sep='\s+', header=None, usecols=[1],
names=['birdname'])
...: birds = birds['birdname']
...: birds
Out[23]:
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
```

Figure 3.66: Confusion Matrix4.

```
In [25]: import numpy as np
....: np.set_printoptions(precision=2)
....: plt.figure(figsize=(30,30), dpi=300)
....: plot_confusion_matrix(cm, classes=birds, normalize=True)
....: plt.show()
Normalized confusion matrix
[[0.33 0. 0.13 ... 0. 0. 0. ]
 [0. 0.65 0. ... 0. 0. 0. ]
 [0.3 0. 0.35 ... 0. 0. 0. ]
 ...
 [0.04 0. 0. ... 0.22 0. 0. ]
 [0. 0. 0. ... 0. 0.4 0. ]
 [0. 0. 0. ... 0. 0. 1. ]]

```

Figure 3.67: Confusion Matrix5.

```
In [26]: from sklearn import tree
....: clftree = tree.DecisionTreeClassifier()
....: clftree.fit(df_train_att, df_train_label)
....: clftree.score(df_test_att, df_test_label)
Out[26]: 0.26610348468849
```

Figure 3.68: Decision Tree1.

```
In [27]: from sklearn import svm
....: clfsvm = svm.SVC()
....: clfsvm.fit(df_train_att, df_train_label)
....: clfsvm.score(df_test_att, df_test_label)
C:\ProgramData\Anaconda3\lib\site-packages\sklearn\svm\base.py:196:
FutureWarning: The default value of gamma will change from 'auto' to
'scale' in version 0.22 to account better for unscaled features. Set
gamma explicitly to 'auto' or 'scale' to avoid this warning.
    "avoid this warning.", FutureWarning)
Out[27]: 0.2932946145723337
```

Figure 3.69: SVM1.

```
In [26]: from sklearn.model_selection import cross_val_score
...: scores = cross_val_score(clf, df_train_att, df_train_label,
cv=5)
...: # show average score and +/- two standard deviations away
(covering 95% of scores)
...: print("Accuracy: %0.2f (+/- %0.2f)" % (scores.mean(),
scores.std() * 2))
C:\ProgramData\Anaconda3\lib\site-packages\sklearn\model_selection
\_split.py:652: Warning: The least populated class in y has only 1
members, which is too few. The minimum number of members in any class
cannot be less than n_splits=5.
% (min_groups, self.n_splits)), Warning)
Accuracy: 0.28 (+/- 0.04)
```

Figure 3.70: Cross Validation1.

```
In [27]: scorestree = cross_val_score(clftree, df_train_att,
df_train_label, cv=5)
...: print("Accuracy: %0.2f (+/- %0.2f)" % (scorestree.mean(),
scorestree.std() * 2))
C:\ProgramData\Anaconda3\lib\site-packages\sklearn\model_selection
\_split.py:652: Warning: The least populated class in y has only 1
members, which is too few. The minimum number of members in any class
cannot be less than n_splits=5.
% (min_groups, self.n_splits)), Warning)
Accuracy: 0.15 (+/- 0.03)
```

Figure 3.71: Cross Validation2.

```
'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: 0.07 (+/- 0.04)
```

Figure 3.72: Cross Validation3.

```
Max features: 45, num estimators: 130, accuracy: 0.28 (+/- 0.03)
C:\ProgramData\Anaconda3\lib\site-packages\sklearn\model_selection
\_split.py:652: Warning: The least populated class in y has only 1
members, which is too few. The minimum number of members in any class
cannot be less than n_splits=5.
    % (min_groups, self.n_splits)), Warning)
Max features: 45, num estimators: 150, accuracy: 0.28 (+/- 0.02)
C:\ProgramData\Anaconda3\lib\site-packages\sklearn\model_selection
\_split.py:652: Warning: The least populated class in y has only 1
members, which is too few. The minimum number of members in any class
cannot be less than n_splits=5.
    % (min_groups, self.n_splits)), Warning)
Max features: 45, num estimators: 170, accuracy: 0.27 (+/- 0.03)
C:\ProgramData\Anaconda3\lib\site-packages\sklearn\model_selection
\_split.py:652: Warning: The least populated class in y has only 1
members, which is too few. The minimum number of members in any class
cannot be less than n_splits=5.
    % (min_groups, self.n_splits)), Warning)
Max features: 45, num estimators: 190, accuracy: 0.28 (+/- 0.04)
```

Figure 3.73: Pengamatan Komponen Informasi1.

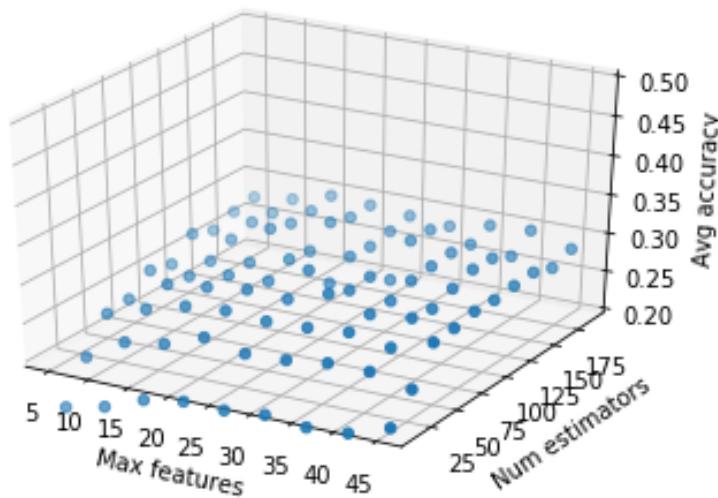


Figure 3.74: Pengamatan Komponen Informasi2.

```
\internals.py", line 4425, in reindex_indexer
    for blk in self.blocks]

  File "C:\ProgramData\Anaconda3\lib\site-packages\pandas\core
\internals.py", line 4425, in <listcomp>
    for blk in self.blocks]

  File "C:\ProgramData\Anaconda3\lib\site-packages\pandas\core
\internals.py", line 1258, in take_nd
    allow_fill=True, fill_value=fill_value)

  File "C:\ProgramData\Anaconda3\lib\site-packages\pandas\core
\algorithms.py", line 1652, in take_nd
    out = np.empty(out_shape, dtype=dtype)

MemoryError
```

Figure 3.75: Error1.

```
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))
```

Figure 3.76: Error2.

```

df_train_att = df_att[:1000]
df_train_label = df_label[:1000]
df_test_att = df_att[1000:]
df_test_label = df_label[1000:]

df_train_label = df_train_label['label']
df_test_label = df_test_label['label']

```

Figure 3.77: Solusi1.

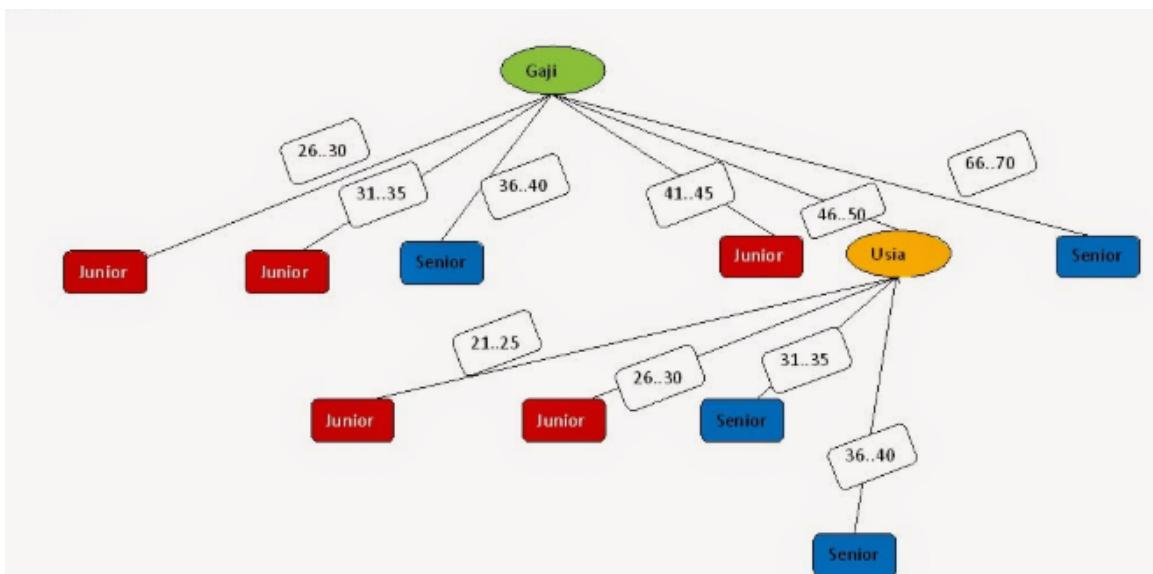


Figure 3.78: Random Forest.

```

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

```

Figure 3.79: Kode membaca file.csv

```

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

```

Figure 3.80: Window Console

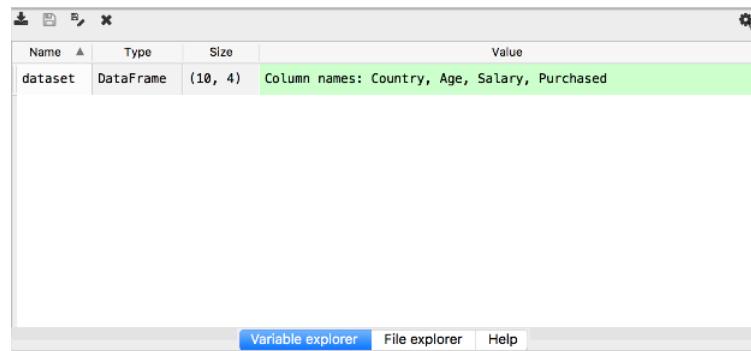


Figure 3.81: Variable Explorer

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

Figure 3.82: Dataset Cell

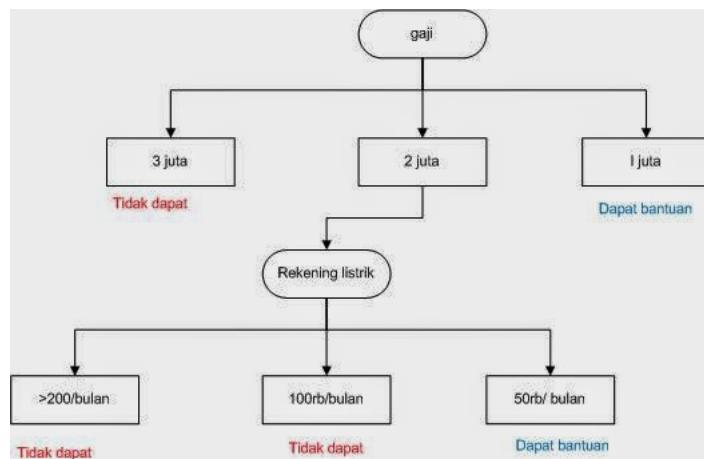


Figure 3.83: Pohon Keputusan

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

Figure 3.84: Data Testing

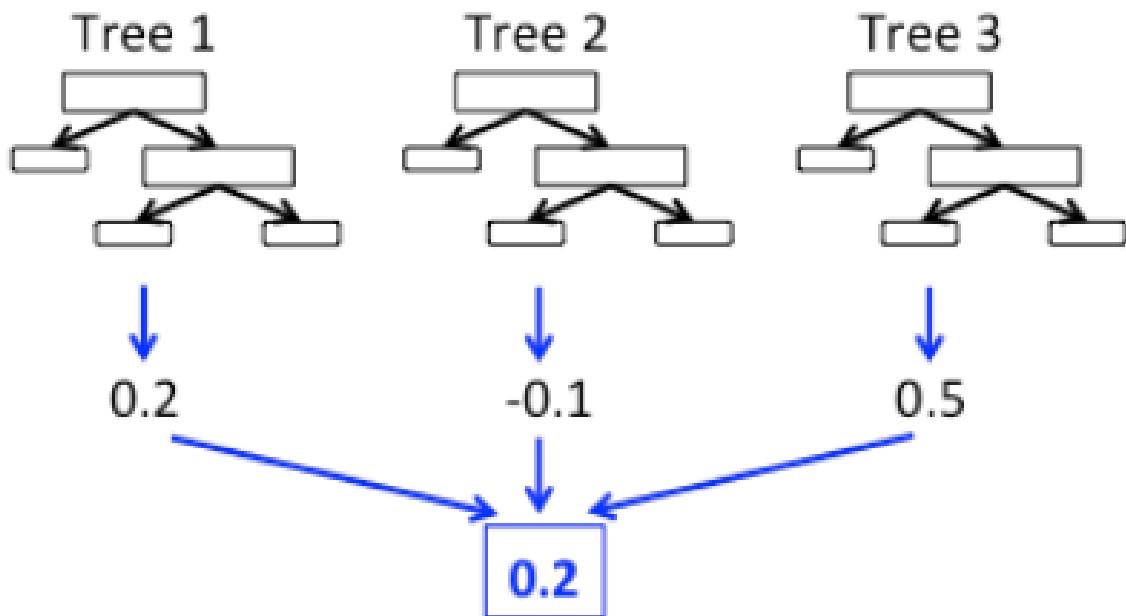


Figure 3.85: Voting.

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

Figure 3.86: Aplikasi Pandas

```
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 3.87: Hasil Pandas

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

Figure 3.88: Aplikasi Numpy

```

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

```

In [13]:

Figure 3.89: Hasil Numpy

```

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

```

Figure 3.90: Aplikasi Matplotlib

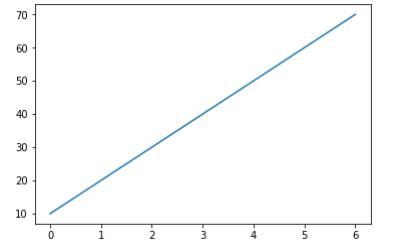


Figure 3.91: Hasil Matplotlib

```

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

```

Figure 3.92: 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 3.93: Gambar2

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

Figure 3.94: Gambar3

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

Figure 3.95: Gambar 4

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

Figure 3.96: Gambar 5

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

Figure 3.97: Gambar 6

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

Figure 3.98: Gambar 7

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

Figure 3.99: Gambar 8

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

Figure 3.100: Gambar 9

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

Figure 3.101: Gambar 10

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

Figure 3.102: Gambar 11

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

Figure 3.103: Gambar 12

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

Figure 3.104: 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 3.105: Gambar 14

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

Figure 3.106: 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 3.107: Gambar 16

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

Figure 3.108: Gambar 17

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

Figure 3.109: Gambar 18

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

Figure 3.110: Memetakan ke confusion matrix

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

Figure 3.111: Melihat hasil

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

Figure 3.112: 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 3.113: 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 3.114: Melakukan perintah plot

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

Figure 3.115: 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 3.116: Decission Tree

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

Figure 3.117: Pengecekan cross validation random forest

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

Figure 3.118: Pengecekan cross validation decision tree

```

In [42]: max_features_opts = range(5, 50, 5)
...: n_estimators_opts = range(10, 200, 20)
...: rf_params = np.empty((len(max_features_opts)*len(n_estimators_opts),4),
float)
...: i = 0
...: for max_features in max_features_opts:
...:     for n_estimators in n_estimators_opts:
...:         clf = RandomForestClassifier(max_features=max_features,
n_estimators=n_estimators)
...:         scores = cross_val_score(clf, df_train_att, df_train_label,
cv=5)
...:         rf_params[i,0] = max_features
...:         rf_params[i,1] = n_estimators
...:         rf_params[i,2] = scores.mean()
...:         rf_params[i,3] = scores.std() * 2
...:         i += 1
...:         print("Max features: %d, num estimators: %d, accuracy: %.2f
(+/- %.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 (+/- 0.01)
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 3.119: Pengamatan Komponen

```

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

```

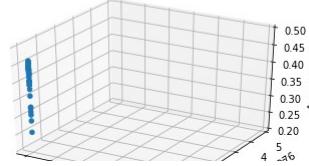


Figure 3.120: Plot informasi

```

parser_f
    return _read(filepath_or_buffer, kwds)

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

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

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

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

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

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

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

```

Figure 3.121: Error

# Chapter 4

## Experiment and Result

brief of experiment and result.

### 4.1 Experiment

Please tell how the experiment conducted from method.

### 4.2 Result

Please provide the result of experiment

### 4.3 Imron Sumadireja/1164076

#### 4.3.1 Teori

##### 1. Klasifikasi Teks dan Gambar Ilustrasi

Klasifikasi teks merupakan sebuah model yang digunakan untuk mengkategorikan teks ke dalam kelompok-kelompok yang lebih terorganisir. Jadi untuk setiap kalimat yang di masukan ke dalam mesin, mesin tersebut akan menjadikan setiap kata dari kalimat tersebut menjadi sebuah kolom. Untuk ilustrasinya bisa dilihat pada gambar berikut 4.1



Figure 4.1: Klasifikasi teks.

## 2. Mengapa klasifikasi bunga tidak bisa menggunakan machine learning

Karena machine learning tidak dapat menampilkan inputan sesuai dengan apa yang kita inputkan. Karena inputan tersebut serupa namun mesin memberikan output yang berbeda, biasanya output atau error ini disebut dengan istilah noise. Untuk contoh sederhananya misalkan kita inputkan salah satu label yang terdapat pada bunga, output yang dihasilkan oleh mesin tersebut ialah label yang lain. Itu dikarenakan bunga banyak jenis yang serupa namun tidak sama. Untuk ilustrasinya bisa dilihat pada gambar berikut 4.2

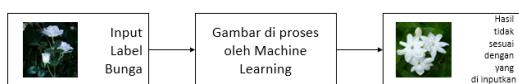


Figure 4.2: Klasifikasi Bunga.

## 3. Teknik pembelajaran mesin pada untuk kata-kata yang digunakan pada Youtube

Teknik yang digunakan pada youtube salah satunya ialah keywords. Dengan keywords tersebut mesin dapat memberikan video sesuai dengan keyword yang kita inputkan pada kolom pencarian. Teknik pembelajarannya tergantung user memberikan input teks seperti apa, karena pada youtube itu sendiri akan menyesuaikan dengan apa yang biasa kita inputkan dan akan memfilter video secara otomatis seuai dengan keyword yang biasa kita inputkan. Contoh ilustrasi sederhananya seperti berikut 4.3

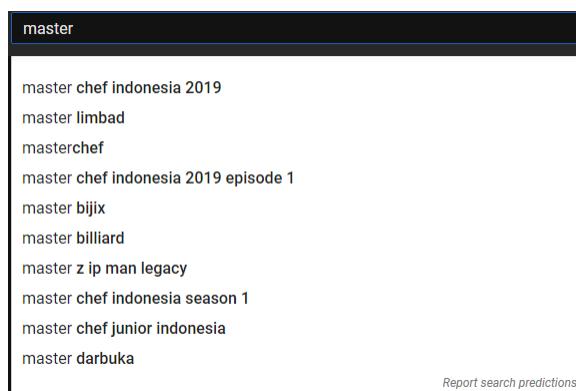


Figure 4.3: Klasifikasi teks Youtube.

## 4. Vektorisasi data

Vektorisasi data merupakan pemecahan atau pembagian data berupa teks, sebagai contoh terdapat 5 paragraf, data teks tersebut di pecah menjadi kalimat-kalimat yang lebih sederhana, lalu di pecah lagi menjadi kata untuk setiap kalimatnya.

#### 5. Bag of Words

Representasi penyederhanaan sebuah kalimat atau perhitungan setiap kata pada suatu kalimat dengan presentase berapa kali muncul kata tersebut untuk setiap kalimatnya. Contoh ilustrasi sederhananya seperti berikut 4.4

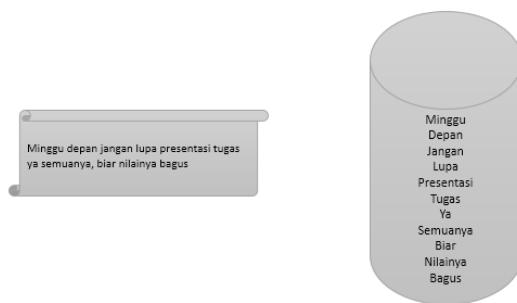


Figure 4.4: Bag of Words.

#### 6. Apa itu TF-IDF

TF-IDF merupakan metode untuk menghitung bobot setiap kata pada suatu kalimat yang paling sering digunakan. TF-IDF ini akan menghitung nilai Term Frequency dan Inverse Document Frequency pada setiap kata dalam setiap kalimat yang muncul dengan diimbangi dengan jumlah dokumen dalam korpus yang mengandung kata. Contoh ilustrasi sederhananya seperti gambar berikut 4.5

### 4.3.2 Praktikum / Imron Sumadireja / 1164076

1. Buat aplikasi sederhana menggunakan pandas dengan format csv sebanyak 500 baris

```
import pandas as pd
d = pd.read_csv("F:/Imron/..../praktikum/PraktikumChapter4/trial.csv")
```

- Baris pertama menjelaskan import library pandas dengan inisialisasi pd untuk mengelola dataframe

**1. Hapus Karakter Khusus**

Teks 1	Teks 2
Besok jangan lupa sholat jumat ya temen-temen semua biar kita dapat pahala disisi-Nya	Besok jangan lupa baca surah al-kahfi ya temen-temen biar rumah kita dalam lindungan-Nya di akhir kelak

**2. Tentukan bobot untuk setiap term dari dokumen-dokumen tersebut**

Term	TF		IDF	
	Teks 1 (Q)	Teks 2 (D1)	df	log(n/df)
Besok	1	1	2	0
jangan	1	1	2	0
lupa	1	1	2	0
sholat	1	0	1	0.30102999566398
jumat	1	0	1	0.30102999566398
ya	1	1	2	0
temen-temen	1	1	2	0
semua	1	0	1	0.30102999566398
biar	1	1	2	0

Figure 4.5: TF-IDF.

- Baris kedua untuk membaca file dengan format csv pada direktori tertentu
- Hasilnya seperti gambar berikut 4.6

d - DataFrame					
Index	Sector_score	LOCATION_ID	PARA_A	SCORE_A	PARZ
0	3.89	23	4.18	6	2.5
1	3.89	6	0	2	4.83
2	3.89	6	0.51	2	0.23
3	3.89	6	0	2	10.8
4	3.89	6	0	2	0.08
5	3.89	6	0	2	0.83
6	3.89	7	1.1	4	7.41
7	3.89	8	8.5	6	12.03
8	3.89	8	8.4	6	11.05
9	3.89	8	3.98	6	0.99
10	3.89	8	5.43	6	10.77
11	3.89	8	15.38	6	40.14
12	3.89	8	5.47	6	7.63
13	3.89	8	1.89	4	0.35

Figure 4.6: Data Frame.

2. Memecah dataframe tersebut menjadi dua bagian yaitu 450 row pertama dan 50 row kedua

```
d_train=d[:450]
d_test=d[450:]
```

- Baris pertama membagi data training menjadi 450
- Baris kedua membagi data menjadi 50 atau sisa dari data yang tersedia
- Hasilnya seperti gambar berikut 4.7

d_test	DataFrame	(51, 18)	Column names: Sector_score, LOCATION_ID, PARA_A, SCORE_A, PARA_B, SCOR ...
d_train	DataFrame	(450, 18)	Column names: Sector_score, LOCATION_ID, PARA_A, SCORE_A, PARA_B, SCOR ...

Figure 4.7: Data Frame.

3. Praktik vektorisasi dan klasifikasi dari data katty perry dan tunjukan keluarannya

```
import pandas as pd
d = pd.read_csv("F:/Imron/..../Chapter03/Youtube02-KatyPerry.csv")
```

- Baris pertama untuk import library pandas berguna untuk mengelola dataframe
- Baris kedua membaca file dengan format csv pada direktori tersebut
- Untuk hasilnya seperti gambar berikut 4.8

Index	COMMENT_ID	AUTHOR	DATE	CONTENT	CLA
0	z12pgdhovmrk...	lekanaVEO1	2014-07-22T1...	i love this so much. AND...	1
1	z13yx345uxep...	Pyunghee	2014-07-27T0...	http://www.billboar...	1
2	z12lsjvi3wa5...	Erica Ross	2014-07-27T0...	Hey guys! Please join ...	1
3	z13jcjuovxbw...	Aviel Haimov	2014-08-01T1...	http://psnboss.com/...	1
4	z13qybuau2yfy...	John Bello	2014-08-01T2...	Hey everyone. Watch this t...	1
5	z12rw1o4zvid...	Nere Overstylish	2014-08-02T2...	check out my rapping hope...	1
6	z13xizvwrki2...	Jayki L	2014-08-03T2...	Subscribe pleaaaase to...	1
7	z12ogvgbmre3...	djh3mi	2014-08-06T2...	hey guys!! visit my cha...	1
8	z125efjjoyax...	Manuel Ortiz	2014-08-07T1...	Nice! http://www.barnesan...	1
9	z121s34ysrzo...	Mike Bennett	2014-08-07T1...	http://www.twitch.t...	1
10	z13ijjrgqqyq...	Lil Misme	2014-08-09T0...	Hey Guys this is Glamour B...	1
11	z122xxgkipvd...	Emilie	2014-08-09T2...	Hey guys! My mom said if ...	1
12	z122hzpj5v3k...	Eduarda Ketryny	2014-08-11T0...	https://www.facebook...	1
13	z12pufnuvfn...	Tennika Chua	2014-08-11T1...	https://	1

Figure 4.8: Vektorisasi dan Klasifikasi.

```
spam=d.query('CLASS == 1')
nospam=d.query('CLASS == 0')
```

- Coding tersebut untuk membagi menjadi 2 tabel spam dan nospam, untuk hasilnya seperti berikut 4.9

```

nosspam DataFrame (175, 5) Column names: COMMENT_ID, AUTHOR, DATE, CONTENT,
          CLASS
spam DataFrame (175, 5) Column names: COMMENT_ID, AUTHOR, DATE, CONTENT,
          CLASS

```

Figure 4.9: Spam dan NoSpam.

```

from sklearn.feature_extraction.text import CountVectorizer
vectorizer = CountVectorizer ()

```

- Baris pertama untuk import countvectorizer berfungsi untuk memecah data tersebut menjadi sebuah kata yang lebih sederhana
- Baris kedua untuk menjalankan fungsi tersebut, pada code ini tidak ada hasilnya dikarenakan spyder tidak mendukung hasil dari instansi.

```

dvec = vectorizer.fit_transform(d['CONTENT'])
dvec

```

- Baris pertama untuk melakukan pemecahan data pada dataframe yang terdapat pada kolom konten
- Untuk menampilkan hasil dari code sebelumnya, hasilnya seperti gambar berikut 4.10

```

In [53]: dvec
Out[53]:
<350x1738 sparse matrix of type '<class 'numpy.int64'>'  

with 5275 stored elements in Compressed Sparse Row format>

```

Figure 4.10: Vektorisasi.

```
Daptarkata= vectorizer.get_feature_names()
```

- Code tersebut untuk menampilkan setiap kata pada dataframe yang sudah di vektorisasi, untuk hasilnya seperti gambar berikut 4.11

```
dshuf = d.sample(frac=1)
```

- Code tersebut untuk mengacak dataframe tersebut agar tidak berurutan lagi sesuai dengan waktu, untuk hasilnya seperti gambar berikut 4.12

```
d_train=dshuf [:300]
```

```
d_test=dshuf [300:]
```

Index	Type	Size	Value
0	str	1	00
1	str	1	000
2	str	1	002
3	str	1	018
4	str	1	04
5	str	1	053012
6	str	1	0cb8qfjaa
7	str	1	0d878a889c
8	str	1	0dbhjzdw0lbsjbi40gxm0d0p5krhv8xinql153__wqbahs8zx4mjhw5vwrkpxfoeks
9	str	1	0laviqu2b
10	str	1	10

Figure 4.11: Daftar Kata.

Index	COMMENT_ID	AUTHOR	DATE	CONTENT	CLA
221	z123ynxxyou1...	Brandon Tomlin	2014-10-17T1...	Like my page... please...	1
190	z130tpc5mmbq...	William Fernandez	2014-10-08T1...	Katy Perry - Roar (Official)	0
5	z12rw1o4zvid...	Nere Overstylish	2014-08-02T2...	check out my rapping hope...	1
31	z12jenlhyre0...	Special Pentruitive	2014-08-22T0...	&lt;script&gt;	1
285	z123tvlijsqaa...	Michael Jurek	2014-11-02T1...	I did a cover if u want to...	1
268	z13ez3wdxdsnj...	Chris Madzier	2014-10-29T0...	Katy Perry - Roar (Official)	0
52	z121gbuy2unh...	Santeri Saarikari	2014-09-03T1...	Hey guys go to check my ...	1
73	z12tvfb10snt...	blondedbythe...	2014-09-09T2...	She's an old Whore!	0
197	z13utrmmosbia...	Tiny Tim	2014-10-11T0...	What does that tattoo ...	0
2	z12lsjvi3wa5...	Erica Ross	2014-07-27T0...	Hey guys! Please join ...	1
290	z13ufbpg5sm...	OFFICIAL LEXIS	2014-11-04T2...	Hi everyone! Do you like ...	1
327	z13wsnj44xfi...	monal shah	2014-11-10T0...	https://apps.facebook...	1
189	z13iyv15vzql...	Tomas Adomatits	2014-10-08T1...	https://www.paidvert...	1
215	z124whinh1un...	Bi - Bases of	2014-10-15T1...	Visit my	1

Figure 4.12: Vektorisasi.

- Code tersebut untuk membagi data menjadi data training dan data testing, hasilnya seperti gambar berikut 4.13

```
d_train_att = vectorizer.fit_transform(d_train['CONTENT'])
d_train_att
```

- Baris pertama untuk memecah data pada tabel training di kolom content dari setiap kalimat menjadi kata
- Untuk menampilkan hasil dari code tersebut
- Hasilnya seperti gambar berikut 4.14

d_test	DataFrame	(50, 5)	Column names: COMMENT_ID, AUTHOR, DATE, CONTENT, CLASS
d_train	DataFrame	(300, 5)	Column names: COMMENT_ID, AUTHOR, DATE, CONTENT, CLASS

Figure 4.13: Pembagian data training dan testing.

```
In [57]: d_train_att = vectorizer.fit_transform(d_train['CONTENT'])
...: d_train_att
Out[57]:
<300x1564 sparse matrix of type '<class 'numpy.int64'>' with 4501 stored elements in Compressed Sparse Row format>
```

Figure 4.14: Pemecahan data pada table training.

```
d_test_att=vectorizer.transform(d_test['CONTENT'])
d_test_att
```

- Baris pertama untuk memecah data pada table testing di kolom content dari setiap kalimatnya menjadi kata
- Untuk menampilkan hasil dari code tersebut
- Hasilnya seperti gambar berikut 4.15

```
In [58]: d_test_att=vectorizer.transform(d_test['CONTENT'])
...: d_test_att
Out[58]:
<50x1564 sparse matrix of type '<class 'numpy.int64'>' with 596 stored elements in Compressed Sparse Row format>
```

Figure 4.15: Pemecahan data pada table testing.

```
d_train_label=d_train['CLASS']
d_test_label=d_test['CLASS']
```

- Baris pertam untuk menampilkan kolom class dari data training
- Baris kedua untuk menampilkan kolom class dari data testing
- Hasilnya seperti gambar berikut 4.16 dan 4.17

#### 4. Klasifikasi dari data vektorisasi menggunakan klasifikasi SVM

```
from sklearn import svm
clfsvm = svm.SVR(gamma = 'auto')
clfsvm.fit(d_train_att, d_train_label)
clfsvm.score(d_test_att, d_test_label)
```

- Baris pertama untuk import method svm dari library sklearn

Index	CLASS
221	1
190	0
5	1
31	1
285	1
268	0
52	1
73	0
197	0
2	1
290	1
327	1
189	1
215	1

  
    
  Background color

Figure 4.16: Menampilkan kolom class.

- Nilai gamma ini berfungsi untuk memperhitungkan perolehan presentase akhir agar lebih baik
- Untuk menyatukan data training atribut dengan label untuk di latih menggunakan metode svm
- Untuk memunculkan score dari hasil latihan tersebut, latihan ini sama seperti k-fold dengan menggunakan 2 data
- Hasilnya seperti gambar berikut 4.18

5. Klasifikasi dari data vektorisasi menggunakan klasifikasi Decision Tree

Index	CLASS
4	1
297	0
254	1
288	1
127	0
223	0
321	0
121	0
267	1
185	1
228	0
200	1
58	0
114	0

Background color

Figure 4.17: Menampilkan kolom class.

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

- Baris pertama untuk import metode tree pada library sklearn
- Baris kedua menjalankan decision tree classifier pada metode tree
- Untuk menyatukan data training label dan atribut untuk dilatih menggunakan metode decision tree
- Untuk memunculkan score dari hasil latihan dengan menggunakan metode decision tree

```

In [63]: from sklearn import svm
...: clfsvm = svm.SVR(gamma = 'auto')
...: clfsvm.fit(d_train_att, d_train_label)
...: clfsvm.score(d_test_att, d_test_label)
Out[63]: 0.3000293786871929

```

Figure 4.18: SVM.

- Hasilnya seperti berikut 4.19

```

In [67]: from sklearn import tree
...: clftree = tree.DecisionTreeClassifier()
...: clftree.fit(d_train_att, d_train_label)
...: clftree.score(d_test_att, d_test_label)
Out[67]: 0.9

```

Figure 4.19: Decision Tree.

6. Plotlah confusion matrix dari praktik modul ini menggunakan matplotlib

```

from sklearn.metrics import confusion_matrix
pred_labels=clf.predict(d_test_att)
cm=confusion_matrix(d_test_label,pred_labels)

```

- Baris pertama untuk import metode confusion matrix pada library sklearn.metrics
- Baris kedua menjelaskan bahwa data testing atribut akan di jalankan untuk di normalisasikan
- Baris ketiga akan menjalankan data testing label dan data testing atribut untuk di normalisasikan pada step selanjutnya, dan code tersebut tidak mengeluarkan hasil apa-apa dikarenakan code tersebut hanya mempersiapkan data

```

import matplotlib.pyplot as plt

def plot_confusion_matrix(cm, classes,
                         normalize=False,
                         title='Confusion matrix',
                         cmap=plt.cm.Blues):

    if normalize:
        cm = cm.astype('float') / cm.sum(axis=1)[:, np.newaxis]

```

```

        print("Normalized confusion matrix")
else:
    print('Confusion matrix, without normalization')

print(cm)

```

- Baris pertama untuk import library matplotlib dengan inisialisasi plt
- Fungsi dari plot confusion matrix ini menormalisasikan atau menyiapkan data untuk ditampilkan berupa grafik, untuk hasilnya seperti berikut 4.20 maksudnya data tersebut di prediksi ada 26 data yang termasuk bukan spam dan memang bukan spam namun ada 1 data yang di prediksi bukan spam tetapi hasilnya spam, begitupun sebaliknya terdapat 22 prediksi data spam dan itu memang termasuk dalam kategori spam tetapi ada 1 data yang di prediksi merupakan spam namun hasilnya bukan spam.

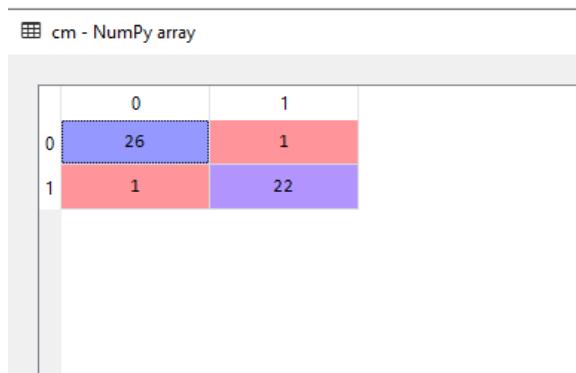


Figure 4.20: Confusion Matrix.

## 7. Menjalankan program cross validation

```

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

skor_rata2=scores.mean()
skoresd=scores.std()

```

- Baris pertama untuk import metode cross validation
- Melatih data training atribut dan label dengan menggunakan cross validation atau hampir sama dengan k-fold

- Untuk menampilkan hasil dari cross validation tersebut, seperti gambar berikut 4.21

skor_rata2	float64	1	0.9197591923682504
skoresd	float64	1	0.02768844128101342

Figure 4.21: Cross Validation.

```
from sklearn.model_selection import cross_val_score
scores = cross_val_score(clf, d_train_att, d_train_label, cv=5)
# show average score and +/- two standard deviations away (covering 95 \% of
print("Accuracy: \%.2f (+/-\%.2f)" \% (scores.mean(), scores.std() * 2))
```

- Code tersebut akan menentukan akurasi menggunakan cross validation dari hasil akurasi yang model decision tree, hasilnya seperti gambar berikut 4.22

```
In [80]: from sklearn.model_selection import cross_val_score
...: scores = cross_val_score(clf, d_train_att, d_train_label, cv=5)
...: # show average score and +/- two standard deviations away
...: (covering 95% of scores)
...: .... print("Accuracy: \%.2f (+/- \%.2f)" \% (scores.mean(), scores.std()
...: * 2))
Accuracy: 0.93 (+- 0.03)
```

Figure 4.22: Cross Validation.

## 8. Buatlah program pengamatan komponen informasi

```
max_features_opts = range(1, 10, 1)
n_estimators_opts = range(2, 40, 4)
rf_params = np.empty((len(max_features_opts)*len(n_estimators_opts),4), float)
i = 0
for max_features in max_features_opts:
    for n_estimators in n_estimators_opts:
        clf = RandomForestClassifier(max_features=max_features, n_estimators=n_estimators)
        scores = cross_val_score(clf, 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: \%, num estimators: \%, accuracy: \%.2f (+/- \%
(max_features, n_estimators, scores.mean(), scores.std() * 2))
```

- Code tersebut menjelaskan bahwa hasil dari data cross validation sebelumnya yang telah dilakukan pelatihan. Untuk range pada max features ini berfungsi untuk menampilkan parameter dari data sebelumnya dan untuk yang estimator berfungsi untuk mengatur akurasi pada gambar yang akan ditampilkan. Code tersebut tidak menampilkan keluaran

```

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(0.9, 1)
ax.set_xlabel('Max features')
ax.set_ylabel('Num estimators')
ax.set_zlabel('Avg accuracy')
plt.show()

```

- Code tersebut akan menampilkan grafik seperti berikut 4.23 grafik tersebut dihasilkan dari data-data yang telah dilatih sebelumnya dengan menggunakan algoritma decision tree, cross validation, svm. Untuk codingan diatas pada baris kedua itu untuk import axes 3D yang berfungsi untuk menampilkan grafik 3D lalu untuk baris ke 11 itu diartikan rata-rata akurasi yang didapatkan pada pelatihan sebelumnya yang sudah dilakukan.

## 4.4 Penanganan Error

### 4.4.1 Imron Sumadireja /1164076

Hasil praktikum yang telah saya lakukan terdapat beberapa kendala error diantaranya sebagai berikut:

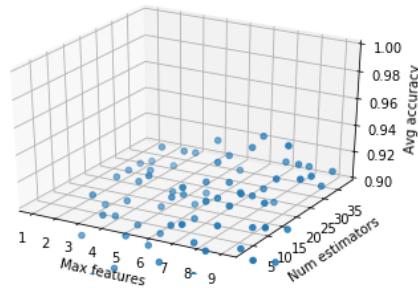


Figure 4.23: Komponen Informasi.

```
In [88]: from sklearn import svm
...: clfsvm = svm.SVC()
...: clfsvm.fit(d_train_att, d_train_label)
...: clfsvm.score(d_test_att, d_test_label)
C:\ProgramData\Anaconda3\lib\site-packages\sklearn\svm\base.py: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[88]: 0.46
```

Figure 4.24: Screenshot error.

1. Screenshot error 4.24
2. Error tersebut bermasalah dengan codingan yang seharusnya SVR, itu dikarenakan pada versi spyder yang saya gunakan SVC ini tidak dapat dirunning, walaupun di running itu akan menampilkan warning. Untuk menghilangkan warning maka solusi yang saya dapatkan seperti berikut 4.25

```
In [89]: from sklearn import svm
...: clfsvm = svm.SVR(gamma = 'auto')
...: clfsvm.fit(d_train_att, d_train_label)
...: clfsvm.score(d_test_att, d_test_label)
Out[89]: 0.3000293786871929
```

Figure 4.25: Solusi error.

## 4.5 Yusniar Nur Syarif Sidiq/1164089

### 4.5.1 Teori / Yusniar Nur Syarif Sidiq / 1164089

1. Jelaskan apa itu klasifikasi teks, sertakan gambar ilustrasi buatan sendiri. Klasifikasi teks merupakan sebuah proses pemberian tag atau kategori kedalam teks sesuai dengan isinya. Fungsi dari klasifikasi teks yaitu untuk melakukan klasifikasi atau pengelompokan teks ke dalam sebuah label tertentu. Untuk contoh klasifikasi teks dapat dilihat pada figure 4.26

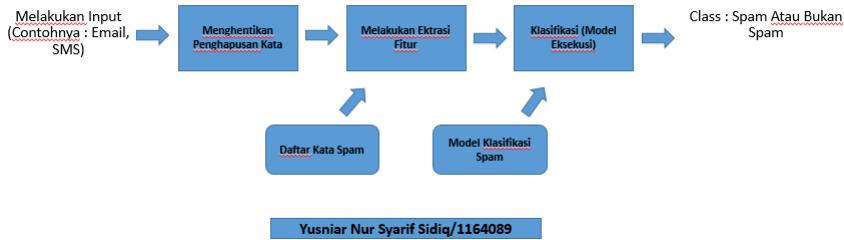


Figure 4.26: Contoh Klasifikasi Teks YN

Dimana dalam figure 4.26 menjelaskan terdapat daftar kata berupa kata-kata yang cukup banyak muncul pada email anda dengan tujuan menawarkan sesuatu barang atau hal lainnya, maka dapat dikategorikan bahwa kata tersebut merupakan spam.

2. Jelaskan mengapa klarifikasi bunga tidak bisa menggunakan machine learning, sertakan ilustrasi sendiri. Karena semua bunga belum tentu memiliki ciri-ciri yang sama, atau bisa dibilang adanya data noise dalam klasifikasi bunga yang dapat menyebabkan tidak bisa menggunakan Machine Learning. Akan saya beri contoh berdasarkan ilustrasi saya sendiri, terdapat bunga mawar bewarna merah yang memiliki jumlah 5 kelopak, lalu terdapat bunga selain mawar yang berwarna merah dan memiliki jumlah kelopak yang sama yaitu 5 serta memiliki kategori yang cukup banyak. Lalu terdapat bunga yang tidak cukup jelas datanya baik warnanya maupun jumlah kelopaknya, data tersebut akan menyababkan data noise. Untuk ilustrasi gambar akan saya berikan 2 buah gambar bunga dengan warna yang sama akan tetapi jenis yang berbeda, dapat dilihat pada figure 4.27



Figure 4.27: Contoh Klasifikasi Bunga YN

3. Jelaskan bagaimana teknik pembelajaran mesin pada teks pada kata-kata yang digunakan di youtube Dapat menggunakan teknik bag-of-words pada klasifikasi berbasis text dan kata guna mengklasifikasikan sebuah komentar yang

ada dalam internet sebagai kata spam atau bukan. Contohnya pada kolom komentar dapat di cek seberapa sering kata yang muncul dalam kalimat. Setiap kata bisa disebut sebagai baris dan kolomnya, hal ini merupakan dimana kategori kata spam atau tidak. Untuk ilustrasi gambarnya dapat dilihat pada figure 4.28

	COMMENT_ID	DATE	CONTENT	CLASS
345	z13th1q4yzlhfb1bl23qxjpjeyrdj	2014-11-14T13:27:52	How can this have 2 billion views when there's...	0
346	z13fenz1wfpb5e51xe4chdxakpzgchyexzo0k	2014-11-14T13:28:08	I don't know why I'm watching this in 2014	0
347	z130zd5b3tudko04ccbeophohjuxzppvb	2015-05-23T13:04:32	subscribe to me for call of duty vids and give...	1
348	z12he50arvkvivsu04cctawgxzkfjsjc4	2015-06-05T14:14:48	hi guys please my android photo editor downlo...	1
349	z13vhvu54u2ewpp5h4ccb4zuoardmijy0k	2015-06-05T18:05:16	The first billion viewed this because they tho...	0

Figure 4.28: Teknik Pembelajaran Mesi Pada Teks Youtube YN

4. Jelaskan apa yang dimaksud vektorisasi data. Vektorisasi data merupakan pembagian dan pemecahan data lalu data tersebut akan dilakukan perhitungan. Vektorisasi dapat kita maksudkan setiap data yang mungkin kita petakan ke integer tertentu. Misalnya saja kita memiliki data array yang cukup besar maka setiap kata cocok dengan slot unik dalam array. Contoh kita memiliki banyak kata yang tersesun dengan beberapa paragraf, data tersebut nantinya akan kita pecah menjadi beberapa kata dalam tiap kalimatnya.
5. Jelaskan apa itu bag of words dengan kata-kata yang sederhana dan ilustrasi sendiri 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. Ilustrasinya yaitu terdapat satu kalimat dan kalimat tersebut akan dipecah menjadi kata per kata, untuk lebih jelasnya dapat dilihat pada figure 4.29



Figure 4.29: Bag Of Words YN

6. Jelaskan apa itu TF-IDF, ilustrasikan dengan gambar sendiri TF-IDF dapat memberikan kita frekuensi kata dalam setiap dokumen sehingga dapat menggantikan data menjadi number. TD-IDF merupakan sebuah metode untuk dapat menghitung bobot setiap kata dalam kalimat yang sering digunakan. Untuk lebih jelasnya dapat dilihat dari figure ??

Kata	Doc 1	Doc 2	Doc 3
Yusniar	10	5	3
Suka	0	7	10
Mobil	12	8	6

**Yusniar Nur Syarif Sidiq/1164089**

Figure 4.30: TF-IDF YN

#### 4.5.2 Praktek Program / Yusniar Nur Syarif Sidiq / 1164089

1. Buat data dummy dengan format csv sebanyak 500 baris dan melakukan load ke dataframe pandas. Jelaskan arti setiap baris kode yang dibuat.

```
import pandas as pd
yn = pd.read_csv("AttributeDataSet.csv")
```

Dimana pada baris pertama akan melakukan import pandas yang di rename menjadi pd. Pada baris kedua kita akan membuat variabel yn lalu di isikan dengan file csv nya. Untuk hasil output dalam spyder bisa dilihat pada figure 4.31

Figure 4.31: Import DataFrame 500 Baris

2. Dari DataFrame tersebut dipecah menjadi dua DataFrame yaitu 450 row pertama dan 50 row sisanya.

```
d_train=yn[:450]
d_test=yn[450:]
```

Dimana pada baris pertama menjelaskan bahwa akan membagi data training sebanyak 450. Pada baris kedua akan membuat data testing sebanyak 50 yang merupakan sisanya. Untuk hasil output dalam spyder bisa dilihat pada figure 4.32

d_test	DataFrame	(50, 14)	Column names: Dress_ID, Style, Price, Rating, Size, Season, NeckLine, ...
d_train	DataFrame	(450, 14)	Column names: Dress_ID, Style, Price, Rating, Size, Season, NeckLine, ...

Figure 4.32: Membagi Data Menjadi 2 DataFrame

3. Praktekan vektorisasi dan klarifikasi dari data (NPM mod 4, jika 0 maka katty perry, 1 LMFAO, 2 Eminem, 3 Shakira). Tunjukan keluarannya dari komputer sendiri dan artikan maksud setiap keluaran yang dapatkan.

```
import pandas as pd
yn=pd.read_csv("Youtube03-LMFAO.csv")
```

Dimana hasil dari output tersebut akan membaca data csv yaitu Youtube03-LMFAO. Untuk hasil output dalam spyder bisa dilihat pada figure 4.33

yn	DataFrame	(438, 5)	Column names: COMMENT_ID, AUTHOR, DATE, CONTENT, CLASS
----	-----------	----------	--

Figure 4.33: Membaca File csv

Selanjutnya perhatikan source code dibawah ini,

```
spam=yn.query('CLASS == 1')
nospam=yn.query('CLASS == 0')
```

Dimana pada source code tersebut akan membagi data spam dan bukan spam, yang berarti data spam akan diberi tanda 1 dan data yang bukan spam akan diberi tanda 0. Untuk hasil output dalam spyder bisa dilihat pada figure 4.34

nospam	DataFrame	(202, 5)	Column names: COMMENT_ID, AUTHOR, DATE, CONTENT, CLASS
spam	DataFrame	(236, 5)	Column names: COMMENT_ID, AUTHOR, DATE, CONTENT, CLASS

Figure 4.34: Membagi Data Spam Dan Bukan Spam

Lanjut ke source code berikutnya,

```
from sklearn.feature_extraction.text import CountVectorizer
vectorizer = CountVectorizer()
```

Dimana source code diatas akan memanggil lib vektorisasi dan akan melakukan fungsi bag of word yaitu menghitung kata yang muncul perkaliat. Untuk hasil output dalam spyder bisa dilihat pada figure 4.35

Selanjutnya perhatikan source code berikut,

```
In [90]: from sklearn.feature_extraction.text import CountVectorizer  
...: vectorizer = CountVectorizer()
```

Figure 4.35: Fungsi Bag Of Word

```
dvec = vectorizer.fit_transform(yn['CONTENT'])  
dvec
```

Dimana pada source code tersebut akan melakukan vektorisasi pada colom content dan akan ditampilkan hasilnya. Untuk output dalam spyder dapat dilihat pada figure 4.36

```
Out[91]:  
<438x957 sparse matrix of type '<class 'numpy.int64'>'  
with 3724 stored elements in Compressed Sparse Row format>
```

Figure 4.36: Melakukan Vektorisasi Pada Colom Content

Lanjut ke source code berikutnya,

```
dk=vectorizer.get_feature_names()
```

Dimana pada source code tersebut akan menampilkan data yang sudah di vektorisasi. Untuk output dalam spyder dapat dilihat pada figure 4.37

Index	Type	Size	Value
0	str	1	00
1	str	1	000
2	str	1	10
3	str	1	100

Figure 4.37: Data Yang Sudah Di Vektorisasi

Selanjutnya perhatikan source code dibawah,

```
yeay = yn.sample(frac=1)
```

Dimana pada source code tersebut akan melakukan pengacakan data pada database agar sempurna saat akan melakukan klasifikasi. Untuk output dalam spyder dapat dilihat pada figure 4.38

Lanjut ke source code berikutnya,

```
yn_train=yeay[:300]  
yn_test=yeay[300:]
```

```
yeay    DataFrame (438, 5) Column names: COMMENT_ID, AUTHOR, DATE, CONTENT, CLASS
```

Figure 4.38: Pengacakan Data

Dimana pada source code tersebut akan membagi menjadi 2 DataFrame yaitu 300 pada data training dan sisanya data testing. Untuk output dalam spyder dapat dilihat pada figure 4.39

```
yn_test    DataFrame (138, 5) Column names: COMMENT_ID, AUTHOR, DATE, CONTENT, CLASS
yn_train    DataFrame (300, 5) Column names: COMMENT_ID, AUTHOR, DATE, CONTENT, CLASS
```

Figure 4.39: Membagi Data Yang Sudah Di Vektorisasi

Selanjutnya perhatikan source code dibawah,

```
yn_train_att=vectorizer.fit_transform(yn_train['CONTENT'])
yn_train_att
```

Dimana akan dilakukannya training pada data training dan akan di vektorisasi. Untuk output dalam spyder dapat dilihat pada figure 4.40

```
Out[95]: '- - -
<300x706 sparse matrix of type '<class 'numpy.int64'>
with 2510 stored elements in Compressed Sparse Row format>
```

Figure 4.40: Vektorisasi Pada Data Training

Selanjutnya perhatikan source code dibawah,

```
yn_test_att=vectorizer.transform(yn_test['CONTENT'])
yn_test_att
```

Dimana akan dilakukannya testing pada data testing dan akan di vektorisasi. Untuk output dalam spyder dapat dilihat pada figure 4.41

Lanjut ke source code berikut ini,

```
yn_train_label=yn_train['CLASS']
yn_test_label=yn_test['CLASS']
```

Dimana pada source code tersebut akan mengambil data spam dan bukan spam dari data training dan testing. Untuk output dalam spyder dapat dilihat pada figure 4.42

```
Out[96]:
<138x706 sparse matrix of type '<class 'numpy.int64'>'  
with 948 stored elements in Compressed Sparse Row format>
```

Figure 4.41: Vektorisasi Pada Data Testing

yn_test	DataFrame	(138, 5)	Column names: COMMENT_ID, AUTHOR, DATE, CONTENT, CLASS
yn_test_label	Series	(138,)	Series object of pandas.core.series module
yn_train	DataFrame	(300, 5)	Column names: COMMENT_ID, AUTHOR, DATE, CONTENT, CLASS
yn_train_label	Series	(300,)	Series object of pandas.core.series module

Figure 4.42: Mengambil Data Spam Dan Bukan Dari Traning Dan Testing Data

- Cobalah klasifikasikan dari data vektorisasi yang di tentukan di nomor sebelumnya dengan klasifikasi SVM.

```
from sklearn import svm  
clfsvm = svm.SVR(gamma='auto')  
clfsvm.fit(yn_train_att, yn_train_label)  
clfsvm.score(yn_test_att, yn_test_label)
```

Dimana pada source code tersebut akan melakukan import svm dari library sklearn dan akan melakukan klasifikasi dari data yang sudah di vektorisasikan. Hasil output dari source code tersebut berupa score prediksi dari svm. Untuk output dalam spyder dapat dilihat pada figure 4.43

```
In [104]: from sklearn import svm  
...: clfsvm = svm.SVR(gamma='auto')  
...: clfsvm.fit(yn_train_att, yn_train_label)  
...: clfsvm.score(yn_test_att, yn_test_label)  
Out[104]: 0.5314306819603158
```

Figure 4.43: Score Prediksi Dari SVM

- Cobalah klasifikasikan dari data vektorisasi yang di tentukan nomor sebelumnya dengan klasifikasi Decision Tree.

```
from sklearn import tree  
clftree = tree.DecisionTreeClassifier()  
clftree.fit(yn_train_att, yn_train_label)  
clftree.score(yn_test_att, yn_test_label)
```

Dimana source code tersebut akan melakukan import modul tree dari library sklearn dan akan melakukan klasifikasi dari data yang sudah di vektorisasikan. Hasil dari output source code tersebut akan berupa score dari prediksi Decision Tree. Untuk output dalam spyder dapat dilihat pada figure 4.44

```
In [105]: from sklearn import tree
...: clftree = tree.DecisionTreeClassifier()
...: clftree.fit(yn_train_att, yn_train_label)
...: clftree.score(yn_test_att, yn_test_label)
Out[105]: 0.9492753623188406
```

Figure 4.44: Score Prediksi Dari Decision Tree

6. Plotlah confusion matrix dari praktek modul ini menggunakan matplotlib.

```
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)
```

Dalam source code tersebut akan melakukan import library matplotlib dan di rename menjadi plt lalu memasukkan fungsi confusion matrix. Untuk output dalam spyder dapat dilihat pada figure 4.45

```
In [106]: 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)
```

Figure 4.45: Import Matplotlib

Selanjutnya perhatikan source code berikut ini,

```

import numpy as np
np.set_printoptions(precision=2)
plot_confusion_matrix(cm, classes=yn, normalize=True)
plt.show()

```

Dimana pada source code tersebut akan melakukan import pada library numpy yang dimana akan di rename menjadi np. Numpy disini berfungsi sebagai pengolah data matix arti dari numpy sendiri yaitu numarical matrix. Hasil dari source code tersebut akan memunculkan data matrix yang sudah di normalisasi. Untuk output dalam spyder dapat dilihat pada figure 4.46

```

In [107]: import numpy as np
...: np.set_printoptions(precision=2)
...: plot_confusion_matrix(cm, classes=yn, normalize=True)
...: plt.show()
Normalized confusion matrix
[[1.  0. ]
 [0.08 0.92]]

```

Figure 4.46: Matrix Normalisasi

## 7. Jalankan program cross validation.

```

from sklearn.model_selection import cross_val_score
scores = cross_val_score(clf, yn_train_att, yn_train_label, cv=5)
print("Accurac")

```

Dimana pada source code diatas akan menampilkan score akurasi pada Random Forest. Untuk output dalam spyder dapat dilihat pada figure 4.47

```

In [108]: from sklearn.model_selection import cross_val_score
...: scores = cross_val_score(clf, yn_train_att, yn_train_label, cv=5)
...: # show average score and +/- two standard deviations away (covering 95% of
...: scores)
...: print("Accuracy: %0.2f (+/- %0.2f)" % (scores.mean(), scores.std() * 2))
Accuracy: 0.94 (+/- 0.04)

```

Figure 4.47: Score Prediksi Akurasi RF

Selanjutnya perhatikan source code dibawah ini,

```

scorestree = cross_val_score(clftree, yn_train_att, yn_train_label, cv=5)
print("Accuracy")

```

Dimana pada source code diatas akan menampilkan score akurasi pada Decision Tree. Untuk output dalam spyder dapat dilihat pada figure 4.48

Selanjutnya perhatikan source code dibawah ini,

```
In [109]: scorestree = cross_val_score(clftree, yn_train_att, yn_train_label, cv=5)
...: print("Accuracy: %0.2f (+/- %0.2f)" % (scorestree.mean(), scorestree.std() * 2))
Accuracy: 0.98 (+/- 0.05)
```

Figure 4.48: Score Prediksi Akurasi DT

```
scoressvm = cross_val_score(clfsvm, yn_train_att, yn_train_label, cv=5)
print("Accuracy")
```

Dimana pada source code diatas akan menampilkan score akurasi pada SVM.

Untuk output dalam spyder dapat dilihat pada figure 4.49

```
In [110]: scoressvm = cross_val_score(clfsvm, yn_train_att, yn_train_label, cv=5)
...: print("Accuracy: %0.2f (+/- %0.2f)" % (scoressvm.mean(), scoressvm.std() * 2))
Accuracy: 0.51 (+/- 0.16)
```

Figure 4.49: Score Prediksi Akurasi SVM

8. Buatlah program pengamatan komponen informasi.

```
max_features_opts = range(1, 10, 1)
n_estimators_opts = range(2, 40, 4)
rf_params = np.empty((len(max_features_opts)*len(n_estimators_opts),4), float)
i = 0
for max_features in max_features_opts:
    for n_estimators in n_estimators_opts:
        clf = RandomForestClassifier(max_features=max_features, n_estimators=n_estimators)
        scores = cross_val_score(clf, yn_train_att, yn_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")
```

Output dari source code diatas akan melakukan pengulangan dari data-data sebelumnya yang sudah di vektorisasi. Untuk output dalam spyder dapat dilihat pada figure 4.50

Selanjutnya perhatikan source code berikut,

```

Max features: 1, num estimators: 2, accuracy: 0.82 (+/- 0.10)
Max features: 1, num estimators: 6, accuracy: 0.86 (+/- 0.07)
Max features: 1, num estimators: 10, accuracy: 0.90 (+/- 0.06)
Max features: 1, num estimators: 14, accuracy: 0.91 (+/- 0.11)
Max features: 1, num estimators: 18, accuracy: 0.92 (+/- 0.05)
Max features: 1, num estimators: 22, accuracy: 0.93 (+/- 0.04)
Max features: 1, num estimators: 26, accuracy: 0.94 (+/- 0.06)
Max features: 1, num estimators: 30, accuracy: 0.92 (+/- 0.04)
Max features: 1, num estimators: 34, accuracy: 0.92 (+/- 0.05)
Max features: 1, num estimators: 38, accuracy: 0.92 (+/- 0.04)
Max features: 2, num estimators: 2, accuracy: 0.83 (+/- 0.05)
Max features: 2, num estimators: 6, accuracy: 0.87 (+/- 0.09)
Max features: 2, num estimators: 10, accuracy: 0.90 (+/- 0.06)

```

Figure 4.50: Pengulangan Data Vektorisasi

```

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(0.8, 1)
ax.set_xlabel('Max features')
ax.set_ylabel('Num estimators')
ax.set_zlabel('Avg accuracy')
plt.show()

```

Hasil output dari source code tersebut adalah menampilkan data pengulangan yang sudah di vektorisasi dalam bentuk grafik. Untuk output dalam spyder dapat dilihat pada figure 4.51

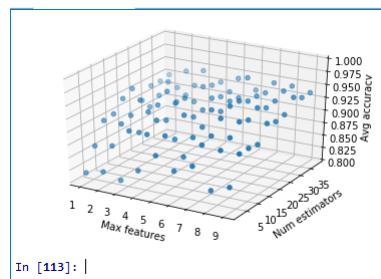


Figure 4.51: Grafik Data Pengulangan

#### 4.5.3 Penangan Error / Yusniar Nur Syarif Sidiq / 1164089

1. Screenshot Error Dimana error yang saya dapat ditunjukan pada figure 4.52

```
from sklearn import svm
clfsvm = svm.SVR()
clfsvm.fit(yn_train_att, yn_train_label)
clfsvm.score(yn_test_att, yn_test_label)
```

Figure 4.52: Syntax Error

2. Tuliskan kode error dan jenisnya

```
from sklearn import svm
clfsvm = svm.SVR()
clfsvm.fit(yn_train_att, yn_train_label)
clfsvm.score(yn_test_att, yn_test_label)
```

Dimana error tersebut merupakan syntax error yaitu terjadi kekurangan source code atau salah penulisannya.

3. Solusi pemecahan masalah error Solusi dari error tersebut adalah dengan lengkapi source codenya. Untuk source code lengkapnya dapat dilihat dibawah ini

```
from sklearn import svm
clfsvm = svm.SVR(gamma='auto')
clfsvm.fit(yn_train_att, yn_train_label)
clfsvm.score(yn_test_att, yn_test_label)
```

# **Chapter 5**

## **Conclusion**

brief of conclusion

### **5.1 Conclusion of Problems**

Tell about solving the problem

### **5.2 Conclusion of Method**

Tell about solving using method

### **5.3 Conclusion of Experiment**

Tell about solving in the experiment

### **5.4 Conclusion of Result**

tell about result for purpose of this research.

### **5.5 Yusniar Nur Syarif Sidiq/1164089**

#### **5.5.1 Pemahaman Teori / Yusniar Nur Syarif Sidiq / 1164089**

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

Dikarenakan mesin hanya mampu membaca data dengan bentuk angka maka dari itu diperlukan vektorisasi kata atau bisa disebut dengan mengubah

kata menjadi bentuk vektor agar mesin seolah-olah paham apa yang kita maksudkan. Kal ini saya memberikan ilustrasi sederhana, dimana ada sebuah data mengenai kucing, tikus, dan pulpen. Bagaimana cara mesin untuk membaca data tersebut ?, yaitu dengan cara dilakukannya vektorisasi kata. Fungsi dari vektorisasi itu sendiri ialah sebagai identitas, misalnya di dalam dokumen kucing terdapat kata-kata yang sudah di vektorisasi dan hasilnya adalah 0.012,0.024,.....,0.300, pada dokumen tikus menghasilkan 0.015,0.026,.....,0.0276, dan pada pulpen menghasilkan -0.191,...,-0.045. Data vektor tersebut merupakan identitas dari data kucing, tikus dan pulpen. Apabila nilai vektor cenderung sama, maka data tersebut memiliki similarity atau data yang memiliki konten kata yang sama akan tetapi berbeda dengan pulpen yang memiliki data vektorisasi minus, sehingga dapat disimpulkan pulpen memiliki konten kata yang berbeda. Dengan adanya vektorisasi tersebut maka mesin seolah-olah mengerti bahwa kucing dan tikus memiliki konten yang sama yaitu merupakan objek dari binatang sedangkan pulpen merupakan objek benda mati. Mengenai contoh gambarnya dapat dilihat pada figure 5.1

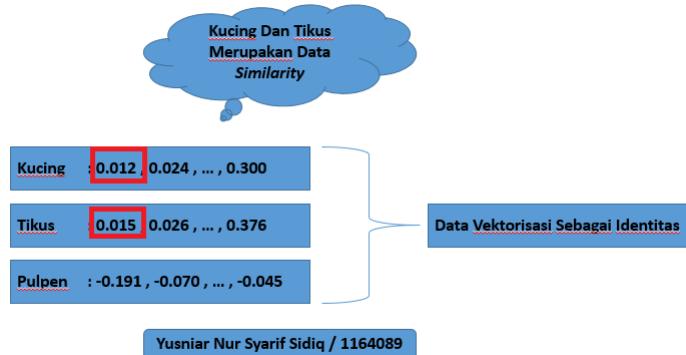


Figure 5.1: Kata-Kata Hasil Vektorisasi

2. Jelaskan mengapa dimensi dari vektor dataset google bisa sampe 300. Dilengkapi dengan ilustrasi atau gambar.

Dikarenakan pada setiap dataset didalamnya memiliki identitas masing-masing yang menyebabkan jumlah vektor dataset google bisa sampe 300. Saya akan memberikan sebuah ilustrasi yang saya dapat yaitu bagaimana pembagian dataset pada google. Didalam dataset google tersebut memiliki beberapa objek yaitu spatula, cat, dan dog. Dimana ketiga dataset tersebut akan dilakukan proses perbandingan dataset sehingga diproleh hasil antara cat dan dog yaitu

76% dikarekan pada dataset cat dan dog memiliki kesamaan data sedangkan untuk hasil perbandingan antara cat dan spatula yaitu 12%. Mengapa lebih sedikit, dikarenakan data yang dimiliki oleh cat dan spatula tidak memiliki kesamaan data. Hal ini dapat membuktikan setelah dilakukan vektorisasi mesin jadi dapat membedakan mana dataset yang memiliki kesamaan dan mana yang bukan. Mengenai pengertian dan ikustrasi tersebut dapat kita liat kedalam bentuk figure 5.2

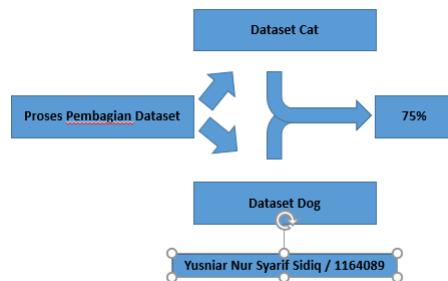


Figure 5.2: Dataset Google

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

Konsep pada vektorisasi kata yaitu dimana kata tersebut merupakan sebuah hasil pengolahan kata dari sebuah kalimat-kalimat yang telah kita olah. Misalnya saat kita membuka sosial media dan terdapat banyak komentar didalamnya. Ada sebuah kalimat yang mengatakan 'Follow instagram aku yah teman-teman'; dimana dalam kalimat tersebut memiliki kata kunci instagram dan kata tersebut akan dijadikan data training untuk mesin. Diamana nantinya mesin akan menampilkan kata-kata yang ada kaitannya dengan kata kunci tersebut. Figure 5.4 merupakan sebuah ilustrasi pada vektorisasi kata.



Figure 5.3: Vektorisasi Kata

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

Sebenarnya konsep pada vektorisasi dokumen dan vektorisasi kata itu sama saja, hal yang membedakannya itu hanya proses awalnya. Dimana pada vektorisasi kata akan membaca kalimat per kalimat namun pada vektorisasi dokumen akan membaca keseluruhan kalimat yang terdapat pada sebuah dokumen yang nantinya kalimat-kalimat tersebut akan dipecah menjadi kata per kata. Pada figure 5.4 merupakan contoh ilustrasi sederhana.

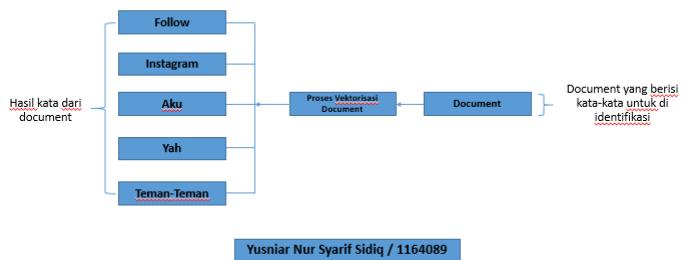


Figure 5.4: Vektorisasi Document

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

Mean merupakan nilai rata-rata dari suatu data. Mean dapat dicari dengan cara membagi jumlah data dengan banyak data sehingga diperolehlah nilai rata-rata dari suatu data. Sedangkan standar deviasi merupakan sebuah teknik statistik yang digunakan dalam menjelaskan homogenitas kelompok. Contoh sederhananya dimana kita menjumpai dataset yang memiliki 10 data yang berbeda dan 5 atribut yang berbeda. Untuk memperoleh nilai mean kita harus menghitung keseluruhan data tersebut lalu hasilnya akan dibagi dengan jumlah datanya. Setelah kita mendapatkan nilai mean maka kita bisa menghitung standar deviasi untuk memperoleh nilai statisknya.

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

Skip-gram merupakan sebuah teknik yang digunakan pada area speech processing yang dimana n-gram dibentuk lalu ditambah dengan tindakan skip. Misalkan ada sebuah kalimat yaitu *I hit the tennis ball ;* kita akan membuatnya menjadi skip gram 3 kata maka akan menjadi,

- I hit the

- Hit the tennis
- The tennis ball

Figure 5.5 merupakan ilustrasi sederhana.

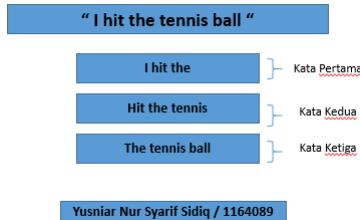


Figure 5.5: Kata-Kata Hasil Vektorisasi

### 5.5.2 Praktek Pemrograman / Yusniar Nur Syarif Sidiq / 1164089

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.

Dimana output dari source code tersebut merupakan data vektor untuk kata love. Perhatikan figure 5.6.

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

```
In [67]: gmodel['love']
Out[67]:
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,
```

Figure 5.6: Vektorisasi Kata

Output source code dibawah akan memunculkan data vektor untuk kata faith. Perhatikan figure 5.7

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

Output source code dibawah akan memunculkan data vektor untuk kata fall. Perhatikan figure 5.8

```
In [68]: gmodel['faith']
Out[68]:
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, -0.18839844,  0.10253906, -0.01367188,  0.23144531,
      -0.05957031, -0.22949219, -0.00604248,  0.26171875,  0.10302734,
```

Figure 5.7: Vektorisasi Kata

```
In [69]: gmodel['fall']
Out[69]:
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,
       -0.11085156,  0.01092529, -0.08300781, -0.0189209,  -0.1953125,
       -0.1015625,   0.13671675,  0.09228516, -0.12109375,  0.12695312,
       0.03417969,  0.2109375,   0.01977539,  0.125,      0.01544189,
```

Figure 5.8: Vektorisasi Kata

`gmodel['fall']`

Output source code dibawah akan memunculkan data vektor untuk kata sick.  
Perhatikan figure 5.9

`gmodel['sick']`

```
In [70]: gmodel['sick']
Out[70]:
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,
       2.24384199e-03,  9.66796875e-02, -1.66015625e-01, -1.12304688e-01,
       1.66015625e-01,  1.79687500e-01,  5.92841016e-03,  2.45117188e-01,
       8.74023438e-02, -2.56347656e-02,  3.41796875e-01,  4.98846875e-02,
```

Figure 5.9: Vektorisasi Kata

Output source code dibawah akan memunculkan data vektor untuk kata clear.  
Perhatikan figure 5.10

`gmodel['clear']`

```
In [71]: gmodel['clear']
Out[71]:
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, -4.76874219e-02, -6.83593750e-02, -1.21093750e-01,
       3.22265625e-02,  3.14453125e-01, -1.11816406e-01,  8.00781250e-02,
```

Figure 5.10: Vektorisasi Kata

Output source code dibawah akan memunculkan data vektor untuk kata shine.  
Perhatikan figure 5.11

`gmodel['shine']`

```
In [72]: gmodel['shine']
Out[72]:
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,
```

Figure 5.11: Vektorisasi Kata

Output source code dibawah akan memunculkan data vektor untuk kata bag.  
Perhatikan figure 5.12

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

```
In [73]: gmodel['bag']
Out[73]:
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,
       0.18347656,  -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,
```

Figure 5.12: Vektorisasi Kata

Output source code dibawah akan memunculkan data vektor untuk kata car.  
Perhatikan figure 5.13

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

```
In [74]: gmodel['car']
Out[74]:
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 ,  -0.10480391, -0.10742188, -0.01879883,
       0.05200195,  -0.00216675,  0.06445312,  0.14453125, -0.04541016,
      0.16113281,  -0.01611328, -0.03088379,  0.08447266,  0.16210938,
```

Figure 5.13: Vektorisasi Kata

Output source code dibawah akan memunculkan data vektor untuk kata wash.  
Perhatikan figure 5.14

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

```
In [75]: gmodel['wash']
Out[75]:
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,
```

Figure 5.14: Vektorisasi Kata

Output source code dibawah akan memunculkan data vektor untuk kata motor.  
Perhatikan figure 5.15

```
In [76]: gmodel['motor']
Out[76]:
array([ 5.73730469e-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,
      -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,  4.58984375e-01,
```

Figure 5.15: Vektorisasi Kata

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

Output source code dibawah akan memunculkan data vektor untuk kata cycle. Perhatikan figure 5.16

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

```
In [77]: gmodel['cycle']
Out[77]:
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,
```

Figure 5.16: Vektorisasi Kata

Pada source code dibawah menunjukkan hasil score perbandingan kata apakah kata cycle dan motor memiliki ke samaan atau tidak, perhatikan figure 5.17

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

```
In [78]: gmodel.similarity('cycle', 'motor')
Out[78]: 0.1794929675764453
```

Figure 5.17: Perbandingan Kata

Pada source code dibawah menunjukkan hasil score perbandingan kata apakah kata car dan wash memiliki ke samaan atau tidak, perhatikan figure 5.18

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

```
In [79]: gmodel.similarity('car', 'wash')
Out[79]: 0.20769942357569984
```

Figure 5.18: Perbandingan Kata

Pada source code dibawah menunjukkan hasil score perbandingan kata apakah kata bag dan shine memiliki ke samaan atau tidak, perhatikan figure 5.19

```
In [80]: gmodel.similarity('bag','shine')
Out[80]: 0.032655659470844235
```

Figure 5.19: Perbandingan Kata

```
gmodel.similarity('bag','shine')
```

Pada source code dibawah menunjukkan hasil score perbandingan kata apakah kata clear dan sick memiliki ke samaan atau tidak, perhatikan figure 5.20

```
gmodel.similarity('clear','sick')
```

```
In [81]: gmodel.similarity('clear','sick')
Out[81]: 0.15908542017205854
```

Figure 5.20: Perbandingan Kata

Pada source code dibawah menunjukkan hasil score perbandingan kata apakah kata fall dan faith memiliki ke samaan atau tidak, perhatikan figure 5.21

```
gmodel.similarity('fall','faith')
```

```
In [82]: gmodel.similarity('fall','faith')
Out[82]: 0.056926477919440686
```

Figure 5.21: Perbandingan Kata

2. Jelaskan dengan kata dan ilustrasi fungsi dari extract\_words dan PermuteSentences.

```
import re
def extract_words(sent):
    sent = sent.lower()
    sent = re.sub(r'<[^>]+>', ' ', sent)
    sent = re.sub(r'(\w)\'(\w)', ' ', sent)
    sent = re.sub(r'\W', ' ', sent)
    sent = re.sub(r'\s+', ' ', sent)
    return sent.split()
```

Dimana source code tersebut akan membersihkan tag-tag yang tidak perlu dan mengubahnya ke dalam bentuk array list perkata menggunakan 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

```

Dimana source code diatas berfungsi untuk melakukan pengacakan data supaya memperoleh data yang teratur.

3. Jelaskan fungsi library gensim dan Doc2Vec

```

from gensim.models.doc2vec import TaggedDocument
from gensim.models import Doc2Vec

```

Dimana fungsi dari library gensim sendiri yaitu untuk memodelkan bahasa dengan sistem unsupervised yang artinya tanpa label. Doc2Vec itu sendiri berfungsi untuk melakukan vectorisasi kata dalam satu document. Pada source code diatas dapat dijelaskan bahwa pada baris pertama akan melakukan import TaggedDocument yang artinya memasukkan document beserta tag-tagnya. Pada baris kedua yaitu akan melakukan import model Doc2Vec dari library gensim.

4. Jelaskan dengan kata dan praktik cara menambahkan data training dari file yang dimasukkan kepada variabel dalam rangka melatih model Doc2Vec.

```

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]))

```

Pada source code diatas akan dilakukan import data training dari folder Imdb. Dalam source code tersebut terdapat modul os yang dimana artinya program python yang ada pada pc kita dapat berinteraksi dengan sistem operasi yang pc kita gunakan. Setelah melakukan import pada modul os disini kita akan membuat sebuah variabel baru bernama unsup\_sentences. Selanjutnya kita akan mencari dimana folder Imdb disimpan. Selanjutnya kita akan melakukan open terhadap folder Imdb tersebut dengan menggunakan perintah with open. Selanjutnya kita akan membuat variabel baru yang bernama sent dan akan kita isikan perintah untuk membaca data folder Imdb. Selanjutnya kita akan membaut variabel baru lagi yang bernama word dan akan di isikan dengan perintah extract\_word yang dimana artinya kita akan melakukan extract kata yang ada pada variabel sent. Hasil output source code tersebut dapat dilihat pada figure 5.22.

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

Figure 5.22: Import Data Training.

```
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,
```

Pada source code diatas kita akan melakukan import data training kembali sehingga membuat data training kita bertambah. Jika source code sebelumnya kita telah menambahkan data training dari folder Imdb, kali ini kita akan menambahkan data training dari folder txt\_sentoken. Untuk setiap fungsi source code nya sama saja seperti sebelumnya. Untuk output dari source code tersebut dapat dilihat pada figure 5.23

```
with open("stanfordSentimentTreebank/original_rt_snippets.txt", encoding='UTF-8')
```

dirname	str	1	txt_sentoken/neg
fname	str	1	cv999_14636.txt
i	int	1	24
sent	str	1	after watching _a_night_at_the_roxbury_ , you'll be left with exactly ...
unsup_sentences	list	164720	[TaggedDocument, TaggedDocument, TaggedDocument, TaggedDocument, Tagge ...]
words	list	11	['after', 'watching', '_a_night_at_the_roxbury_', 'yo', 't', 'be', 'le ...']

Figure 5.23: Import Data Training.

```
for i, sent in enumerate(f):
    words = extract_words(sent)
    unsup_sentences.append(TaggedDocument(words, ["rt-%d" % i]))
```

Pada source code sebelumnya kita telah menambahkan data training dari folder Imdb dan txt\_stoken kali ini kita akan menambahkan data training kembali dengan menggunakan folder stanfordSentimentTreebank sehingga data training kita kembali bertambah. Untuk output dari source code tersebut dapat dilihat pada figure 5.24.

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

Figure 5.24: Import Data Training.

5. Jelaskan dengan kata dan praktik mengapa harus dilakukan pengocokan dan pembersihan data.

```
mute=PermuteSentences(unsup_sentences)
```

Pada source code diatas kita akan melakukan pengocokan atau pengacakan data. Sebelum dijadikan model kita perlu melakukan pengocokan agar mendapatkan hasil yang random dan bagus. Output dari source code diatas dapat dilihat pada figure 5.25.

```
In [16]: mute=PermuteSentences(unsup_sentences)
```

Figure 5.25: Proses Instansiasi Pengocokan Data.

Untuk pembersihan data perhatikan source code berikut,

```
model.delete_temporary_training_data(keep_inference=True)
```

Dikarenakan kita telah melakukan running data google yang jumlahnya mencapai 300000 data, maka perlu dilakukannya pembersihan data. Mengapa harus di bersihkan ?, agar memory pada pc kita memiliki space yang lega. Output dari source code tersebut dapat dilihat pada figure 5.28.

```
In [18]: model.delete_temporary_training_data(keep_inference=True)
```

Figure 5.26: Proses Instansiasi Pembersihan Data.

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

```
model.save('haci.d2v')
```

Pada source code diatas kita akan melakukan save pada data training yang sudah kita latih. Mengapa dilakukan save ?, agar kita dapat melakukan load saat ingin melatih datanya lagi. Data yang kita simpan akan terlihat seperti pada figure ??



Figure 5.27: Save Data Training.

Apabila kita ingin menghapus tempory data maka dapat menggunakan source code dibawah,

```
model.delete_temporary_training_data(keep_inference=True)
```

Tujuan dari penghapusan temporary data yaitu untuk membuat memory pada pc kita menjadi lega dikarenakan memory itu terbatas. Untuk ouput dari source code tersebut dapat dilihat pada figure 5.28.

7. Jelaskan dengan kata dan praktek maksud dari infer\_code.

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

```
In [18]: model.delete_temporary_training_data(keep_inference=True)
```

Figure 5.28: Proses Instansiasi Penghapusan Temporary Training Data.

Pada source code diatas kita akan melakukan infer\_code menggunakan model vector. Dimana fungsinya untuk menghitung atau mengkalkulasikan vektor dari kata yang dinputkan. Untuk output dari source code tersebut dapat dilihat pada figure 5.29

```
In [20]: model.infer_vector(extract_words("I will go home"))
Out[20]:
array([ 0.08174401,  0.1466673 , -0.24367264,  0.00348233, -0.16098163,
       0.31946293,  0.3119793 ,  0.09638206,  0.3246754 , -0.2236056 ,
       0.2582764 , -0.37287837, -0.00675154,  0.33590862,  0.28073108,
      -0.11735611, -0.06623545,  0.3199687 ,  0.21057786,  0.15321174,
      -0.1666727 ,  0.09892513, -0.23584224, -0.06347883, -0.02818101,
      0.2018524 , -0.34420416, -0.08902034,  0.01193086,  0.02601452,
     -0.11417291, -0.2962953 ,  0.13439985,  0.13803898,  0.09495226,
     -0.0528096 , -0.07157236,  0.29524505, -0.06434024,  0.16367984,
      0.1145281 ,  0.16330966, -0.08577985,  0.07457893,  0.18846703,
     -0.06975296, -0.24471088,  0.40807548, -0.03852524,  0.17686465,
      0.38876832, -0.0032483 ], dtype=float32)
```

Figure 5.29: Hasil Infer Vector.

8. Jelaskan dengan kata dan praktek maksud dari cosine\_similarity.

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

Dimana cosine\_similarity ini akan melakukan perbandingan vektorisasi terhadap dua kata, apabila hasil vektorisasi dari kedua kata tersebut adalah 50% hal ini dapat diperkirakan data kata tersebut memiliki kesamaan, dan apabila kurang dari 50% maka kata tersebut tidak memiliki kesamaan. Output dari source code diatas dapat dilihat pada figure 5.30.

```
In [21]: from sklearn.metrics.pairwise import cosine_similarity
...: cosine_similarity(
...:     [model.infer_vector(extract_words("she going to school, after
wash hand")),
...:      [model.infer_vector(extract_words("Services sucks."))]])
Out[21]: array([[0.49376744]], dtype=float32)
```

Figure 5.30: Consine Similarity.

9. Jalankan dengan praktek score dari cross validation masing-masing metode.

Untuk source code cross validation pada metode CLF dapat dilihat dibawah ini,

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

Dimana score yang diperoleh dari cross validation clf dapat dilihat pada figure 5.31

```
In [26]: scores = cross_val_score(clf, sentvecs, sentiments, cv=5)
...: np.mean(scores), np.std(scores)
Out[26]: (0.764, 0.022226110770892885)
```

Figure 5.31: Cross Validation CLF.

Untuk source code cross validation pada metode Randon Forest, dapat dilihat dibawah ini,

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

Dimana score yang diperoleh dari cross validation Random Forest dapat dilihat pada figure 5.32

```
C:\Users\NS\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[27]: (0.7153333333333334, 0.01575506973079531)
```

Figure 5.32: Cross Validation Random Forest.

Untuk source code cross validation pada metode CountVectorizer, dapat dilihat dibawah ini,

```
scores = cross_val_score(pipeline,sentences,sentiments, cv=5)
np.mean(scores), np.std(scores)
```

Dimana score yang diperoleh dari cross validation CountVectorizer dapat dilihat pada figure 5.33

```
C:\Users\NS\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[28]: (0.7373333333333334, 0.014007934259633788)
```

Figure 5.33: Cross Validation CountVectorizer.

```
File "<ipython-input-1-89dfdf2fc7606>", line 1, in <module>
    model.delete_temporary_training_data(keep_inference=True)

NameError: name 'model' is not defined
```

Figure 5.34: Name Error.

### 5.5.3 Penanganan Error / Yusniar Nur Syarif Sidiq / 1164089

1. Lakukan screenshot code yang error saat melakukan praktikum, tuliskan codenya dan nama errornya, lalu bagaimana cara menanganinya.

Untuk error yang saya dapat dapat dilihat pada figure 5.34.

Jenis error tersebut merupakan name error atau bisa juga disebut variabel error, dikarenakan variabel tersebut tidak terbaca,

```
model.delete_temporary_training_data(keep_inference=True)
```

Solusi dari error tersebut yaitu kita perlu membuat variabel model terlebih dahulu, maka kitakan source code pada figure 5.35 diatas source code errornya.

```
model = Doc2Vec(mute, dm=0, hs=1, vector_size=52)
```

Figure 5.35: Penangan Error.

## 5.6 Imron Sumadireja / 1164076

### 5.6.1 Teori

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

Karena untuk mengetahui presentase kata yang sering muncul dalam setiap kalimatnya, yang berguna untuk menetukan kata kunci, serta untuk memberikan kemudahan bagi mesin untuk mempelajari kata yang bentuk aslinya serupa namun tak sama seperti bebek dan ayam. Selain itu fungsi dari vektorisasi kata ini untuk membaca setiap kata-katanya dengan merubah kata menjadi sebuah angka atau identitas itu sendiri 5.36.

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



Figure 5.36: Ilustrasi Vektorisasi Kata.

Karena pada masing-masing objek itu memiliki identitas tersendiri, contoh sederhananya. Pada dataset google ini memiliki 3 buah objek diantaranya, cat, dog, dan spatula. Lalu dari masing-masing objek itu dibandingkan datasetnya antara cat dan dog lalu cat dan spatula. Hasil yang didapatkan untuk cat dan dog itu sekitar 76% sedangkan untuk cat dan spatula itu memiliki presentase 12% itu artinya bahwa mesin dapat membedakan objek yang hampir serupa namun tak sama. Untuk ilustrasi sederhananya bisa dilihat pada gambar 5.37.

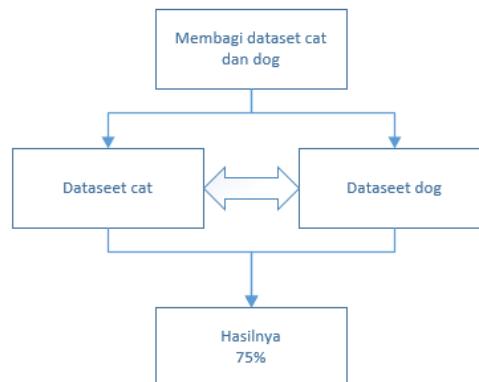


Figure 5.37: Ilustrasi Vektorisasi Dataset Google.

3. Jelaskan konsep vektorisasi untuk kata. Dilengkapi dengan ilustrasi

Konsep untuk vektorisasi kata ini sama halnya dengan kita input suatu kata pada mesin pencari. Lalu 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 subscribe my channel thank you guys ), pada kalimat tersebut terdapat konteks yakni channel, kata tersebut akan dijadikan data latih untuk mesin. Jadi ketika kita inputkan kata channel, maka mesin akan menampilkan keterkaitannya dengan kata tersebut. Ilustrasinya bisa dilihat pada gambar berikut 5.38.

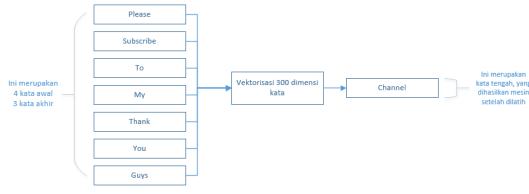


Figure 5.38: Ilustrasi Vektorisasi Kata.

4. Jelaskan konesep vektorisasai untuk dokumen. Dilengkapi dengan ilustrasi

Sama halnya dengan vektorisasi kata, yang membedakan hanya pada proses awalnya. Untuk vektorisasi dokumen ini, mesin akan membaca semua kalimat yang terdapat pada dokumen tersebut, lalu kalimat yang terdapat pada dokumen akan di pecah menjadi kata-kata. Untuk ilustrasinya dapat dilihat pada gambar berikut 5.39.

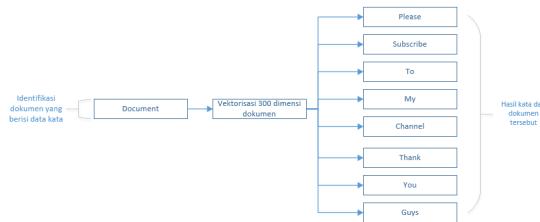


Figure 5.39: Ilustrasi Vektorisasi Dokumen.

5. Jelaskan apa mean dan standar deviasi.

Mean merupakan nilai rata-rata. Untuk mendapatkan mean ini kita tinggal menjumlahkan data yang tersedia lalu dibagi dengan banyaknya data tersebut. Sedangkan standar deviasi adalah nilai statistik yang digunakan untuk menentukan bagaimana sebaran data dalam sampel, dan seberapa dekat titik data individu dengan rata-rata nilai sampel.

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

Skip-gram sama halnya dengan vektorisasi kata, namun untuk skip-gram ia dibalik prosesnya. Yang sebelumnya dari kalimat lalu di olah untuk menemukan salah satu kata, kali ini dari keyword tersebut akan diolah menjadi suatu kalimat yang memiliki keterkaitannya dengan keyword tersebut. Ilustrasinya bisa dilihat pada gambar berikut ??.

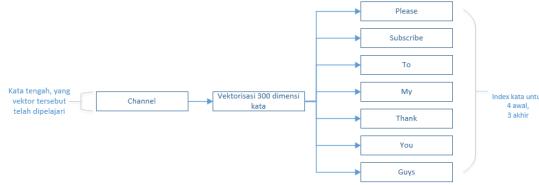


Figure 5.40: Ilustrasi Skip-Gram.

### 5.6.2 Praktikum / Imron Sumadireja / 1164076

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

```
import gensim, logging
logging.basicConfig(format='%(asctime)s : %(levelname)s : %(message)s', level=logging.INFO)
gmodel = gensim.models.KeyedVectors.load_word2vec_format('F:/Imron/Kuliah/Semester 6/Artificial Intelligence-Projects-for-Beginners/Chapter03/GoogleNews-vectors-negative300.bin', binary=True, limit=500000)
```

Hasil keluaran pada gambar 5.41 untuk keluaran ke 99 itu untuk import library gensim. Gensim itu sendiri berguna untuk melakukan pemodelan dengan dataset atau topik yang telah ditentukan. Untuk keluaran 100 logging itu library opsional karena logging hanya untuk menampilkan berupa log untuk setiap code yang dijalankan. Dan keluaran 101 itu hasil load data dari file vektor google itu, disini saya menggunakan limit karena kondisi laptop yang tidak mempunyai untuk melakukan running data sebesar 3 juta file, jadi saya batasi hanya melakukan running 500 ribu data saja.

```
In [99]: import gensim, logging
In [100]: logging.basicConfig(format='%(asctime)s : %(levelname)s : %(message)s', level=logging.INFO)

In [101]: gmodel = gensim.models.KeyedVectors.load_word2vec_format('F:\Imron\Kuliah\Semester 6\Artificial Intelligence-Projects-for-Beginners\Chapter03\GoogleNews-vectors-negative300.bin', binary=True, limit=500000)
2019-03-28 21:21:39,262 : INFO : loading projection weights from F:\Imron\Kuliah\Semester 6\Artificial Intelligence-Projects-for-Beginners\Chapter03\GoogleNews-vectors-negative300.bin
2019-03-28 21:21:44,990 : INFO : loaded (500000, 300) matrix from F:\Imron\Kuliah\Semester 6\Artificial Intelligence-Projects-for-Beginners\Chapter03\GoogleNews-vectors-negative300.bin
```

Figure 5.41: Hasil import gensim dan logging.

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

Keluaran pada gambar 5.42 ini untuk menampilkan data hasil vektorisasi data dari kata love, hasil tersebut berjumlah kurang lebih 300 data. Untuk kata faith, fall, sick, clear, dan lain-lain itu sama saja untuk menampilkan hasil vektorisasi dari setiap katanya. Selanjutnya akan dibandingkan setiap katanya untuk menentukan presentase kemiripan pada setiap kata tersebut.

```
In [102]: gmodel['love']
Out[102]:
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,
```

Figure 5.42: Data vektor dari kata love.

```
gmodel.similarity('love', 'faith')
```

Hasil keluaran pada gambar 5.43 merupakan presentase dari perbandingan kata love dan faith dengan hasil 37%. Hasil tersebut didapatkan dengan 37% karena mesin dapat membaca bahwa kata love dengan faith ini hampir memiliki arti yang sama.

```
In [103]: gmodel.similarity('love', 'faith')
Out[103]: 0.37053479345872803
```

Figure 5.43: Hasil similaritas.

```
gmodel.similarity('fall', 'sick')
```

Hasil keluaran pada gambar 5.44 adalah hasil presentase dari perbandingan antara kata fall dan sick dengan hasil 8%. Hasil tersebut tidak terlalu baik, karena mesin menentukan bahwa kata fall dengan sick itu tidak serupa karena memiliki presentase yang kecil.

```
gmodel.similarity('clear', 'shine')
```

```
In [104]: gmodel.similarity('fall','sick')
Out[104]: 0.08965754281727235
```

Figure 5.44: Hasil similaritas.

Hasil keluaran pada gambar 5.45 merupakan hasil presentase dari perbandingan antara kata clear dengan shine dengan hasil presentase 11%. Presentase tersebut dapat memberikan kita gambaran bahwa mesin dapat menentukan bahwa kata clear dan shine itu berbeda.

```
In [105]: gmodel.similarity('clear','shine')
Out[105]: 0.11652747535830858
```

Figure 5.45: Hasil similaritas.

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

Hasil keluaran pada gambar 5.46 adalah hasil presentase dari perbandingan kata bag dengan wash dan hasil presentasenya 18%. Presentase tersebut cukup bagus karena mesin dapat membedakan kata antara bag dan wash.

```
In [106]: gmodel.similarity('bag','wash')
Out[106]: 0.18425911198018527
```

Figure 5.46: Hasil similaritas.

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

Hasil keluaran pada gambar 5.47 merupakan hasil dari perbandingan kata car dengan motor dan hasil presentasenya 48%. Ini terbilang bagus dan bisa dikatakan mirip karena mesin dapat menentukan presentase yang cukup baik.

```
In [107]: gmodel.similarity('car','motor')
Out[107]: 0.481017283200157
```

Figure 5.47: Hasil similaritas.

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

```

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()

```

Code tersebut berguna untuk menghapus tag-tag html yang tidak diperlukan pada dokumen yang telah dilakukan pelatihan dan vektorisasi, hasilnya pun tidak mengeluarkna apa-apa, hanya pemberitahuan bahwa code tersebut telah berhasil dijalankan. Dan code tersebut akan merubah kalimat yang didalamnya menjadi list kata-kata dalam bentuk array dengan menggunakan 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

```

Code tersebut untuk melakukan pengacakan data agar data tersebut dapat menghasilkan hasil yang mempuni. Code tersebut bisa digunakan atau tidak, tetapi alangkah baiknya untuk digunakan agar hasil akhir yang didapatkan bisa prima.

3. Jelaskan fungsi dari librari gensim TaggedDocument dan Doc2Vec disertai praktek pemakaiannya

```

from gensim.models.doc2vec import TaggedDocument
from gensim.models import Doc2Vec

```

Fungsi dari library gensim untuk pemodelan topik tanpa pengawasan dan pemrosesan bahasa alamai, atau bisa kita sebut dengan unsupervised. Fungsi dari doc2vec itu sendiri ialah untuk membandingkan bobot data yang terdapat pada dokumen yang lainnya, apakah kata-kata didalamnya ada yang sama atau tidak. Lalu untuk tagged document itu memasukan kata-kata pada setiap dokumennya untuk di vektorisasi. Dan hasil dari running coding tersebut tidak mengeluarkan apa-apa, seperti gambar berikut 5.48

```
In [7]: from gensim.models.doc2vec import TaggedDocument
...: from gensim.models import Doc2Vec
```

Figure 5.48: Import library.

4. Jelaskan dengan kata dan praktek cara menambahkan data training dari file yang dimasukkan kepada variabel dalam rangka melatih model doc2vec

```
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]))
```

Untuk menambahkan data training kita melakukan import library os, library os itu sendiri berfungsi untuk melakukan interaksi antara python dengan os laptop kita masing-masing, setelah itu kita buat variable unsup sentences. Selanjutnya pilih direktori tempat data kita disimpan. Selanjutnya itu untuk menyortir data yang terdapat pada folder aclImdb dan membaca file tersebut dengan ekstensi .txt. Hasil dari code pertama tersebut ialah terdapatnya data hasil running dari folder aclImdb 5.49

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

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

Figure 5.49: Menambahkan data training.

Untuk code tersebut sama saja seperti yang sebelumnya, yang membedakan hanya file yang akan digunakan untuk dilakukan training. Hasil dari code tersebut menambahkan data sekitar 60 ribu data, seperti gambar berikut 5.50

unsup_sentences	list	164720	[TaggedDocument, TaggedDocument, TaggedDocument, Tagge ...]
-----------------	------	--------	---

Figure 5.50: Menambahkan data training.

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

Untuk code tersebut sama seperti code sebelumnya yakni untuk menambahkan data training, bagian ketiga ini menambahkan data training sekitar 10 ribu data. Hasilnya seperti gambar berikut 5.51

unsup_sentences	list	175325	[TaggedDocument, TaggedDocument, TaggedDocument, Tagge ...]
-----------------	------	--------	---

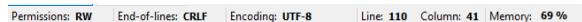
Figure 5.51: Menambahkan data training.

5. Jelaskan dengan kata dan praktek kenapa harus dilakukan pengocokan dan pembersihan data

```
# Pengacakan data
mute = PermuteSentences(unsup_sentences)

# Pembersihan data
model.delete_temporary_training_data(keep_inference=True)
```

Untuk bagian pengacakan data itu berguna untuk mengacak data supaya pada saat data di running bisa berjalan lebih baik dan hasil presentase akhirnya bisa lebih baik. Sedangkan untuk pembersihan data untuk memberikan ruang bagi ram laptop kita setelah melakukan running data sebanyak 3 juta lebih, agar lebih ringan saat proses selanjutnya. Hasil dari pengacakan data tidak ditampilkan pada spyder, namun untuk hasil dari pembersihan data pada gambar berikut 5.52. Dan sebelumnya memori yang terpakai itu sekitar 80% lebih, setelah dikosongkan jadi 64%.



Permissions: RW End-of-lines: CRLF Encoding: UTF-8 Line: 110 Column: 41 Memory: 69 %

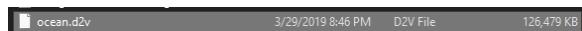
Figure 5.52: Random dan Clear Data.

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

```
# Save data
model.save('ocean.d2v')

# Delete temporary data
model.delete_temporary_training_data(keep_inference=True)
```

Save data ini berfungsi untuk menyimpan file hasil dari proses pelatihan data sebelumnya, model tersebut dilakukan penyimpanan untuk memberikan keringanan pada ram agar saat kita akan melakukan pelatihan lagi, model tersebut tinggal di load saja tanpa harus melakukan pelatihan dari awal dan bisa menghemat waktu untuk hasilnya bisa dilihat pada gambar 5.53. Sedangkan untuk delete temporary training data ini berguna untuk menghapus data latihan yang sebelumnya sudah dilakukan dan disimpan, bertujuan untuk memberikan keringanan pada ram. Karena setelah melakukan proses pelatihan ram biasanya jadi tercekik sampai laptop jadi lag. Itulah fungsi dari delete temporary training data.



ocean.d2v 3/29/2019 8:46 PM D2V File 126,479 KB

Figure 5.53: Save dan Delete Temporary Training Data.

7. Jalankan dengan kata dan praktek maksud dari infer code

```
model.infer_vector(extract_words("jangan lupa tobat guys"))
```

Infer vector itu sendiri berguna untuk membandingkan kata yang tercantum dengan vektor yang mana pada dokumen yang sudah di load pada step sebelumnya. Selain itu infer vector juga untuk menghitung atau mengkalkulasikan vektor dari kata yang dicantumkan dari model yang telah kita buat. Alangkah baiknya kata yang dicantumkan itu lebih panjang lagi agar hasilnya bisa lebih baik lagi. Hasilnya seperti gambar berikut 5.54

```
In [19]: model.infer_vector(extract_words("jangan lupa tobat guys"))
Out[19]:
array([ 0.0934862 , -0.15404993,  0.09099125, -0.08693747,  0.14003305,
       -0.1588757 ,  0.06595911,  0.21125187, -0.06264033,  0.04404573,
       0.18780565, -0.13669884, -0.08058065, -0.12307179,  0.057868 ,
      -0.18482346, -0.17466539,  0.12411116,  0.02510692, -0.07644303,
      -0.17154858,  0.06926782,  0.05917045, -0.0185995 , -0.01474151,
      -0.06697576, -0.11295937,  0.16467911, -0.13769971, -0.07350631,
     0.18245138, -0.23417065, -0.01979371, -0.08834667,  0.05344705,
     -0.11293025,  0.03887824,  0.19377275, -0.07948153,  0.100474 ,
     -0.13933201,  0.11497335,  0.14718999,  0.05711072,  0.11637995,
    -0.21078026, -0.11118771, -0.0962347 ], dtype=float32)
```

Figure 5.54: Infer Code.

8. Jelaskan dengan praktik dan kata maksud dari cosine similarity

```
from sklearn.metrics.pairwise import cosine_similarity
cosine_similarity(
    [model.infer_vector(extract_words("katakataaaaa")),
     [model.infer_vector(extract_words("Services sucks."))])
```

Cosine similarity ini berfungsi untuk membandingkan vektorisasi data diantara kedua kata yang di inputkan, jika hasil presentase dari kedua kata tersebut lebih dari 50% itu memiliki kemungkinan kata tersebut terdapat dalam 1 file. Namun jika kurang dari 50% itu kemungkinan kata tersebut tidak terdapat dalam 1 file. Hasil yang didapatkan pada code tersebut hanya 0.8% itu dikarenakan kata pertama dan kedua tidak memiliki kesamaan vektorisasi dan tidak terdapat pada salah satu dokumen. Untuk hasil presentasenya seperti gambar berikut 5.55

```
In [20]: from sklearn.metrics.pairwise import cosine_similarity
...: cosine_similarity(
...:     [model.infer_vector(extract_words("katakataaaaa")),
...:      [model.infer_vector(extract_words("Services sucks."))])
Out[20]: array([[0.0082403]], dtype=float32)
```

Figure 5.55: Cosine Similarity.

# **Chapter 6**

## **Discussion**

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

# **Chapter 7**

## **Discussion**

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

# **Chapter 8**

## **Discussion**

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

# **Chapter 9**

## **Discussion**

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

# **Chapter 10**

## **Discussion**

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

# **Chapter 11**

## **Discussion**

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

# **Chapter 12**

## **Discussion**

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

# **Chapter 13**

## **Discussion**

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

# **Chapter 14**

## **Discussion**

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

# **Appendix A**

## **Form Penilaian Jurnal**

gambar A.1 dan A.2 merupakan contoh bagaimana reviewer menilai jurnal kita.

NO	UNSUR	KETERANGAN	MAKS	KETERANGAN
1	Keefektifan Judul Artikel	Maksimal 12 (dua belas) kata dalam Bahasa Indonesia atau 10 (sepuluh) kata dalam Bahasa Inggris	2	a. Tidak lugas dan tidak ringkas (0) b. Kurang lugas dan kurang ringkas (1) c. Ringkas dan lugas (2)
2	Pencantuman Nama Penulis dan Lembaga Penulis		1	a. Tidak lengkap dan tidak konsisten (0) b. Lengkap tetapi tidak konsisten (0,5) c. Lengkap dan konsisten (1)
3	Abstrak	Dalam Bahasa Indonesia dan Bahasa Inggris yang baik, jumlah 150-200 kata. Isi terdiri dari latar belakang, metode, hasil, dan kesimpulan. Isi tertuang dengan kalimat yang jelas.	2	a. Tidak dalam Bahasa Indonesia dan Bahasa Inggris (0) b. Abstrak kurang jelas dan ringkas, atau hanya dalam Bahasa Inggris, atau dalam Bahasa Indonesia saja (1) c. Abstrak yang jelas dan ringkas dalam Bahasa Indonesia dan Bahasa Inggris (2)
4	Kata Kunci	Maksimal 5 kata kunci terpenting dalam paper	1	a. Tidak ada (0) b. Ada tetapi kurang mencerminkan konsep penting dalam artikel (0,5) c. Ada dan mencerminkan konsep penting dalam artikel (1)
5	Sistematika Pembahasan	Terdiri dari pendahuluan, tinjauan pustaka, metode penelitian, hasil dan pembahasan, kesimpulan dan saran, daftar pustaka	1	a. Tidak lengkap (0) b. Lengkap tetapi tidak sesuai sistem (0,5) c. Lengkap dan bersistem (1)
6	Pemanfaatan Instrumen Pendukung	Pemanfaatan Instrumen Pendukung seperti gambar dan tabel	1	a. Takermanfaatkan (0) b. Kurang informatif atau komplementer (0,5) c. Informatif dan komplementer (1)
7	Cara Pengacuan dan Pengutipan		1	a. Tidak baku (0) b. Kurang baku (0,5) c. Baku (1)
8	Penyusunan Daftar Pustaka	Penyusunan Daftar Pustaka	1	a. Tidak baku (0) b. Kurang baku (0,5) c. Baku (1)
9	Peristilahan dan Kebahasaan		2	a. Buruk (0) b. Baik (1) c. Cukup (2)
10	Makna Sumbangan bagi Kemajuan		4	a. Tidak ada (0) b. Kurang (1) c. Sedang (2) d. Cukup (3) e. Tinggi (4)

Figure A.1: Form nilai bagian 1.

11	Dampak Ilmiah		7	a. Tidak ada (0) b. Kurang (1) c. Sedang (3) d. Cukup (5) e. Besar (7)
12	Nisbah Sumber Acuan Primer berbanding Sumber lainnya	Sumber acuan yang langsung merujuk pada bidang ilmiah tertentu, sesuai topik penelitian dan sudah teruji.	3	a. < 40% (1) b. 40-80% (2) c. > 80% (3)
13	Derajat Kemutakhiran Pustaka Acuan	Derajat Kemutakhiran Pustaka Acuan	3	a. < 40% (1) b. 40-80% (2) c. > 80% (3)
14	Analisis dan Sintesis	Analisis dan Sintesis	4	a. Sedang (2) b. Cukup (3) c. Baik (4)
15	Penyimpulan	Sangat jelas relevasinya dengan latar belakang dan pembahasan, dirumuskan dengan singkat	3	a. Kurang (1) b. Cukup (2) c. Baik (3)
16	Unsur Plagiat		0	a. Tidak mengandung plagiat (0) b. Terdapat bagian-bagian yang merupakan plagiat (-5) c. Keseluruhannya merupakan plagiat (- 20)
<b>TOTAL</b>			<b>36</b>	
Catatan : Nilai minimal untuk diterima <b>25</b>				

Figure A.2: form nilai bagian 2.

# Appendix B

## FAQ

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

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

M : Pa saya sudah cari jurnal terindeks scopus tapi ga nemu. D : Kamu punya mata de? coba dicolok dulu. Kamu udah lakuin apa aja? tolong di list laporkan ke grup Tingkat Akhir. Tinggal buka google scholar klik dari tahun 2014, cek nama jurnalnya di scimagojr.com beres.

M : Pa saya belum dapat tempat intership, jadi ga tau mau presentasi apa? D : kamu kok ga nyambung, yang dipresentasikan itu yang kamu baca bukan yang akan kamu lakukan.

M : Pa ini jurnal harus yang terindex scopus ga bisa yang lain ? D : Index scopus menandakan artikel tersebut dalam standar semantik yang mudah dipahami dan dibaca serta bukan artikel asal jadi. Jika diluar scopus biasanya lebih sukar untuk dibaca dan dipahami karena tidak adanya proses review yang baik dan benar terhadap artikel.

M : Pa saya tidak mengerti D : Coba lihat standar alasan

M : Pa saya bingung D : Coba lihat standar alasan

M : Pa saya sibuk D : Mbahmu....

M : Pa saya ganteng D : Ndasmu....

M : Pa saya kece D : wes karepmu lah....

Biasanya anda memiliki alasan tertentu jika menghadapi kendala saat proses bimbingan, disini saya akan melakukan standar alasan agar persepsi yang diterima sama dan tidak salah kaprah. Penggunaan kata alasan tersebut antara lain :

1. Tidak Mengerti : anda boleh menggunakan alasan ini jika anda sudah melakukan tahapan membaca dan meresumekan 15 jurnal. Sudah mencoba dan mempraktekkan teorinya dengan mencari di youtube dan google minimal 6 jam sehari selama 3 hari berturut-turut.
2. Bingung : anda boleh mengatakan alasan bingung setelah maksimal dalam berusaha menyelesaikan tugas bimbingan dari dosen(sudah dilakukan semua). Anda belum bisa mengatakan alasan bingung jika anda masih belum menyelesaikan tugas bimbingan dan poin nomor 1 diatas. Setelah anda menyelesaikan tugas bimbingan secara maksimal dan tahap 1 poin diatas, tapi anda masih tetap bingung maka anda boleh memakai alasan ini.

# Bibliography

- [1] Abdillah Baraja. Kecerdasan buatan tinjauan historikal. *Speed-Sentra Penelitian Engineering dan Edukasi*, 1(1), 2008.
- [2] Youssef Bassil. Expert pc troubleshooter with fuzzy-logic and self-learning support. *arXiv preprint arXiv:1204.0181*, 2012.
- [3] 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.
- [4] Stuart J Russell and Peter Norvig. *Artificial intelligence: a modern approach*. Malaysia; Pearson Education Limited,, 2016.
- [5] Kevin Warwick. *Artificial intelligence: the basics*. Routledge, 2013.