

Modul Praktikum

Kecerdasan Buatan



Rolly Maulana Awangga

0410118609

Applied Bachelor of Informatics Engineering

Program Studi D4 Teknik Informatika

Applied Bachelor Program of Informatics Engineering

Politeknik Pos Indonesia

Bandung 2019

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

Acknowledgements

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

Abstract

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

Contents

| | |
|---|-----------|
| 1 Mengenal Kecerdasan Buatan dan Scikit-Learn | 1 |
| 1.1 Teori | 1 |
| 1.2 Instalasi | 2 |
| 1.3 Penanganan Error | 2 |
| 1.4 Fadila/1164072 | 2 |
| 1.4.1 Teori | 2 |
| 1.4.2 Instalasi | 6 |
| 1.4.3 Penanganan Error | 19 |
| 1.5 Rahmi Roza/1164085 | 20 |
| 1.5.1 Teori | 20 |
| 1.5.2 Instalasi | 24 |
| 1.5.3 Penanganan Error | 26 |
| 1.6 Lusia Violita Aprilian/1164080 | 27 |
| 1.6.1 Artificial Intelligence | 27 |
| 1.6.2 Supervised Learning dan Data | 30 |
| 1.6.3 Learning and Predicting | 31 |
| 1.6.4 Model Persistence | 32 |
| 1.6.5 Conventions | 32 |
| 1.6.6 Penanganan Error | 33 |
| 2 Related Works | 35 |
| 2.1 Fadila/1164072 | 35 |
| 2.1.1 Teori | 35 |
| 2.1.2 Praktek Scikit-Learn | 43 |
| 2.1.3 Penanganan Error | 49 |
| 2.2 Lusia Violita Aprilian | 51 |
| 2.2.1 binary classification dilengkapi ilustrasi gambar | 51 |

| | | |
|----------|--|-----------|
| 2.2.2 | supervised learning dan unsupervised learning dan clustering dengan ilustrasi gambar | 52 |
| 2.2.3 | evaluasi dan akurasi dari buku dan disertai ilustrasi contoh dengan gambar | 53 |
| 2.2.4 | bagaimana cara membuat dan membaca confusion matrix, buat confusion matrix | 54 |
| 2.2.5 | bagaimana K-fold cross validation bekerja dengan gambar ilustrasi | 55 |
| 2.2.6 | decision tree dengan gambar ilustrasi | 55 |
| 2.2.7 | Information gain dan entropi dengan gambar ilustrasi | 56 |
| 2.2.8 | binary classification dilengkapi ilustrasi gambar | 56 |
| 2.2.9 | supervised learning dan unsupervised learning dan clustering dengan ilustrasi gambar | 57 |
| 2.2.10 | evaluasi dan akurasi dari buku dan disertai ilustrasi contoh dengan gambar | 59 |
| 2.2.11 | bagaimana cara membuat dan membaca confusion matrix, buat confusion matrix | 59 |
| 2.2.12 | bagaimana K-fold cross validation bekerja dengan gambar ilustrasi | 60 |
| 2.2.13 | decision tree dengan gambar ilustrasi | 61 |
| 2.2.14 | Information gain dan entropi dengan gambar ilustrasi | 61 |
| 2.2.15 | scikit-learn | 62 |
| 2.2.16 | Penanganan Error | 66 |
| 2.3 | Rahmi Roza/1164085 | 67 |
| 2.3.1 | Teori | 67 |
| 2.3.2 | Scikit-learn | 74 |
| 2.3.3 | Penanganan Error | 79 |
| 3 | Methods | 80 |
| 3.1 | Lusia Violita Aprilian/1164080 | 80 |
| 3.1.1 | Teori | 80 |
| 3.1.2 | Praktek | 83 |
| 3.1.3 | Penanganan Error | 88 |
| 3.2 | Rahmi Roza/1164085 | 88 |
| 3.2.1 | Teori | 88 |
| 3.2.2 | Praktek | 92 |

| | |
|--|------------|
| 3.3 Fadila/1164072 | 101 |
| 3.3.1 Teori | 101 |
| 3.3.2 Praktek | 107 |
| 3.3.3 Penanganan Error | 120 |
| 4 Experiment and Result | 130 |
| 4.1 Experiment | 130 |
| 4.2 Result | 130 |
| 4.3 Lusia Violita Aprilian/1164080 | 130 |
| 4.3.1 Teori | 130 |
| 4.3.2 Praktek | 133 |
| 4.3.3 Penanganan Error | 137 |
| 4.4 Rahmi Roza-1164085 | 137 |
| 4.4.1 Teori | 137 |
| 4.4.2 Praktek | 140 |
| 4.4.3 Penanganan Error | 146 |
| 4.5 Fadila-1164072 | 146 |
| 4.5.1 Teori | 146 |
| 4.5.2 Praktek Fadila | 150 |
| 4.5.3 Penanganan Error Fadila | 156 |
| 4.5.4 Praktek | 157 |
| 5 Conclusion | 158 |
| 5.1 Conclusion of Problems | 158 |
| 5.2 Conclusion of Method | 158 |
| 5.3 Conclusion of Experiment | 158 |
| 5.4 Conclusion of Result | 158 |
| 5.5 Rahmi Roza-1164085 | 158 |
| 5.5.1 Teori | 158 |
| 5.6 Fadila-1164072 | 162 |
| 5.6.1 Teori | 162 |
| 6 Discussion | 168 |
| 7 Discussion | 169 |
| 8 Discussion | 170 |

| | |
|--------------------------------|------------|
| 9 Discussion | 171 |
| 10 Discussion | 172 |
| 11 Discussion | 173 |
| 12 Discussion | 174 |
| 13 Discussion | 175 |
| 14 Discussion | 176 |
| A Form Penilaian Jurnal | 177 |
| B FAQ | 180 |
| Bibliography | 182 |

List of Figures

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

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

| | | |
|------|---|----|
| 2.26 | codingan kesembilan | 47 |
| 2.27 | codingan ke-10 | 48 |
| 2.28 | codingan ke-11 | 48 |
| 2.29 | codingan ke-12 | 49 |
| 2.30 | error 1 | 49 |
| 2.31 | error 2 | 50 |
| 2.32 | error 3 | 50 |
| 2.33 | error 4 | 51 |
| 2.34 | Binary Classification | 52 |
| 2.35 | Supervised Learning | 52 |
| 2.36 | Unsupervised Learning | 53 |
| 2.37 | Cluster | 54 |
| 2.38 | Evaluasi dan Akurasi | 54 |
| 2.39 | K-fold cross validation | 55 |
| 2.40 | Decision Tree | 56 |
| 2.41 | Information gain | 56 |
| 2.42 | Entropi | 56 |
| 2.43 | Binary Classification | 57 |
| 2.44 | Supervised Learning | 57 |
| 2.45 | Unsupervised Learning | 58 |
| 2.46 | Cluster | 59 |
| 2.47 | Evaluasi dan Akurasi | 59 |
| 2.48 | K-fold cross validation | 60 |
| 2.49 | Decision Tree | 61 |
| 2.50 | Information gain | 61 |
| 2.51 | Entropi | 62 |
| 2.52 | load dataset | 62 |
| 2.53 | generate binary label | 62 |
| 2.54 | use one-hot encoding on categorical columns | 63 |
| 2.55 | shuffle rows, split training dan testing data | 63 |
| 2.56 | fit a decision tree | 63 |
| 2.57 | visualize tree | 64 |
| 2.58 | visualize tree | 64 |
| 2.59 | save tree | 64 |
| 2.60 | t.score | 65 |
| 2.61 | show average score | 65 |

| | |
|---|----|
| 2.62 for max depth in range | 65 |
| 2.63 depth acc | 66 |
| 2.64 matplotlib | 66 |
| 2.65 error | 67 |
| 2.66 penyelesaian | 67 |
| 2.67 binary classification | 68 |
| 2.68 supervised | 68 |
| 2.69 unsupervised | 69 |
| 2.70 clustering | 69 |
| 2.71 Evaluasi | 70 |
| 2.72 Akurasi | 70 |
| 2.73 Contoh Evaluasi Dan Akurasi Secara Bersamaan | 70 |
| 2.74 confusion matrix | 72 |
| 2.75 k-fold classification 1 | 73 |
| 2.76 decision tree | 73 |
| 2.77 informaion gain 1 | 74 |
| 2.78 entropi | 74 |
| 2.79 Gambar pertama | 75 |
| 2.80 Gambar kedua | 75 |
| 2.81 Gambar Ketiga | 75 |
| 2.82 Gambar Keempat | 76 |
| 2.83 Gambar Kelima | 76 |
| 2.84 Gambar Keenam | 76 |
| 2.85 Gambar Ketujuh | 77 |
| 2.86 Gambar Kedelapan | 77 |
| 2.87 Gambar Kesembilan | 77 |
| 2.88 Gambar Kesepuluh | 78 |
| 2.89 Gambar Kesebelas | 78 |
| 2.90 Gambar Keduabelas | 79 |
| 2.91 Gambar Ketigabelas | 79 |
| 3.1 Random Forest | 80 |
| 3.2 Kode Python | 81 |
| 3.3 Pesan Window Console | 81 |
| 3.4 Dataset | 81 |
| 3.5 Hasil Dataset Cell | 82 |

| | | |
|------|---|----|
| 3.6 | Perintah | 82 |
| 3.7 | Tabel Confusion Matriks | 82 |
| 3.8 | Rumus Confusion Matriks | 83 |
| 3.9 | Confusion Matriks | 83 |
| 3.10 | Voting Pada Random Forest | 83 |
| 3.11 | Aplikasi Pandas | 84 |
| 3.12 | Hasil Pandas | 84 |
| 3.13 | Aplikasi Numpy | 84 |
| 3.14 | Hasil Numpy | 85 |
| 3.15 | Aplikasi Matplotlib | 85 |
| 3.16 | Hasil Matplotlib | 85 |
| 3.17 | Membaca Data File | 86 |
| 3.18 | Melihat Data Sebagian | 86 |
| 3.19 | Melihat Jumlah Data | 87 |
| 3.20 | Mengubah menjadi kolom | 87 |
| 3.21 | Lihat sebagian data awal | 88 |
| 3.22 | Melihat jumlah data | 88 |
| 3.23 | Mengelompokkan burung | 89 |
| 3.24 | Melakukan pivot | 89 |
| 3.25 | Melihat data awal imgid | 90 |
| 3.26 | Melihat jumlah data imgid | 90 |
| 3.27 | Data ciri label dari join | 91 |
| 3.28 | Mengubah menjadi kolom | 91 |
| 3.29 | Melihat isi data frame | 92 |
| 3.30 | Membagi data | 92 |
| 3.31 | Kelas Random Forest | 93 |
| 3.32 | Membangun Random forest | 93 |
| 3.33 | Melihat hasil | 94 |
| 3.34 | Lihat hasil score | 94 |
| 3.35 | Memetakan ke confusion matrix | 95 |
| 3.36 | Melihat hasil | 95 |
| 3.37 | Melakukan Plot | 96 |
| 3.38 | Plotting nama data | 96 |
| 3.39 | Melakukan perintah plot | 97 |
| 3.40 | Klasifikasi menggunakan decision tree | 97 |
| 3.41 | Klasifikasi menggunakan SVM | 98 |

| | |
|--|-----|
| 3.42 Pengecekan cross validation random forest | 98 |
| 3.43 Pengecekan cross validation decision tree | 99 |
| 3.44 Pengecekan cross validation SVM | 99 |
| 3.45 Pengamatan Komponen | 100 |
| 3.46 Plot informasi | 100 |
| 3.47 Skrinsut Error | 101 |
| 3.48 Penyelesaian | 101 |
| 3.49 Hasil | 102 |
| 3.50 Random Forest | 102 |
| 3.51 (b) | 103 |
| 3.52 (c) | 103 |
| 3.53 (d) | 104 |
| 3.54 (e) | 104 |
| 3.55 (h) | 105 |
| 3.56 Confussion Matrik | 105 |
| 3.57 Voting Random forest | 106 |
| 3.58 Pandas | 106 |
| 3.59 Numpy | 107 |
| 3.60 Matplotlib | 108 |
| 3.61 Gambar1 | 108 |
| 3.62 Gambar2 | 109 |
| 3.63 Gambar3 | 109 |
| 3.64 Gambar 4 | 110 |
| 3.65 Gambar 5 | 110 |
| 3.66 Gambar 6 | 111 |
| 3.67 Gambar 7 | 111 |
| 3.68 Gambar 8 | 112 |
| 3.69 Gambar 9 | 112 |
| 3.70 Gambar 10 | 113 |
| 3.71 Gambar 11 | 113 |
| 3.72 Gambar 12 | 114 |
| 3.73 Gambar 13 | 114 |
| 3.74 Gambar 14 | 115 |
| 3.75 Gambar 15 | 115 |
| 3.76 Gambar 16 | 116 |
| 3.77 Gambar 17 | 116 |

| | |
|--|-----|
| 3.78 Gambar 18 | 117 |
| 3.79 Gambar 19 | 117 |
| 3.80 Gambar 20 | 118 |
| 3.81 Gambar 21 | 118 |
| 3.82 Gambar 22 | 119 |
| 3.83 Gambar 23 | 119 |
| 3.84 SVM | 120 |
| 3.85 Decission Tree | 120 |
| 3.86 Cross Validation 1 | 121 |
| 3.87 Cross Validation 2 | 121 |
| 3.88 Cross Validation 3 | 122 |
| 3.89 Program Pengamatan Komponen Informasi 1 | 123 |
| 3.90 Program Pengamatan Komponen Informasi 2 | 123 |
| 3.91 Error | 124 |
| 3.92 random forest | 124 |
| 3.93 random forest 1 | 124 |
| 3.94 random forest 2 | 124 |
| 3.95 random forest 3 | 124 |
| 3.96 random forest 4 | 124 |
| 3.97 cross validation | 125 |
| 3.98 confusion matrix | 125 |
| 3.99 voting random forest | 125 |
| 3.100pandas | 125 |
| 3.101numpy | 125 |
| 3.102matplotlib | 126 |
| 3.103random forest | 126 |
| 3.104random forest 2 | 126 |
| 3.105random forest 3 | 126 |
| 3.106random forest 4 | 126 |
| 3.107random forest 5 | 126 |
| 3.108random forest 6 | 126 |
| 3.109random forest 7 | 126 |
| 3.110random forest 8 | 127 |
| 3.111random forest 9 | 127 |
| 3.112random forest 10 | 127 |
| 3.113random forest 11 | 127 |

| | |
|--|-----|
| 3.114random forest 12 | 127 |
| 3.115random forest 13 | 127 |
| 3.116random forest 14 | 127 |
| 3.117random forest 15 | 127 |
| 3.118random forest 16 | 127 |
| 3.119random forest 17 | 127 |
| 3.120random forest 18 | 127 |
| 3.121confusion matrix 1 | 127 |
| 3.122confusion matrix 2 | 128 |
| 3.123confusion matrix 3 | 128 |
| 3.124confusion matrix 4 | 128 |
| 3.125confusion matrix 5 | 128 |
| 3.126svm | 128 |
| 3.127decision tree | 128 |
| 3.128cross validation 1 | 128 |
| 3.129cross validation 2 | 128 |
| 3.130cross validation 3 | 128 |
| 3.131pengamatan komponen informasi 1 | 129 |
| 3.132pengamatan komponen informasi 2 | 129 |
| 3.133error 1 | 129 |
| 3.134error 2 | 129 |
| 3.135error 3 | 129 |
| | |
| 4.1 Lusia-Klasifikasi teks | 131 |
| 4.2 Lusia-Klasifikasi bunga | 131 |
| 4.3 Lusia-Teknik YouTube | 132 |
| 4.4 Lusia-Bag of Word | 132 |
| 4.5 Lusia-TF IDF | 133 |
| 4.6 Lusia-Pandas | 133 |
| 4.7 Lusia-Hasil Pandas | 133 |
| 4.8 Lusia-Pecah data | 134 |
| 4.9 Lusia-Hasil Pecah data | 134 |
| 4.10 Lusia-Vektorisasi dan klasifikasi | 134 |
| 4.11 Lusia-Decission Tree Katty Perry | 135 |
| 4.12 Lusia-Hasil klasifikasi SVM | 135 |
| 4.13 Lusia-Decission Tree | 135 |

| | |
|--|-----|
| 4.14 Lusia-ploting confusion matrix | 136 |
| 4.15 Lusia-Program cross validation | 136 |
| 4.16 Lusia-Program pengamatan komponen informasi | 136 |
| 4.17 Lusia-skrinsut error | 137 |
| 4.18 Klasifikasi Teks Roza | 138 |
| 4.19 Klasifikasi Bunga Roza | 138 |
| 4.20 Youtube Roza | 139 |
| 4.21 Bag of Words Roza | 140 |
| 4.22 TF-ID Rroza | 141 |
| 4.23 Nomor 1 Roza | 141 |
| 4.24 Hasil 1 Roza | 141 |
| 4.25 Hasil 1 Roza | 141 |
| 4.26 Nomor 2 Roza | 142 |
| 4.27 Hasil 2 Roza | 142 |
| 4.28 Nomor 3 Roza | 142 |
| 4.29 Hasil 3 Roza | 142 |
| 4.30 Nomor 4 Roza | 143 |
| 4.31 Nomor 5 Roza | 143 |
| 4.32 Nomor 6 Roza | 144 |
| 4.33 Nomor 7 Roza | 144 |
| 4.34 Hasil 7 Roza | 144 |
| 4.35 Hasil 7 Roza | 144 |
| 4.36 Nomor 8 Roza | 145 |
| 4.37 Nomor 8 Roza | 145 |
| 4.38 Error Roza | 146 |
| 4.39 text-fadila | 147 |
| 4.40 bunga-fadila | 147 |
| 4.41 youtube-fadila | 148 |
| 4.42 bag-fadila | 149 |
| 4.43 tf2-fadila | 150 |
| 4.44 1-fadila | 151 |
| 4.45 lanjutan1-fadila | 151 |
| 4.46 lanjutan1-1-fadila | 151 |
| 4.47 lanjutan1-2-fadila | 151 |
| 4.48 2-fadila | 152 |
| 4.49 lanjutan2-fadila | 152 |

| | | |
|------|---|-----|
| 4.50 | 3-fadila | 152 |
| 4.51 | 4-fadila | 153 |
| 4.52 | 5-fadila | 154 |
| 4.53 | cod6-fadila | 154 |
| 4.54 | 6-fadila | 154 |
| 4.55 | lanjutan6-fadila | 155 |
| 4.56 | 7-fadila | 155 |
| 4.57 | lanjutan7-fadila | 155 |
| 4.58 | 8-fadila | 155 |
| 4.59 | lanjutan8-fadila | 156 |
| 4.60 | error1-fadila | 157 |
| 5.1 | Vektorisasi Kata Roza | 159 |
| 5.2 | Google Dataset Roza | 160 |
| 5.3 | Vektorisasi Kata Roza | 160 |
| 5.4 | Vektorisasi Dokumen Roza | 161 |
| 5.5 | Mean Roza | 161 |
| 5.6 | Standar Deviasi Roza | 161 |
| 5.7 | Skip-Gram Roza | 162 |
| 5.8 | Plagiarisme Roza | 162 |
| 5.9 | Vektorisasi Kata-fadila | 163 |
| 5.10 | Dimensi Vektor Dataset Wikipedia-fadila | 164 |
| 5.11 | Dimensi Vektor Dataset -fadila | 164 |
| 5.12 | Vektorisasi Untuk Kata-fadila | 164 |
| 5.13 | Vektorisasi Untuk Dokumen-fadila | 165 |
| 5.14 | Mean-fadila | 165 |
| 5.15 | Mean Lanjutan-fadila | 166 |
| 5.16 | Standar Devisiasi-fadila | 166 |
| 5.17 | Skip Gram - fadila | 166 |
| 5.18 | Plagiarisme - fadila | 167 |
| A.1 | Form nilai bagian 1. | 178 |
| A.2 | form nilai bagian 2. | 179 |

Chapter 1

Mengenal Kecerdasan Buatan dan Scikit-Learn

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

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

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

1.1 Teori

Praktek teori penunjang yang dikerjakan :

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

1.2 Instalasi

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

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

1.3 Penanganan Error

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

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

1.4 Fadila/1164072

1.4.1 Teori

Teori mencakup resume dari beberapa pembahasan. yaitu :

1. Tentang Kecerdasan Buatan

- Definisi Kecerdasan Buatan.

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

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

- Sejarah Kecerdasan Buatan

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

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

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

Figure 1.1: error model persistence

2. Penanganan Model Persistence

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



Figure 1.2: capturing

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

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

- Perkembangan Kecerdasan Buatan

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

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

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

3. Tentang Pengertian Terhadap Ilmu Yang Lain

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

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

1.4.2 Instalasi

Untuk Instalasinya mencakup i beberapa pembahasan dan tutorial. yaitu :

1. Instalasi Scikit-Learn Dari Anaconda

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

```
C:\WINDOWS\system32>pip install joblib
Collecting joblib
  Downloading https://files.pythonhosted.org/packages/cd/c1/50a758e8247561e58cb87305b1e90b171b8c767b15b12a1734001f41d350
/joblib-0.13.2-py2.py3-none-any.whl (278kB)
    100% |██████████| 286kB 2.3MB/s
  distributed 0.21.3 requires msgpack, which is not installed.
Installing collected packages: joblib
Successfully installed joblib-0.13.2
You are using pip version 10.0.1, however version 19.0.3 is available.
You should consider upgrading via the 'python -m pip install --upgrade pip' command.
C:\WINDOWS\system32>
```

Figure 1.3: penanganan error model persistence

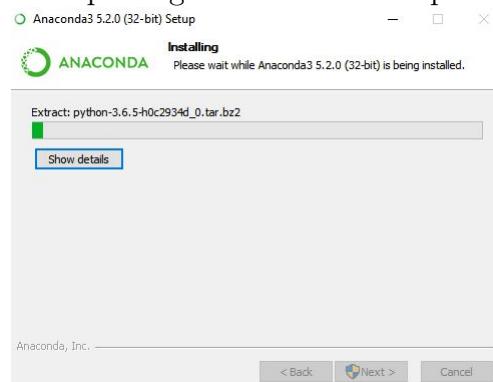
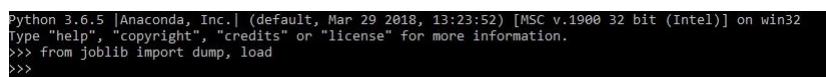


Figure 1.4: install anaconda 1

1. Pengujian Penanganan Model Persistence

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



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

Figure 1.5: penanganan error model persistence

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



Figure 1.6: install anaconda 2

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

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

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



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

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

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

C:\Users\ASUS>
```

Figure 1.7: Pengecekan Anaconda

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



```
!merriweather install -U scikit-learn
Collecting scikit-learn
 Downloading https://files.pythonhosted.org/packages/eec/d1/09e0cb07b8aeef2223aabb21a2129d873f32d47202a2cscrf/scikit_learn-0.20.2-cp36-cp36m-win32.whl (6.3M)
 100% |██████████| 6.3M 1.1MB/s
Requirement already up-to-date: numpy<1.18.0,>=1.17.2 in c:\programdata\anaconda\lib\site-packages (from scikit-learn) (1.17.0)
Requirement already up-to-date: scipy<0.19.0,>=0.17.3 in c:\programdata\anaconda\lib\site-packages (from scikit-learn) (1.1.0)

Installing collected packages: scikit-learn
 Downloading scikit-learn-0.20.2-cp36-cp36m-win32.whl (6.3M)
 100% |██████████| 6.3M 1.1MB/s
 Installing scikit-learn-0.20.2-cp36-cp36m-win32.whl
 Successfully installed scikit-learn-0.20.2
 One or more dependency conflicts were detected while trying to upgrade scikit-learn. To fix them, you can try using --no-deps or --ignore-deps.
```

Figure 1.8: instalasi pip scikit-learn

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

```

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

The following packages will be downloaded:
 package | build
 conda-4.6.7 | py36_0 1.7 MB
The following packages will be UPDATED:
 conda: 4.5.4-py36_0 -> 4.6.7-py36_0
Proceed ((y/n)) y

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

```

Figure 1.9: instalasi conda scikit-learn

```

In [42]: solok = pd.get_dummies(solok, columns=['sex', 'school', 'address', 'famsize', 'Pstatus', 'Mjob',
...: 'Fjob', ...: 'reason', 'guardian', 'schoolsup', 'famsup', 'paid', 'activities', ...: 'nursery', 'higher', 'internet', 'romantic'])
...: ...: solok.head()
Out[42]:
 age Medu Fedu ... internet_yes romantic_no romantic_yes
0 18 4 4 ... 0 1 0
1 17 1 1 ... 1 1 0
2 15 1 1 ... 1 1 0
3 15 4 2 ... 1 0 1
4 16 3 3 ... 0 1 0
[5 rows x 57 columns]

```

Figure 1.10: uji coba codingan

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

- Loading An Example Dataset

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

```

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

```

Figure 1.11: pengujian loading an example dataset

```

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

```

Figure 1.12: pengujian loading an example dataset

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

```

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

```

Figure 1.13: hasil print uji cobat

7. Untuk penjelasan uji cobanya sudah selesai.

- Learning And Predicting

- Penerapan Learning Dan Predicting Pada Python Di CMD

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

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

Figure 1.14: pengujian learning dan predicting

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

7. Penjelasan untuk baris 7 ialah :  
svc ini mengimplementasikan yang namanya data berupa klasifikasi dukungan vektor.
8. Untuk penjelasan uji cobanya sudah selesai.

- Model Persistence

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

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

Figure 1.15: pengujian model persistence 1

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

Figure 1.16: pengujian model persistence 2

4. Contohnya nampak seperti pada gambar.
  5. Apabila tampilanya telah nampak seperti gambar diatas, maka pengujianya telah selesai dan berhasil.
- Penjelasan Perintah Yang Di Uji, ( sesuai dengan contoh perintah pada gambar).
  - Model Persistence 1

1. Penjelasan untuk baris 1 ialah :  
Memanggil ataupun memasukkan datasets dari sklear.
2. Penjelasan untuk baris 2 ialah :  
Memanggil ataupun memasukkan svm dari sklearn
3. Penjelasan untuk baris 3 ialah :  
Membuat variabel baru yaitu clf dimana akan memanggil svm.SVC yang telah mendefinisikan sebuah parameter yaitu gamma.

4. Penjelasan untuk baris 4 ialah :

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

5. Penjelasan untuk baris 5 ialah :

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

6. Penjelasan untuk baris 6 ialah :

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

7. Penjelasan untuk baris 7 ialah :

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

- Model Persistence 2

1. Penjelasan untuk baris 1 ialah :

Melakukan pemanggilan terhadap library pickle

2. Penjelasan untuk baris 2 ialah :

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

3. Penjelasan untuk baris 3 ialah :

Membuat variabel clf2 dengan memanggil pickle loads dengan pendefinisian variabel s

4. Penjelasan untuk baris 4 ialah :

Prediksi nilai baru dari variabel clf2 dengan parameternya yaitu X

5. Penjelasan untuk baris 5 ialah :

set array dari prediksi variabel clf2

6. Penjelasan untuk baris 6 ialah :

Parameter X dengan array 0 akan menghasilkan set array 0 juga

7. Untuk penjelasan uji cobanya sudah selesai.

- Conventions

- Penerapan Conventions Pada Python Di CMD

1. Type Casting

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

\* Selanjutnya masuk ke python

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

```

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

C:\Users\ASUS>python
Python 3.6.5 |Anaconda, Inc.| (default, Mar 29 2018, 13:23:52) [MSC v.1900 32 bit (Intel)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> import numpy as np
>>> from sklearn import manifold
>>> from sklearn.svm import SVC
>>> rng = np.random.RandomState(0)
>>> X = rng.rand(10, 2000)
>>> X = np.array(X, dtype='float32')
>>> X.dtype
dtype('float32')
>>> X_new = manifold.GaussianRandomProjection(n_components=2, random_state=0).fit_transform(X)
>>> X_new
array([[0.125, -0.055], ...])
>>> X_new.dtype
dtype('float64')
>>>

```

Figure 1.17: pengujian type casting 1

```

C:\Users\ASUS>python
Python 3.6.5 |Anaconda, Inc.| (default, Mar 29 2018, 13:23:52) [MSC v.1900 32 bit (Intel)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> import numpy as np
>>> from sklearn import datasets
>>> from sklearn.svm import SVC
>>> iris = datasets.load_iris()
>>> X = iris.data
>>> y = iris.target
>>> clf = SVC(C=1.0, cache_size=200, class_weight=None, coef0=0.0,
 decision_function_shape='ovr', degree=3, gamma='scale', kernel='rbf',
 max_iter=-1, probability=False, random_state=None, shrinking=True,
 tol=0.001, verbose=False)
>>> list(clf.predict(iris.data[:3]))
[0, 1, 2]
>>> clf.fit(iris.data, iris.target)
SVC(C=1.0, cache_size=200, class_weight=None, coef0=0.0,
decision_function_shape='ovr', degree=3, gamma='scale', kernel='rbf',
max_iter=-1, probability=False, random_state=None, shrinking=True,
tol=0.001, verbose=False)
>>> list(clf.predict(iris.data[-3:]))
['setosa', 'setosa', 'setosa']
>>>

```

Figure 1.18: pengujian type casting 2

## 2. Refitting And Updating Parameters

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

```

C:\Users\ASUS>python
Python 3.6.5 |Anaconda, Inc.| (default, Mar 29 2018, 13:23:52) [MSC v.1900 32 bit (Intel)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> import numpy as np
>>> from sklearn import SVC
>>> rng = np.random.RandomState(0)
>>> X = rng.rand(100, 10)
>>> y = rng.binomial(1, 0.5, 100)
>>> X_test = rng.rand(5, 10)
>>> clf = SVC()
>>> clf.set_params(kernel='linear').fit(X, y)
SVC(C=1.0, cache_size=200, class_weight=None, coef0=0.0,
decision_function_shape='ovr', degree=3, gamma='auto_deprecated',
kernel='linear', max_iter=-1, probability=False, random_state=None,
shrinking=True, tol=0.001, verbose=False)
>>> clf.predict(X_test)
array([1, 0, 1, 1, 0])
>>> clf.set_params(kernel='rbf', gamma='scale').fit(X, y)
SVC(C=1.0, cache_size=200, class_weight=None, coef0=0.0,
decision_function_shape='ovr', degree=3, gamma='scale', kernel='rbf',
max_iter=-1, probability=False, random_state=None, shrinking=True,
tol=0.001, verbose=False)
>>> clf.predict(X_test)
array([1, 0, 1, 1, 0])
>>>

```

Figure 1.19: pengujian refitting and updating

## 3. Multiclass And Multilable Fitting

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

```

C:\Users\ASUS>python
Python 3.6.5 |Anaconda, Inc.| (default, Mar 29 2018, 13:23:52) [MSC v.1900 32 bit (Intel)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> from sklearn import SVC
>>> from sklearn.multiclass import OneVsRestClassifier
>>> from sklearn.preprocessing import LabelBinarizer
>>> X = [[1, 2], [2, 4], [4, 5], [3, 2], [3, 1]]
>>> y = [0, 0, 1, 1, 2]
>>> classif = OneVsRestClassifier(estimator=SVC(gamma='scale', random_state=0))
>>> classif.fit(X, y).predict(X)
array([0, 0, 1, 1, 2])
>>> y = LabelBinarizer().fit_transform(y)
>>> classif.fit(X, y).predict(X)
array([[1, 0, 0],
 [0, 1, 0],
 [0, 0, 1],
 [0, 0, 1],
 [0, 0, 0]])

```

Figure 1.20: pengujian multiclass and multiable 1

```

>>> from sklearn.preprocessing import MultiLabelBinarizer
>>> y = [[0, 1], [0, 2], [1, 3], [0, 2, 3], [2, 4]]
>>> y = MultiLabelBinarizer().fit_transform(y)
>>> classif.fit(X, y).predict(X)
array([[1, 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 1.21: pengujian multiclass and multiable 2

- \* Setelah masuk kedalam python, silahkan masukkan perintah ( scriptsnya ) sesuai dengan contoh yang akan diberikan
- 4. Apabila semua proses yang telah dilakukan terlihat seperti contoh-contoh diatas, maka pengujian telah selesai.
- Penjelasan Perintah Yang Di Uji, berdasarkan contoh perintah-perintah diatas :
- 1. Type Casting :
  - \* Type Casting 1
    - Penjelasan untuk baris 1 ialah :  
Memasukkan dan memanggil module / library numpy sebagai np
    - Penjelasan untuk baris 2 ialah :  
Memasukkan dan memanggil random projection dari sklearn
    - Penjelasan untuk baris 3 ialah :  
Membuat variabel rng, dimana memanggil np yang akan mengambil dan mengeksekusi random state dengan parameter 0
    - Penjelasan untuk baris 4 ialah :  
Membuat variabel baru lainnya yaitu X dengan memanggil variabel rng dengan 2 parameter yaitu 10 dan 2000
    - Penjelasan untuk baris 5 ialah :  
Membuat variabel X lagi namun dengan pemanggilan yang berbeda yaitu array dari np dengan parameternya x dan dtype='float32'.

- Penjelasan untuk baris 6 ialah :  
Pemanggilan dtype dari variabel X
- Penjelasan untuk baris 7 ialah :  
Hasil dari pemanggilan dtype dari variabel X
- Penjelasan untuk baris 8 ialah :  
Membuat variabel transformer dengan pemanggilan gaussian random projection
- Penjelasan untuk baris 9 ialah :  
Membuat variabel baru yaitu X new dengan memanggil variabel transformer yang berada pada model fit metode dengan parameternya yaitu X
- Penjelasan untuk baris 10 ialah :  
Pemanggilan dtype dari variabel X new
- Penjelasan untuk baris 11 ialah :  
Hasil dari pemanggilan dtype variabel X new
- Untuk penjelasan uji cobanya sudah selesai.

#### \* Type Casting 2

- Penjelasan untuk baris 1 ialah :  
Memasukkan dan memanggil dataset dari sklearn
- Penjelasan untuk baris 2 ialah :  
Memasukkan dan memanggil csv dari sklearn
- Penjelasan untuk baris 3 ialah :  
Membuat variabel iris dengan memanggil load iris dari datasets
- Penjelasan untuk baris 4 ialah :  
Membuat variabel clf dengan memanggil svc dengan parameter gamma
- Penjelasan untuk baris 5 ialah :  
Membuat clf pada module fit metode dengan 2 parameter yaitu iris data dan iris target
- Penjelasan untuk baris 6 ialah :  
membuat variabel list dengan prediksi clf
- Penjelasan untuk baris 7 ialah :  
SVC mengimplementasikan yang namanya data berupa klasifikasi dukungan vektor.

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

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

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

### 1.4.3 Penanganan Error

Terdapat error pada pengujian diatas dan penanganannya, yaitu:

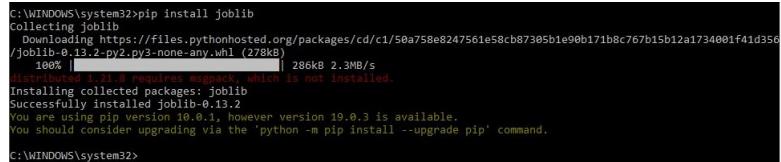
1. Model Persistence
  - Errornya ditandai dengan tidak terdefinisinya module joblib pada komputer
  - Hal itulah yang menyebabkan tidak terprosesnya perintah terkait

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

Figure 1.22: error model persistence

## 2. Penanganan Model Persistence

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

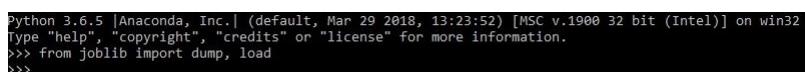


```
C:\WINDOWS\system32>pip install joblib
Collecting joblib
 Downloading https://files.pythonhosted.org/packages/cd/c1/50a758e8247561e58cb87305b1e90b171b8c767b15b12a1734001f41d350
/joblib-0.13.2-py2.py3-none-any.whl (278kB)
 100% |██████████| 286kB 2.3MB/s
Requirement already up-to-date: six<2.0,>=1.5.2 in ./pip-req-build which is not installed.
Installing collected packages: joblib
Successfully installed joblib-0.13.2
You are using pip version 10.0.1, however version 19.0.3 is available.
You should consider upgrading via the 'python -m pip install --upgrade pip' command.
C:\WINDOWS\system32>
```

Figure 1.23: penanganan error model persistence

## 1. Pengujian Penanganan Model Persistence

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



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

Figure 1.24: penanganan error model persistence

## 1.5 Rahmi Roza/1164085

### 1.5.1 Teori

Teori mencakup resume dari beberapa pembahasan. yaitu :

- Definisi Kecerdasan Buatan.

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

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

- Sejarah Kecerdasan Buatan

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

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

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

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

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

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

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

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

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

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

#### – Perkembangan Kecerdasan Buatan

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

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

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

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

- Tentang Pengertian Terhadap Ilmu Yang Lain

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

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

### 1.5.2 Instalasi



Figure 1.25: gambar1

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



Figure 1.26: gambar2

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



Figure 1.27: gambar3



Figure 1.28: gambar4

2. Baris 2 = set array

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

1. Baris 1 = memasukkan dan memanggil numpy sebagai np



Figure 1.29: gambar5

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

### 1.5.3 Penanganan Error

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



Figure 1.30: gambar6



Figure 1.31: gambar7

## 1.6 Lusia Violita Aprilian/1164080

### 1.6.1 Artificial Intelligence

#### 1. Pengertian AI

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

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

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

## 2. Sejarah dan Perkembangan

- 1941 ( Era Komputer Elektronik )

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

- 1943 – 1956 ( Era Persiapan AI )

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

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

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

- 1952 – 1969 ( Awal Perkembangan )

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

Pada tahun 1959, Program komputer bernama General Problem Solver

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

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

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

- 1966 – 1974 ( Perkembangan AI Lambat )

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

- 1969 – 1979 ( Sistem berbasis pengetahuan )

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

- 1980 – 1988 ( AI menjadi Industri )

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

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

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

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

## 1.6.2 Supervised Learning dan Data

### 1. Supervised Learning

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

### 2. Klasifikasi

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

### 3. Regresi

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

#### 4. Unsupervised Learning

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

#### 5. Data Set

Objek yang merepresentasikan data dan relasi didalam memori.

#### 6. Training Set

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

#### 7. Testing Set

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

### 1.6.3 Learning and Predicting

#### 1. Learning and predicting 1

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

Figure 1.32: Learning and predicting 1

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

#### 2. Learning and predicting 2

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

Figure 1.33: Learning and predicting 2

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

#### 3. Learning and predicting 3

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

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

Figure 1.34: Learning and predicting 3

#### 1.6.4 Model Persistence

- menjelaskan maksud dari tulisan tersebut dan mengartikan per baris.

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

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

Figure 1.35: Model Persistence

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

#### 1.6.5 Conventions

- menjelaskan maksud dari tulisan tersebut dan mengartikan per baris.

- Baris 1 = memasukkan dan memanggil numpy sebagai np

```

import numpy as np
from sklearn import random_projection

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

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

```

Figure 1.36: Conventions

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

### 1.6.6 Penanganan Error

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

1. skrinsut error

```

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

```

Figure 1.37: skrinsut error

2. Tuliskan kode eror dan jenis errornya

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

3. Solusi pemecahan masalah error

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

Figure 1.38: gb 1

# Chapter 2

## Related Works

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

### 2.1 Fadila/1164072

#### 2.1.1 Teori

Penyelesaian Tugas Harian 3 ( No. 1-7 )

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

- Pengertian Binary Classification / Klasifikasi Biner:

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

- Ilustrasi Gambar Binary Classification :

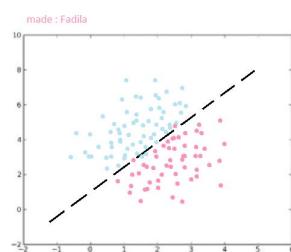


Figure 2.1: binary classification

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

- Pengertian Supervised Learning :

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

- Ilustrasi Gambar Supervised Learning :



Figure 2.2: supervised

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

- Pengertian Unsupervised Learning :

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

- Ilustrasi Gambar Unsupervised Learning :

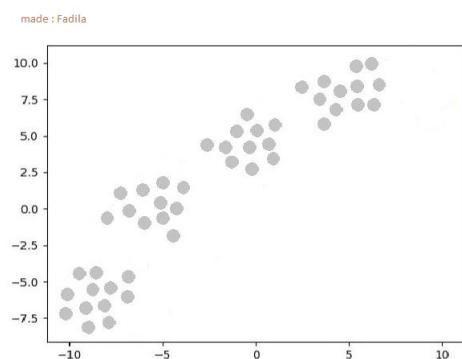


Figure 2.3: unsupervised

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

- Pengertian Clustering :

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

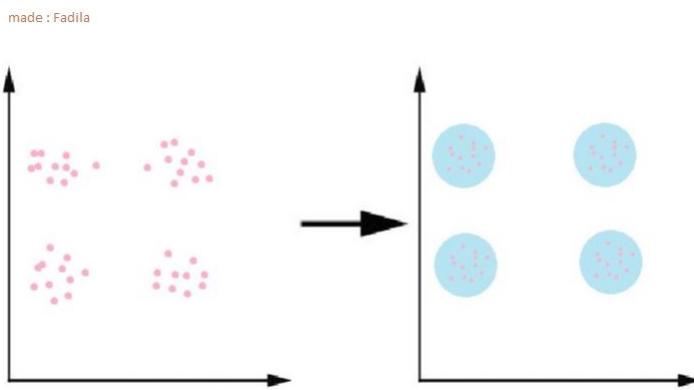


Figure 2.4: clustering

### 3. Evaluasi, Akurasi Dan Ilustrasi Gambar

- Pengertian Evaluasi

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

|                      | <b>Prediksi "Apel"</b> | <b>Prediksi "Jeruk"</b> |
|----------------------|------------------------|-------------------------|
| <b>Benar "Apel"</b>  | 20                     | 5                       |
| <b>Benar "Jeruk"</b> | 3                      | 22                      |

Figure 2.5: Evaluasi

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

- Pengertian Akurasi

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

$$\text{Rumus dari accuracy} = (a+c)/(a+b+c+d)$$

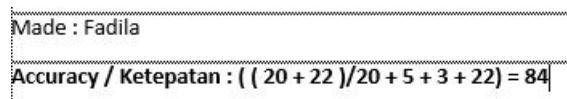


Figure 2.6: Akurasi

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

#### Contoh penggabungan Akurasi Dan Evaluasi

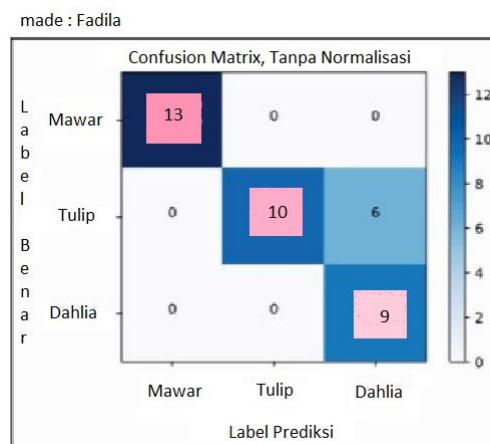


Figure 2.7: Contoh Evaluasi Dan Akurasi Secara Bersamaan

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

- Pengertian Confusion Matrix

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

- Pembacaan Confusion Matrix

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

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

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

- Pembuatan Confusion Matrix

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

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

(c) Kelompokkan klasifikasi tersebut bisa menggunakan klasifikasi biner

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

(e) Contoh dasarnya nampak seperti langkah diatas

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

- Ilustrasi Gambar

| <b>Y</b> | <b>Y Pred</b> | <b>Keluaran Untuk<br/>( threshold ) 0.6</b> | <b>Recall ( pemanajilan / penarikan )</b> | <b>Precision ( Presisi )</b> |
|----------|---------------|---------------------------------------------|-------------------------------------------|------------------------------|
| 0        | 0.5           | 0                                           |                                           |                              |
| 1        | 0.9           | 1                                           |                                           |                              |
| 0        | 0.7           | 1                                           |                                           |                              |
| 1        | 0.7           | 1                                           |                                           |                              |
| 1        | 0.3           | 0                                           |                                           |                              |
| 0        | 0.4           | 0                                           |                                           |                              |
| 1        | 0.5           | 0                                           |                                           |                              |

Made : Fadila

Figure 2.8: confusion matrix

– Penjelasan

(a) Recall

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

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

# made : Fadila

Figure 2.9: recall

(b) Presisi / Precision

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

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

# made : Fadila

Figure 2.10: precision

(c) F-Ukur ( measure )

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

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

# made : Fadila

Figure 2.11: f-measure

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

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

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

- Ilustrasi Gambar

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

# made : Fadila

Figure 2.12: k-fold classification 1

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

# made : Fadila

Figure 2.13: k-fold classification 2

## 6. Decision Tree Dan Ilustrasi Gambar

- Pengertian Decision Tree

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

- Ilustrasi Gambar



made : Fadila

Figure 2.14: decision tree

## 7. Information Gain Dan Entropi

- Pengertian Information Gain

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

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

- Ilustrasi Gambar

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

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

Figure 2.15: informaion gain 1

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

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

Figure 2.16: information gain 2

- Penjelasan :

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

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

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

- Pengertian Entropi

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

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

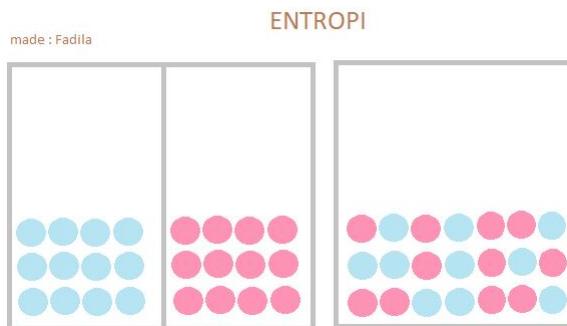


Figure 2.17: entropi

### 2.1.2 Praktek Scikit-Learn

Penyelesaian Tugas Harian 4 ( No. 1-12 )

(a) Pembahasan Codingan Dan Hasilnya

i. Codingan Pertama :

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

- Hasil Codingan Pertama :

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

Figure 2.18: codingan pertama

### ii. Codingan Kedua :

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

- Hasil Codingan Kedua :

```
In [60]: bakso['pass'] = bakso.apply(lambda row: 1 if (row['G1']+row['G2']
+row['G3']) >= 35 else 0, axis=1)
...: bakso = bakso.drop(['G1', 'G2', 'G3'], axis=1)
...: bakso.head()
Out[60]:
 school sex age address famsize ... Dalc Walc health absences pass
0 GP F 18 U GT3 ... 1 1 3 6 0
1 GP F 17 U GT3 ... 1 1 3 4 0
2 GP F 15 U LE3 ... 2 3 3 10 0
3 GP F 15 U GT3 ... 1 1 5 2 1
4 GP F 16 U GT3 ... 1 2 5 4 0

[5 rows x 31 columns]
```

Figure 2.19: codingan kedua

### iii. Codingan Ketiga :

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

- Hasil Codingan Ketiga :

```

In [61]: bakso = pizza.get_dummies(bakso, columns=['sex', 'school', 'address', 'famsize', 'Pstatus', 'Mjob', 'Fjob',
...: 'reason', 'guardian', 'schoolsup', 'famsup', 'paid', 'activities',
...: 'nursery', 'higher', 'internet', 'romantic'])
...: bakso.head()
Out[61]:
 age Medu Fedu ... internet_yes romantic_no romantic_yes
0 18 4 4 ... 0 1 0
1 17 1 1 ... 1 1 0
2 15 1 1 ... 1 1 0
3 15 4 2 ... 1 0 1
4 16 3 3 ... 0 1 0
[5 rows x 57 columns]

```

Figure 2.20: codingan ketiga

#### iv. Codingan Keempat :

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

- Hasil Codingan Keempat :

```

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

```

Figure 2.21: codingan keempat

#### v. Codingan Kelima :

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

- Hasil Codingan Kelima :

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

Figure 2.22: codingan kelima

#### vi. Codingan Keenam :

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

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

- Hasil Codingan Keenam :

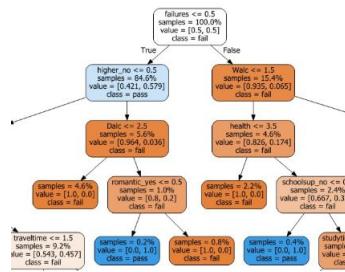


Figure 2.23: codingan keenam

#### vii. Codingan Ketujuh :

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

- Hasil Codingan Ketujuh :

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

Figure 2.24: codingan ketujuh

#### viii. Codingan Kedelapan :

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

- Hasil Codingan Kedelapan :

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

Figure 2.25: codingan kedelapan

ix. Codingan Kesembilan :

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

- Hasil Codingan Kesembilan :

```
In [69]: from sklearn.model_selection import cross_val_score
...: scores = cross_val_score(sate, bakso_att, bakso_pass, cv=5)
...: # show average score and +/- two standard deviations away (covering 95% of scores)
...: print("Accuracy: %0.2f (+/- %0.2f)" % (scores.mean(), scores.std() * 2))
Accuracy: 0.57 (+/- 0.11)
```

Figure 2.26: codingan kesembilan

x. Codingan Ke-10 :

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

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

- Hasil Codingan Ke-10 :

```
In [70]: for max_depth in range(1, 20):
 ...
 state = tree.DecisionTreeClassifier(criterion="entropy", max_depth=max_depth)
 ...
 scores = cross_val_score(state, bako_att, bako_pass, cv=5)
 ...
 print("Max depth: %d, Accuracy: %.2f +/- %.2f" % (max_depth, scores.mean(), scores.std() * 2))
Max depth: 1, Accuracy: 0.54 (+/- 0.01)
Max depth: 2, Accuracy: 0.55 (+/- 0.01)
Max depth: 3, Accuracy: 0.57 (+/- 0.06)
Max depth: 4, Accuracy: 0.57 (+/- 0.06)
Max depth: 5, Accuracy: 0.57 (+/- 0.09)
Max depth: 6, Accuracy: 0.57 (+/- 0.09)
Max depth: 7, Accuracy: 0.58 (+/- 0.09)
Max depth: 8, Accuracy: 0.58 (+/- 0.09)
Max depth: 9, Accuracy: 0.57 (+/- 0.09)
Max depth: 10, Accuracy: 0.57 (+/- 0.09)
Max depth: 11, Accuracy: 0.57 (+/- 0.08)
Max depth: 12, Accuracy: 0.55 (+/- 0.09)
Max depth: 13, Accuracy: 0.55 (+/- 0.09)
Max depth: 14, Accuracy: 0.56 (+/- 0.09)
Max depth: 15, Accuracy: 0.57 (+/- 0.12)
Max depth: 16, Accuracy: 0.57 (+/- 0.12)
Max depth: 17, Accuracy: 0.56 (+/- 0.09)
Max depth: 18, Accuracy: 0.58 (+/- 0.12)
Max depth: 19, Accuracy: 0.57 (+/- 0.12)
```

Figure 2.27: codingan ke-10

xi. Codingan Ke-11 :

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

- Hasil Codingan Ke-11 :

```
In [55]: fadila = np.empty((19,3), float)
.... i = 0
.... for max_depth in range(1, 20):
.... ...
.... state = tree.DecisionTreeClassifier(criterion="entropy",
.... max_depth=max_depth)
.... ...
.... scores = cross_val_score(state, bako_att, bako_pass, cv=5)
.... ...
.... fadila[i,0] = max_depth
.... fadila[i,1] = scores.mean()
.... fadila[i,2] = scores.std() * 2
.... ...
.... i += 1
....
....
.... fadila
out[55]:
array([1.000000e+00, 5.793178e-01, 0.30765599e-01],
[2.000000e+00, 5.5684832e-01, 1.11664034e-01],
[3.000000e+00, 5.2895998e-01, 0.87785678e-02],
[4.000000e+00, 5.1264265e-01, 1.09098399e-01],
[5.000000e+00, 5.0460354e-01, 1.49349115e-01],
[6.000000e+00, 5.0456342e-01, 4.16227110e-02],
[7.000000e+00, 5.0441034e-01, 1.89974107e-01],
[8.000000e+00, 5.0438272e-01, 1.88282540e-01],
[9.000000e+00, 5.04374391e-01, 1.89761594e-01],
[1.000000e+01, 5.04374391e-01, 1.89761594e-01],
[1.100000e+01, 5.02162447e-01, 1.35161115e-01],
[1.200000e+01, 5.00995226e-01, 1.19212183e-01],
[1.300000e+01, 5.00000000e+00, 1.00000000e+00],
[1.400000e+01, 5.11985226e-01, 9.18997401e-02],
[1.500000e+01, 5.04849086e-01, 0.65923726e-02],
[1.600000e+01, 5.04424556e-01, 0.65923726e-02],
[1.700000e+01, 5.042449351e-01, 0.81899779e-02],
[1.800000e+01, 5.04093451e-01, 0.22088877e-01],
[1.900000e+01, 5.04093451e-01, 0.22183932e-01])
```

Figure 2.28: codingan ke-11

xii. Codingan Ke-12 :

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

• Hasil Codingan Ke-12 :

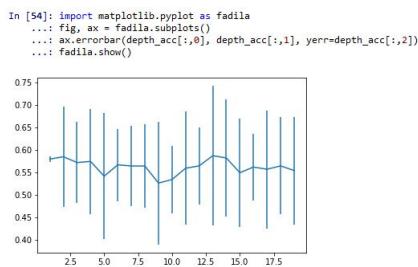


Figure 2.29: codingan ke-12

### 2.1.3 Penanganan Error

#### Pembahasan dan Penyelesaian Error

(a) Error 1 :

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

Figure 2.30: error 1

• Penjelasan :

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

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

(b) Error 2 :

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

Figure 2.31: error 2

- Penjelasan :

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

(c) Error 3 :

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

Figure 2.32: error 3

- Penjelasan :

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

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

(d) Error 4 :

```
File "C:\ProgramData\Anaconda\lib\site-packages\sklearn\tree\tree.py", line 377, in _validate_X_predict
 X = check_array(X, dtype=DTYPE, accept_sparse="csr")
 File "C:\ProgramData\Anaconda\lib\site-packages\sklearn\utils\validation.py", line 582, in check_array
 context)
ValueError: unsupported format character 'b' (0x62) at index 18
```

Figure 2.33: error 4

- Penjelasan :

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

## 2.2 Lusia Violita Aprilian

### 2.2.1 binary classification dilengkapi ilustrasi gambar

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

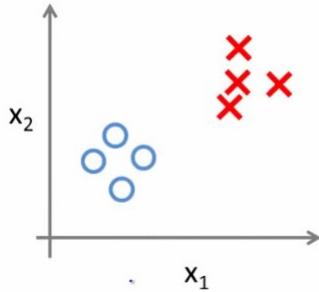


Figure 2.34: Binary Classification

### **2.2.2 supervised learning dan unsupervised learning dan clustering dengan ilustrasi gambar**

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

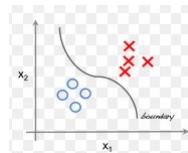


Figure 2.35: Supervised Learning

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.

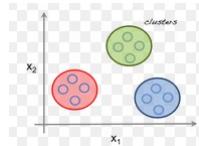


Figure 2.36: Unsupervised Learning

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.

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

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

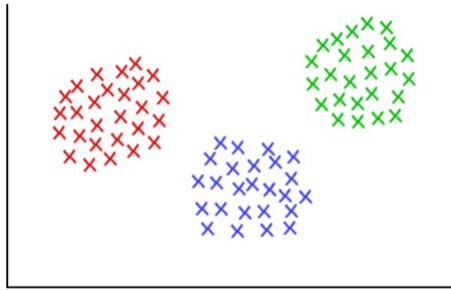


Figure 2.37: Cluster

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

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

Figure 2.38: Evaluasi dan Akurasi

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

1. Cara membuat dan membaca confusion matrix :
  - (a) Tentukan pokok permasalahan dan atributanya, misal gaji dan listik.
  - (b) Buat pohon keputusan
  - (c) Lalu data testingnya
  - (d) Lalu mencari nilai a, b, c, dan d. Semisal a = 5, b = 1, c = 1, dan d = 3.
  - (e) 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 =  $(1+1)/(5+1+1+3) = 0,2$

## 2.2.5 bagaimana K-fold cross validation bekerja dengan gambar ilustrasi

Cara kerja K-fold cross validation :

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



Figure 2.39: K-fold cross validation

## 2.2.6 decision tree dengan gambar ilustrasi

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

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

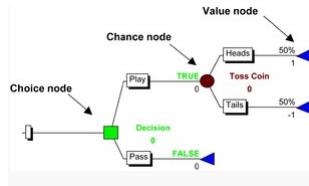


Figure 2.40: Decision Tree

### 2.2.7 Information gain dan entropi dengan gambar ilustrasi

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

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

Figure 2.41: Information gain

- Entropi merupakan kemurnian dalam koleksi contoh yang sewenang-wenang.

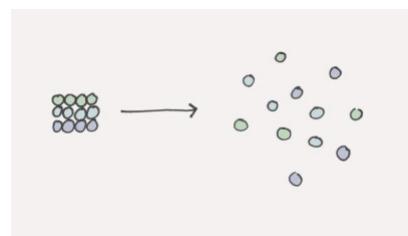


Figure 2.42: Entropi

### 2.2.8 binary classification dilengkapi ilustrasi gambar

Binary classification yaitu berupa kelas positif dan kelas negatif. Klasifikasi biner adalah dikotomisasi yang diterapkan untuk tujuan praktis, dan dalam banyak masalah

klasifikasi biner praktis, kedua kelompok tidak simetris - daripada akurasi keseluruhan, proporsi relatif dari berbagai jenis kesalahan yang menarik. Misalnya, dalam pengujian medis, false positive (mendeteksi penyakit ketika tidak ada) dianggap berbeda dari false negative (tidak mendeteksi penyakit ketika hadir).

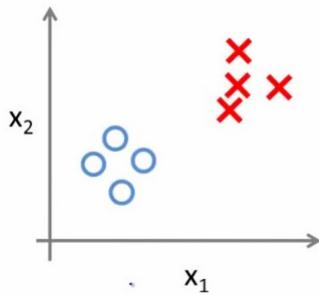


Figure 2.43: Binary Classification

### **2.2.9 supervised learning dan unsupervised learning dan clustering dengan ilustrasi gambar**

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

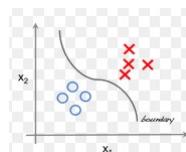


Figure 2.44: Supervised Learning

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.

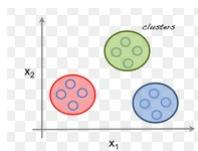


Figure 2.45: Unsupervised Learning

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.

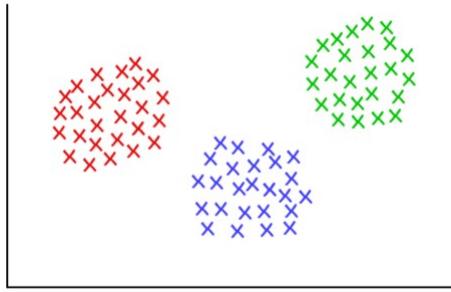


Figure 2.46: Cluster

### **2.2.10 evaluasi dan akurasi dari buku dan disertai ilustrasi contoh dengan gambar**

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

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

Figure 2.47: Evaluasi dan Akurasi

### **2.2.11 bagaimana cara membuat dan membaca confusion matrix, buat confusion matrix**

1. Cara membuat dan membaca confusion matrix :
  - (a) 1) Tentukan pokok permasalahan dan atributanya, misal gaji dan listik.
  - (b) 2) Buat pohon keputusan
  - (c) 3) Lalu data testingnya
  - (d) 4) Lalu mencari nilai a, b, c, dan d. Semisal a = 5, b = 1, c = 1, dan d = 3.

- (e) 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 =  $(1+1)/(5+1+1+3) = 0,2$

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

Cara kerja K-fold cross validation :

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



Figure 2.48: K-fold cross validation

### 2.2.13 decision tree dengan gambar ilustrasi

Decision tree adalah model visual yang terdiri dari node dan cabang, seperti Gambar dijelaskan secara rinci nanti dalam artikel ini. Untuk saat ini, amati bahwa ia tumbuh dari kiri ke kanan, dimulai dengan simpul keputusan root (kuadrat, juga disebut simpul pilihan) yang cabang-cabangnya mewakili dua atau lebih opsi bersaing yang tersedia bagi para pembuat keputusan. Pada akhir cabang awal ini, ada simpul akhir (segitiga, juga disebut simpul nilai) atau simpul ketidakpastian (lingkaran, juga disebut simpul peluang). Node akhir mewakili nilai tetap. Cabang lingkaran mewakili hasil yang mungkin bersama dengan probabilitasnya masing-masing (yang berjumlah 1,0). Di luar cabang-cabang node ketidakpastian awal ini, mungkin ada lebih banyak bujur sangkar dan lebih banyak lingkaran, yang umumnya bergantian sampai setiap jalur berakhir di simpul akhir.

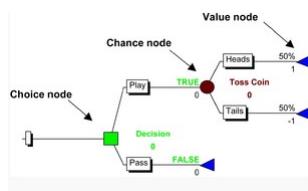


Figure 2.49: Decision Tree

### 2.2.14 Information gain dan entropi dengan gambar ilustrasi

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

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

Figure 2.50: Information gain

2. Entropi merupakan kemurnian dalam koleksi contoh yang sewenang-wenang.

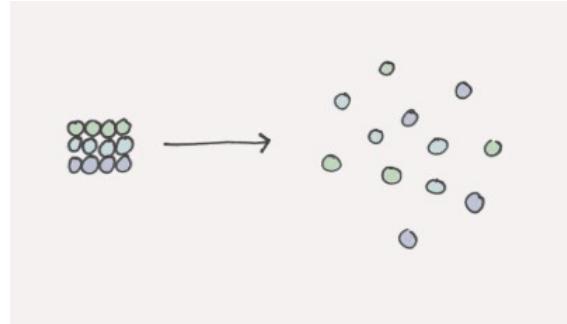


Figure 2.51: Entropi

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

Figure 2.52: load dataset

### 2.2.15 scikit-learn

#### 1. load dataset

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

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

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

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

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

Figure 2.53: generate binary label

```

In [7]: nanas = pd.get_dummies(nanas, columns=['sex', 'school', 'address', 'famsize',
...: 'Pstatus', 'Mjob', 'Fjob',
...: ...,
...: 'paid', 'activities',
...: ...,
...: nanas.head())
Out[7]:
 age Medu Fedu ... internet_yes romantic_no romantic_yes
0 18 4 4 ... 0 1 0
1 17 1 1 ... 1 1 0
2 15 1 1 ... 1 1 0
3 15 4 2 ... 1 0 1
4 16 3 3 ... 0 1 0
[5 rows x 57 columns]

In [8]:

```

Figure 2.54: use one-hot encoding on categorical columns

```

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

```

Figure 2.55: shuffle rows, split training dan testing data

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

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

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

#### 5. fit a decision tree

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

```

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

In [12]:

```

Figure 2.56: fit a decision tree

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

Figure 2.57: visualize tree

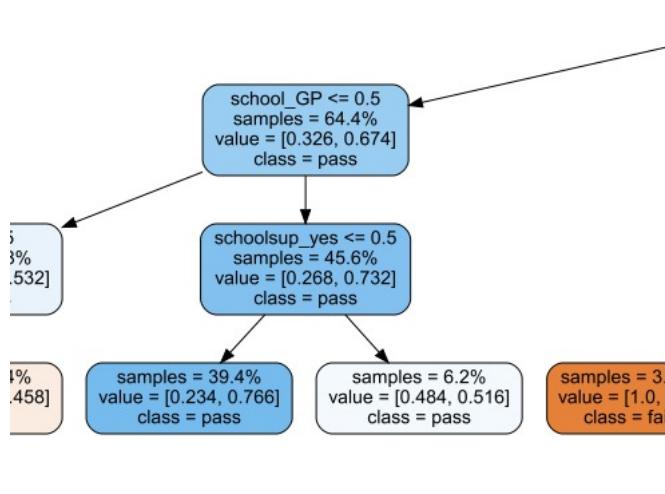


Figure 2.58: visualize tree

terakhir, variable duku memanggil duku dengan fit nanas train att dan nanas train pass.

## 6. visualize tree

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

## 7. save tree

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

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

---

```
In [16]: _____
```

Figure 2.59: save tree

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

Figure 2.60: t.score

```
In [14]: from sklearn.model_selection import cross_val_score
...: scores = cross_val_score(duku, nanas_att, nanas_pass, cv=5)
...: # show average score and +/- two standard deviations away (covering 95% of
...: scores)
...: print("Accuracy: %0.2f (+/- %0.2f)" % (scores.mean(), scores.std() * 2))
Accuracy: 0.59 (+/- 0.10)

In [15]:
```

Figure 2.61: show average score

## 8. t.score

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

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

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

## 10. for max depth in range

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

## 11. depth acc

```
In [16]: for max_depth in range(1, 20):
...: duku = tree.DecisionTreeClassifier(criterion="entropy",
...: max_depth=max_depth)
...: scores = cross_val_score(duku, nanas_att, nanas_pass, cv=5)
...: print("Max depth: %d, Accuracy: %0.2f (+/- %0.2f)" % (max_depth,
...: scores.mean(), scores.std() * 2))
Max depth: 1, Accuracy: 0.58 (+/- 0.01)
Max depth: 2, Accuracy: 0.59 (+/- 0.03)
Max depth: 3, Accuracy: 0.61 (+/- 0.11)
Max depth: 4, Accuracy: 0.60 (+/- 0.10)
Max depth: 5, Accuracy: 0.59 (+/- 0.08)
Max depth: 6, Accuracy: 0.60 (+/- 0.10)
Max depth: 7, Accuracy: 0.60 (+/- 0.08)
Max depth: 8, Accuracy: 0.55 (+/- 0.06)
Max depth: 9, Accuracy: 0.56 (+/- 0.04)
Max depth: 10, Accuracy: 0.57 (+/- 0.06)
Max depth: 11, Accuracy: 0.59 (+/- 0.06)
Max depth: 12, Accuracy: 0.57 (+/- 0.09)
Max depth: 13, Accuracy: 0.54 (+/- 0.13)
Max depth: 14, Accuracy: 0.57 (+/- 0.07)
Max depth: 15, Accuracy: 0.58 (+/- 0.09)
Max depth: 16, Accuracy: 0.55 (+/- 0.12)
Max depth: 17, Accuracy: 0.55 (+/- 0.09)
Max depth: 18, Accuracy: 0.56 (+/- 0.06)
Max depth: 19, Accuracy: 0.58 (+/- 0.06)

In [17]:
```

Figure 2.62: for max depth in range

```

In [88]: depth_acc = np.empty((19,3), float)
...: i = 0
...: for max_depth in range(1, 20):
...: duku = tree.DecisionTreeClassifier(criterion="entropy", max_depth=max_depth)
...: scores = cross_val_score(duku, nanas_att, nanas_pass, cv=5)
...: depth_acc[i,0] = max_depth
...: depth_acc[i,1] = scores.mean()
...: depth_acc[i,2] = scores.std() * 2
...: i += 1
...:
...: Out[88]:
array([[1.00000000e+00, 5.79751704e-01, 6.30768599e-03],
[2.00000000e+00, 5.79814995e-01, 4.59786402e-02],
[3.00000000e+00, 5.72091853e-01, 3.22002693e-02],
[4.00000000e+00, 5.79881532e-01, 8.26691511e-02],
[5.00000000e+00, 5.87665531e-01, 1.00629211e-01],
[6.00000000e+00, 5.7506491e-01, 1.06656313e-01],
[7.00000000e+00, 5.80039760e-01, 8.19527208e-02],
[8.00000000e+00, 5.90179788e-01, 9.29887851e-02],
[9.00000000e+00, 5.90037326e-01, 9.46882094e-02],
[1.00000000e+01, 5.87568160e-01, 1.13990705e-01],
[1.10000000e+01, 5.77473223e-01, 1.11095161e-01],
[1.20000000e+01, 5.77507303e-01, 8.03612112e-02],
[1.30000000e+01, 6.02601428e-01, 4.38647024e-02],
[1.40000000e+01, 5.95037326e-01, 5.40261640e-02],
[1.50000000e+01, 5.77252515e-01, 2.74778000e-02],

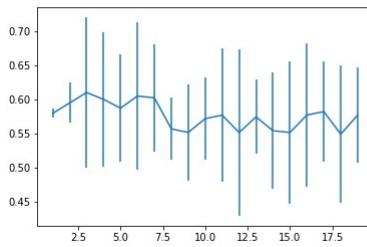
```

Figure 2.63: depth acc

```

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

```



In [19]:

Figure 2.64: matplotlib

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

## 12. matplotlib

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

### 2.2.16 Penanganan Error

#### 1. Skrinsut error

```

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

In [4]:
```

Figure 2.65: error

```

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

Figure 2.66: penyelesaian

## 2. Tuliskan kode eror dan jenis errornya

Kode dan jenis error yang didapatkan adalah :

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

## 3. Solusi pemecahan masalah error tersebut

Berikut adalah solusi dari permasalahan pada no. 1

## 2.3 Rahmi Roza/1164085

### 2.3.1 Teori

Penyelesaian Tugas Harian 3 ( No. 1-7 )

#### 1. Binary Classification Dan Ilustrasi Gambarnya

- Pengertian Binary Classification / Klasifikasi Biner:

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

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

Supervised learning adalah sebuah pendekatan dimana sudah terdapat data yang dilatih, dan terdapat variable yang ditargetkan sehingga

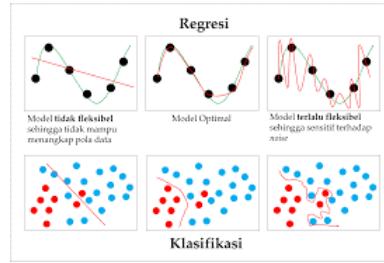


Figure 2.67: binary classification

tujuan dari pendekatan ini adalah mengelompokan suatu data ke data yang sudah ada.

- Ilustrasi Gambar Supervised Learning :

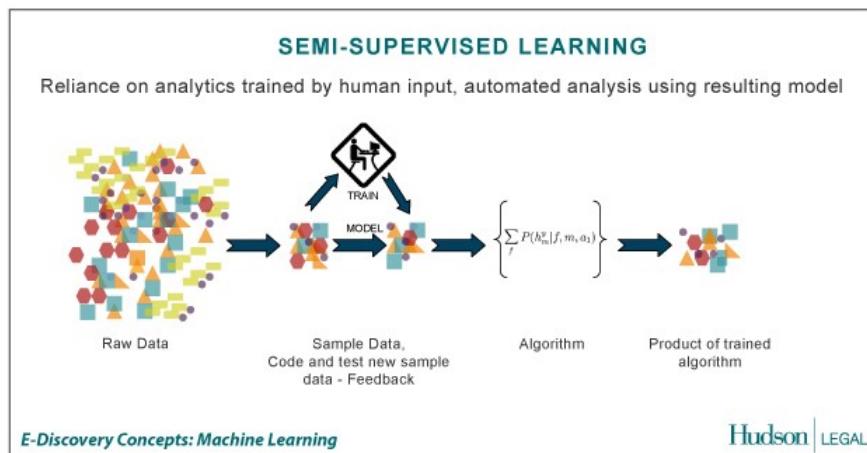


Figure 2.68: supervised

- Pengertian Unsupervised Learning :

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

- Ilustrasi Gambar Unsupervised Learning :

- Pengertian Clustering :

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

## 2. Evaluasi, Akurasi Dan Ilustrasi Gambar

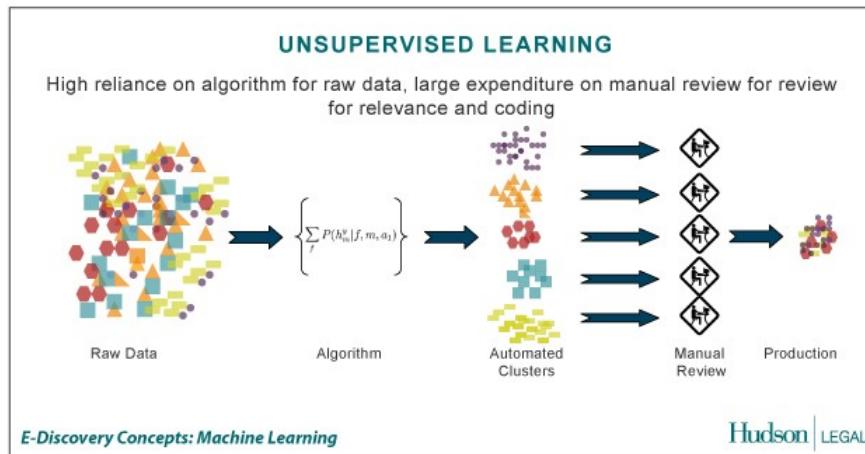


Figure 2.69: unsupervised

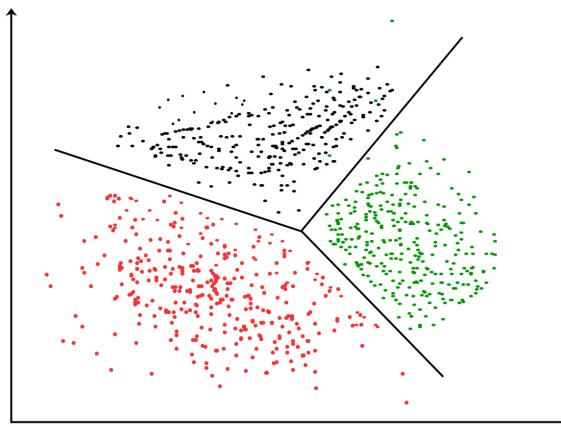


Figure 2.70: clustering

- Pengertian Evaluasi

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

- Pengertian Akurasi

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

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

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

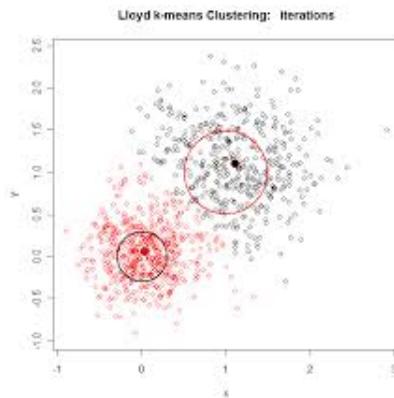


Figure 2.71: Evaluasi

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

Figure 2.72: Akurasi

data tersebut.

Contoh penggabungan Akurasi Dan Evaluasi

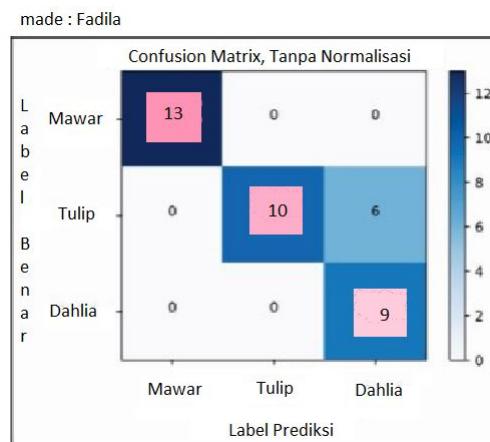


Figure 2.73: Contoh Evaluasi Dan Akurasi Secara Bersamaan

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

- Pengertian Confusion Matrix

Confusion matrix adalah suatu metode yang biasanya digunakan untuk melakukan perhitungan akurasi pada konsep data mining. Rumus ini

melakukan perhitungan dengan 4 keluaran, yaitu: recall, precision, accuracy dan error rate.

- Pembacaan Confusion Matrix
    - (a) Apabila hasil prediksi negatif dan data sebenarnya merupakan negatif.
    - (b) Apabila hasil prediksi positif sedangkan nilai sebenarnya merupakan negatif.
    - (c) Apabila hasil prediksi negatif sedangkan nilai sebenarnya merupakan positif.
    - (d) Apabila hasil prediksi positif dan nilai sebenarnya merupakan positif.
  - Pembuatan Confusion Matrix
    - (a) Menentukan 4 proses klasifikasi yang akan digunakan dalam confusion matrix.
    - (b) 4 Istilah tersebut ada True Positive ( TP ), True Negative ( TN ), False Positive ( FP ) dan False Negative ( FN ).
    - (c) Kelompokkan klasifikasi tersebut bisa menggunakan klasifikasi biner
    - (d) Akan menghasilkan keluaran berupa 2 Kelas ( Positif dan Negatif ) dan penentuan TP, FP ( 1 klasifikasi positif ), FN dan TN ( 1 klasifikasi negatif ).
    - (e) Contoh dasarnya nampak seperti langkah diatas
    - (f) Istilahnya daat didefinisikan dengan objek lain namu dengan alur yang sama ( sesuai rumus baik klasifikasi dll ).
  - Ilustrasi Gambar
4. Cara Kerja K-Fold Classification Dan Ilustrasi Gambar
- (a) Pertama-tama untuk total instance dibagi menjadi N bagian.
  - (b) Fold ke-1 ( atau pertama ) adalah ketika bagian ke-1 menjadi data uji ( testing data ) dan sisanya menjadi data latih ( training data ).
  - (c) Hitung akurasi ( berdasarkan porsi data tersebut. Persamaanya sebagai berikut :
    - (sigma) data klasifikasi
    - (sigma) total data uji
    - x 100 persen

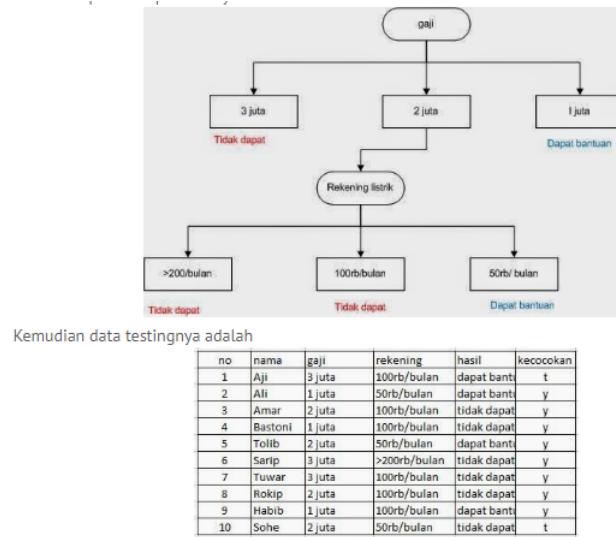


Figure 2.74: confusion matrix

- (d) Fold ke-2 ( kedua ) adalah ketika bagian ke-2 menjadi data uji (testing data) dan sisanya menjadi data latih (training data).
- (e) Kemudian dihitunglah akurasi berdasarkan porsi data yang telah ditentukan
- (f) Demikian seterusnya hingga mencapai fold ke-K. Hitung rata-rata akurasi dari K buah akurasi di atas. Rata-rata akurasi ini menjadi akurasi final atau akhir.

- Ilustrasi Gambar
- Decision Tree Dan Ilustrasi Gambar

- Pengertian Decision Tree

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

**Dataset:**

|                        |       |
|------------------------|-------|
| “                      | ----- |
| K1   K2   K3   K4   K5 |       |
| -----                  |       |

**Data Eksperimen:**

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

Figure 2.75: k-fold classification 1

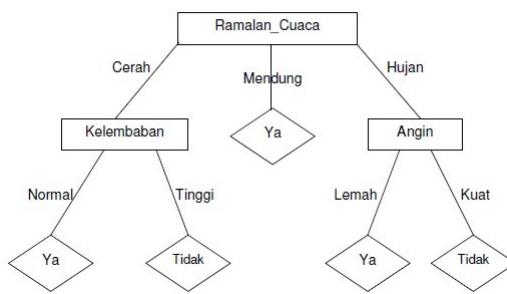


Figure 2.76: decision tree

- Ilustrasi Gambar
- Information Gain Dan Entropi
- Pengertian Information Gain

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

- Ilustrasi Gambar
- Pengertian Entropi

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

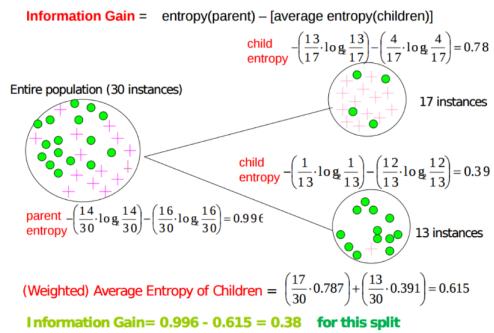


Figure 2.77: informaion gain 1

besar.

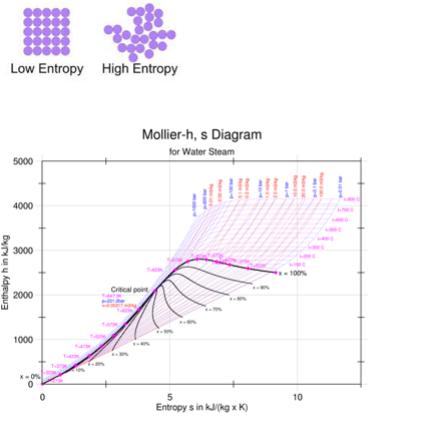


Figure 2.78: entropi

### 2.3.2 Scikit-learn

Penyelesaian Tugas Harian 4 ( No. 1-12 )

- Pembahasan Codingan Dan Hasilnya

(a) Gambar Pertama :

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

– Hasil Gambar Pertama :

(b) Gambar Kedua :

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

Figure 2.79: Gambar pertama

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

– Hasil Gambar Kedua :

```
In [41]: solok['pass'] = solok.apply(lambda row: 1 if (row['G1']+row['G2']+row['G3']) >
35 else 0, axis=1)
...: solok = solok.drop(['G1', 'G2', 'G3'], axis=1)
...: solok.head()
Out[41]:
 school sex age address famsize ... Dalc Walc health absences pass
0 GP F 18 U GT3 ... 1 1 3 6 0
1 GP F 17 U GT3 ... 1 1 3 4 0
2 GP F 15 U LE3 ... 2 3 3 10 0
3 GP F 15 U GT3 ... 1 1 5 2 1
4 GP F 16 U GT3 ... 1 2 5 4 0
[5 rows x 31 columns]
```

Figure 2.80: Gambar kedua

(c) Gambar Ketiga :

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

– Hasil Gambar Ketiga :

```
In [42]: solok = pd.get_dummies(solok, columns=['sex', 'school', 'address', 'famsize', 'Pstatus', 'Mjob',
'Fjob',
...:
...:
...: solok.head())
Out[42]:
 age Medu Fedu ... internet_yes romantic_no romantic_yes
0 18 4 4 ... 0 1 0
1 17 1 1 ... 1 1 0
2 15 1 1 ... 1 1 0
3 15 4 2 ... 1 0 1
4 16 3 3 ... 0 1 0
[5 rows x 57 columns]
```

Figure 2.81: Gambar Ketiga

(d) Gambar Keempat :

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

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

Figure 2.82: Gambar Keempat

– Hasil Gambar Keempat :

(e) Gambar Kelima :

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

– Hasil Gambar Kelima :

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

Figure 2.83: Gambar Kelima

(f) Gambar Keenam :

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

– Hasil Gambar Keenam :

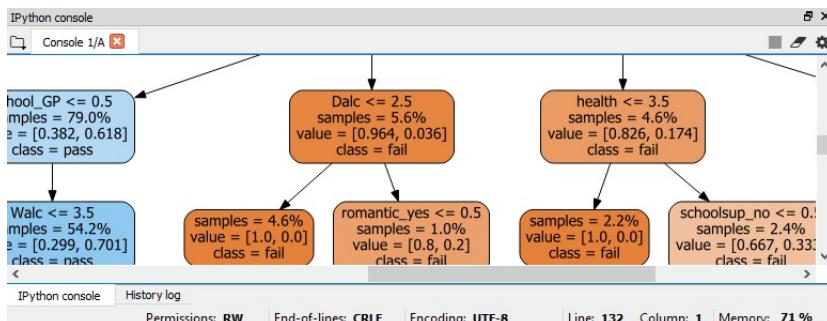


Figure 2.84: Gambar Keenam

(g) Gambar Ketujuh :

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

– Hasil Gambar Ketujuh :

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

Figure 2.85: Gambar Ketujuh

(h) Gambar Kedelapan :

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

– Hasil Gambar Kedelapan :

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

Figure 2.86: Gambar Kedelapan

(i) Gambar Kesembilan:

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

– Hasil Gambar Kesembilan :

```
In [49]: from sklearn.model_selection import cross_val_score
...: scores = cross_val_score(solo, solok_att, solok_pass, cv=5)
...: # show average score and +/- two standard deviations away (covering 95% of scores)
...: print("Accuracy: %.2f (+/- %.2f)" % (scores.mean(), scores.std() * 2))
Accuracy: 0.53 (+/- 0.11)
```

Figure 2.87: Gambar Kesembilan

(j) Gambar Kesepuluh :

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

```

In [50]: for max_depth in range(1, 20):
...: solo = tree.DecisionTreeClassifier(criterion="entropy", max_depth=max_depth)
...: scores = cross_val_score(solo, solok_att, solok_pass, cv=5)
...: print("Max depth: %d, Accuracy: %0.2f (+/- %0.2f)" % (max_depth, scores.mean(), scores.std()
2))
Max depth: 1, Accuracy: 0.58 (+/- 0.01)
Max depth: 2, Accuracy: 0.59 (+/- 0.07)
Max depth: 3, Accuracy: 0.59 (+/- 0.14)
Max depth: 4, Accuracy: 0.54 (+/- 0.02)
Max depth: 5, Accuracy: 0.53 (+/- 0.11)
Max depth: 6, Accuracy: 0.57 (+/- 0.11)
Max depth: 7, Accuracy: 0.56 (+/- 0.11)
Max depth: 8, Accuracy: 0.55 (+/- 0.15)
Max depth: 9, Accuracy: 0.57 (+/- 0.09)
Max depth: 10, Accuracy: 0.56 (+/- 0.11)
Max depth: 11, Accuracy: 0.57 (+/- 0.15)
Max depth: 12, Accuracy: 0.58 (+/- 0.09)
Max depth: 13, Accuracy: 0.57 (+/- 0.09)
Max depth: 14, Accuracy: 0.55 (+/- 0.11)
Max depth: 15, Accuracy: 0.57 (+/- 0.09)
Max depth: 16, Accuracy: 0.57 (+/- 0.09)
Max depth: 17, Accuracy: 0.54 (+/- 0.14)
Max depth: 18, Accuracy: 0.55 (+/- 0.09)
Max depth: 19, Accuracy: 0.55 (+/- 0.09)

```

Figure 2.88: Gambar Kesepuluh

– Hasil Gambar Kesepuluh :

(k) Gambar Kesebelas :

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

– Hasil Gambar Kesebelas :

```

In [51]: depth_acc = np.empty((19,3), float)
...: i = 0
...: for max_depth in range(1, 20):
...: solo = tree.DecisionTreeClassifier(criterion="entropy", max_depth=max_depth)
...: scores = cross_val_score(solo, solok_att, solok_pass, cv=5)
...: depth_acc[i,0] = max_depth
...: depth_acc[i,1] = scores.mean()
...: depth_acc[i,2] = scores.std() * 2
...: i += 1
...:
...: depth_acc
Out[51]:
array([[1.00000000e+00, 5.79751704e-01, 6.30768599e-03],
[2.00000000e+00, 5.90073028e-01, 7.19635495e-02],
[3.00000000e+00, 5.87446446e-01, 1.37192167e-01],
[4.00000000e+00, 5.44337877e-01, 2.41201836e-02],
[5.00000000e+00, 5.29500162e-01, 1.05510772e-01],
[6.00000000e+00, 5.85208422e-01, 1.22237161e-01],
[7.00000000e+00, 5.64788218e-01, 9.32673367e-02],
[8.00000000e+00, 5.37129996e-01, 1.21187315e-01],
[9.00000000e+00, 5.69980526e-01, 1.27258587e-01],
[1.00000000e+01, 5.52448880e-01, 1.49180519e-01],
[1.10000000e+01, 5.67672822e-01, 1.59022626e-01],
[1.20000000e+01, 5.82702845e-01, 1.36814886e-01],
[1.30000000e+01, 5.49758195e-01, 1.52087945e-01],
[1.40000000e+01, 5.67417235e-01, 1.37785627e-01],
[1.50000000e+01, 5.64978903e-01, 1.11136389e-01]]).

```

Figure 2.89: Gambar Kesebelas

(l) Gambar Keduabelas :

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

– Hasil Gambar Keduabelas :

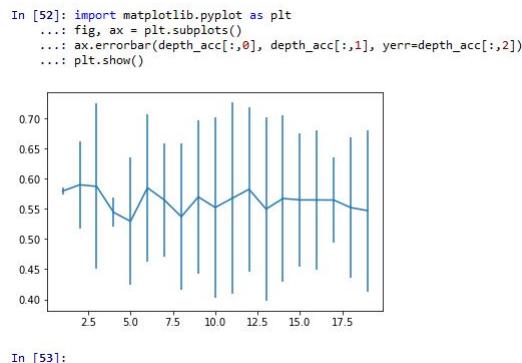


Figure 2.90: Gambar Keduabelas

### 2.3.3 Penanganan Error

(a) Skrinsut Error

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

Figure 2.91: Gambar Ketigabelas

(b) Kode Error dan Jenis Errornya

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

Jenis Error: Pada Grafik

(c) Penanganan

Melakukan install ulang pada graphiz

# Chapter 3

## Methods

### 3.1 Lusia Violita Aprilian/1164080

#### 3.1.1 Teori

##### 1. Random Forest

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

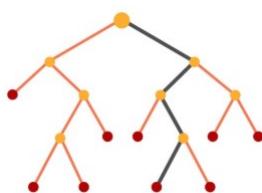


Figure 3.1: Random Forest

##### 2. Membaca Dataset

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

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

Figure 3.2: Kode Python

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

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

Figure 3.3: Pesan Window Console

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

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

Figure 3.4: Dataset

- (e) Lalu klik dataset cell, maka akan muncul seperti berikut :
- (f) Seperti yang terlihat pada gambar tersebut dataset ini memiliki Kolom Country, Age, dan Salary sebagai independent variable-nya dan kolom Purchased sebagai dependent variable-nya.
- (g) Selanjutnya buat 2 matrix of features yang berisi values dari independent variable dan dependent variable.
- (h) Lalu tuliskan perintah berikut :
- (i) Perintah yang telah dibuat di atas akan membuat sebuah global environment baru dan muncul dataset.
- (j) Klikdataset tersebut maka muncul tabel berisi dataset.

### 3. Cross Validation

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

### 4. Menjelaskan Arti Score

| Index | Country | Age | Salary | Purchased |
|-------|---------|-----|--------|-----------|
| 0     | France  | 44  | 72000  | No        |
| 1     | Spain   | 27  | 48000  | Yes       |
| 2     | Germany | 30  | 54000  | No        |
| 3     | Spain   | 38  | 61000  | No        |
| 4     | Germany | 40  | nan    | Yes       |
| 5     | France  | 35  | 58000  | Yes       |
| 6     | Spain   | nan | 52000  | No        |
| 7     | France  | 46  | 79000  | Yes       |
| 8     | Germany | 50  | 83000  | No        |
| 9     | France  | 37  | 67000  | Yes       |

Figure 3.5: Hasil Dataset Cell

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

Figure 3.6: Perintah

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

## 5. Cara Membaca Confusion Matriks

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

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

Figure 3.7: Tabel Confusion Matriks

- Dalam tabel tersebut, baris apple dan orange mengacu pada kasus di mana objek tersebut sebenarnya sebuah apel atau sebuah jeruk. Kolom merujuk pada prediksi yang dibuat oleh model. Kita melihat bahwa dalam contoh ada 20 apel yang diprediksi dengan benar, sementara ada 5 apel yang salah diidentifikasi sebagai jeruk. Idealnya, confusion matriks harus memiliki semua nilai nol, kecuali untuk diagonal. Di sini kita dapat menghitung

akurasi dengan menambahkan angka secara diagonal, sehingga ini semua adalah contoh yang diklasifikasikan dengan benar, dan membagi jumlah tersebut dengan jumlah semua angka dalam matriks:

$$Accuracy = (20 + 22) / (20 + 5 + 3 + 22) = 84$$

Figure 3.8: Rumus Confusion Matriks

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

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

Figure 3.9: Confusion Matriks

## 6. Voting Pada Random Forest

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

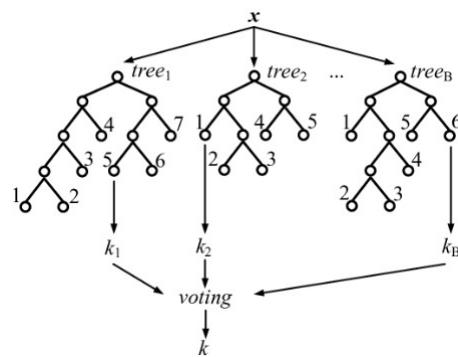


Figure 3.10: Voting Pada Random Forest

### 3.1.2 Praktek

#### 1. Aplikasi Sederhana Menggunakan Pandas

Penjelasan kodingan :

- (a) Memanggil library.

```
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 3.11: Aplikasi Pandas

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

Sehingga menghasilkan :

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

Figure 3.12: Hasil Pandas

## 2. Aplikasi Sederhana Menggunakan Numpy

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

Figure 3.13: Aplikasi Numpy

Penjelasan kodingan :

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

Sehingga menghasilkan :

## 3. Aplikasi Sederhana Menggunakan Matplotlib

Penjelasan kodingan :

- (a) Memanggil library
- (b) Membuat variable yang berisi bahasa pemrograman
- (c) Membuat variable yang berisi popularitas
- (d) Membuat variable untuk menentukan warna
- (e) Membuat variable untuk explode
- (f) Membuat diagram pie atau yang berbentuk lingkaran

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

Figure 3.14: Hasil Numpy

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

Figure 3.15: Aplikasi Matplotlib

(g) Membuat garis koordinat

(h) Menampilkan hasil

Sehingga menghasilkan :

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

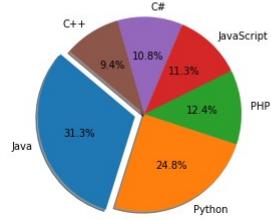


Figure 3.16: Hasil Matplotlib

#### 4. Program Klasifikasi Random Fores

- Yang pertama dataset akan dibaca.
- Selanjutnya sebagian data awal akan dilihat dengan menggunakan listing.
- Selanjutnya jumlah data dilihat dengan menggunakan listing.
- Lalu atribut diubah menjadi kolom dengan menggunakan perintah pivot.
- Selanjutnya atribut yang telah diubah, sebagian data awalnya akan dilihat dengan menggunakan listing kembali.

```

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

```

Figure 3.17: Membaca Data File

```

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

```

Figure 3.18: Melihat Data Sebagian

- Selanjutnya atribut yang telah diubah, jumlah data dilihat dengan menggunakan listing kembali.
- Lalu mengelompokkan burung kedalam spesies yang sama dengan dua kolom imgid dan label.
- Lalu melakukan pivot dimana imgid menjadi index.
- Selanjutnya imgid, sebagian data awalnya akan dilihat dengan menggunakan listing untuk mengecek data.
- Selanjutnya imgid, jumlah data dilihat dengan menggunakan listing untuk mengecek data.
- Lalu melakukan join karena isi datanya adalah sama di antara dua data. Sehingga mendapatkan data ciri labelnya sehingga bisa dikategorikan.
- Kemudian label yang didepan di drop dan berikan label pada data yang telah dilakukan join dengan perintah listing.
- Lalu cek kembali isinya dengan perintah listing.
- Kemudian data dibagi menjadi dua bagian, dimana 8000 row pertama merupakan data training dan sisanya adalah data testing.
- Kelas random forest selanjutnya dipanggil dengan RandomForestClassifier, dengan banyak kolom yang telah ditentukan oleh max feature.
- Kemudian untuk membangun random forest dilakukan perintah fitting dengan maksimum fitur sebanyak 50.

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

Figure 3.19: Melihat Jumlah Data

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

Figure 3.20: Mengubah menjadi kolom

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

## 5. 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.

## 6. Program Klasifikasi SVM dan Decision Tree

### (a) Program Decision Tree

Mengklasifikasikan dataset yang sama menggunakan decision tree.

### (b) Program Klasifikasi SVM

Mengklasifikasikan dataset yang sama menggunakan SVM.

## 7. Program Cross Validation

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

## 8. Program Pengamatan Komponen Informasi

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

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

Figure 3.21: Lihat sebagian data awal

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

Figure 3.22: Melihat jumlah data

### 3.1.3 Penanganan Error

1. Skrinsut Error
2. Tuliskan kode eror dan jenis errornya

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

Jenis Error = File not found

3. Solusi Pemecahan Masalah Error

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

Sehingga didapat hasil seperti berikut :

## 3.2 Rahmi Roza/1164085

### 3.2.1 Teori

Penyelesaian Tugas Harian 5 ( No. 1-6 )

1. Random Forest Dan Ilustrasi Gambarnya

- Pengertian Random Forest:

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

```

In [17]: imglabels = pd.read_csv("CUB_200_2011/image_class_labels.txt",
...: sep=' ', header=None, names=['imgid', 'label'])
...:
...: imglabels = imglabels.set_index('imgid')
...:
...: # description from dataset README:
...: #
...: # The ground truth class labels (bird species labels) for each image are contained
...: # in the file image_class_labels.txt, with each line corresponding to one image:
...: #
...: # <image_id> <class_id>
...: #
...: # where <image_id> and <class_id> correspond to the IDs in images.txt and classes.txt,
...: # respectively.

```

Figure 3.23: Mengelompokkan burung

```

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

```

Figure 3.24: Melalukan pivot

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 foresttidak dilakukan pemangkasan yang merupakan sebuah metode untuk mengurangi kompleksitas ruang.

- Ilustrasi Gambar Random Forest :

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

- Cara Membaca Dataset :

- (a) Buka Anaconda Navigator lalu jalankan Spyder, kemudian import libraries yang dibutuhkan.
- (b) Masukkan kode python untuk membaca file csv, lalu jalankan.
- (c) Maka pada window console akan menampilkan pesan berikut :
- (d) Dari explorer dapat terlihat dataset yang terimport.
- (e) Lalu klik dataset cell, maka akan muncul seperti berikut :
- (f) Seperti yang terlihat pada gambar tersebut dataset ini memiliki Kolom Country, Age, dan Salary sebagai independent variable-nya dan kolom Purchased sebagai dependent variable-nya.
- (g) Selanjutnya buat 2 matrix of features yang berisi values dari independent variable dan dependent variable.
- (h) Lalu tuliskan perintah berikut :

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

Figure 3.25: Melihat data awal imgid

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

Figure 3.26: Melihat jumlah data imgid

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

### 3. Cross Validation

- Pengertian Cross Validation :

Cross Validation atau bisa disebut estimasi rotasi ,adalah sebuah teknik validasi model untuk menilai bagaimana hasil statistik analisis akan menggeneralisasi kumpulan data independen. Teknik ini utamanya digunakan untuk melakukan prediksi model dan memperkirakan seberapa akurat sebuah model prediktif ketika dijalankan dalam praktiknya.

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

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

### 4. Penjelasan / Maksud Dari Score pada :

- Random forest ( 44% )

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

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

Figure 3.27: Data ciri label dari join

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

Figure 3.28: Mengubah menjadi kolom

- Decision Tree ( 27% )

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

- SVM ( 29% )

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

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

## 5. Confusion Matrix Dan Ilustrasinya

- Cara Membaca Confusion Matrix :

Perhitungan confusion matrix adalah sebagai berikut, akan saya beri contoh sederhana yaitu pengambilan keputusan untuk mendapatkan bantuan beasiswa. Saya menggunakan dua atribut, yaitu rekening listrik dan gaji. Yang pertama kita lakukan yaitu mencari 4 nilai yaitu a,b,c, dan d:

$$a = 4$$

$$b = 1$$

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

Figure 3.29: Melihat isi data frame

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

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

Figure 3.30: Membagi data

$$c = 1$$

$$d = 2$$

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

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

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

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

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

Ilustrasi Confusion Matrix :

## 6. Voting Random Forest Dan Ilustrasi Gambarnya.

- Pengertian Voting pada Random Forest :

Metode ensemble dapat mencapai akurasi tinggi dengan membangun beberapa pengklasifikasi dan menjalankan masing-masing secara mandiri. Ketika classifier membuat sebuah keputusan, kamu dapat memanfaatkan yang terbaik keputusan umum dan rata-rata. Jika kita menggunakan metode yang paling umum, itu disebut voting.

- Ilustrasi Gambar Voting Random Forest :

### 3.2.2 Praktek

Penyelesaian Tugas Harian 6 ( No. 1-8 )

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

Figure 3.31: Kelas Random Forest

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

Figure 3.32: Membangun Random forest

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

- Pandas:

- Penjelasan Code Baris 1 : Mengimport library pandas dari python dengan inisiasi pd.
- Penjelasan Code Baris 2 : Variabel s1 mendefinisikan data menggunakan pandas series.
- Penjelasan Code Baris 3 : Mencetak output
- Penjelasan Code Baris 4 : Menampilkan data s1
- Penjelasan Code Baris 5 : Mencetak output
- Penjelasan Code Baris 6 : Mengubah data karakter menjadi numerik
- Penjelasan Code Baris 7 : Mempilkan data s2

- Hasil:

- Aplikasi Sederhana Numpy dan Penjelasan Code ( Perbaris )

- Code Numpy:

- \* Penjelasan Code Baris 1 : Import library numpy
    - \* Penjelasan Code Baris 2 : Menampilkan versi dari numpy.
    - \* Penjelasan Code Baris 3 : Menampilkan konfigurasi dari numpy.

- Hasil:

- Aplikasi Sederhana Matplotlib dan Penjelasan Code ( Perbaris )

- \* Code Matplotlib:

- Penjelasan Code Baris 1 : Mengimport library matplotlib dari python dengan inisiasi plt.
    - Penjelasan Code Baris 2 : Variabel X
    - Penjelasan Code Baris 3 : Variabel Y
    - Penjelasan Code Baris 4 : Menampilkan nilai dari X
    - Penjelasan Code Baris 5 : Menampilkan value dari X

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

Figure 3.33: Melihat hasil

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

Figure 3.34: Lihat hasil score

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

\* Hasil:

– Program Aplikasi Random Forest dan Penjelasan Keluarannya :

\* Code Random Forest 1 :

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

\* Code Random Forest 2 :

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

\* Code Random Forest 3 :

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

\* Code Random Forest 4 :

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

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

Figure 3.35: Memetakan ke confusion matrix

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

Figure 3.36: Melihat hasil

\* Code Random Forest 5 :

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

\* Code Random Forest 6 :

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

\* Code Random Forest 7 :

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

\* Code Random Forest 8 :

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

\* Code Random Forest 9 :

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

\* Code Random Forest 10 :

```

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

```

Figure 3.37: Melakukan Plot

```

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

```

Figure 3.38: Plotting nama data

- Penjelasan : Pada gambar diatas dikarenakan isinya sama, maka bisa melakukan join antara dua data yang diesekusi ( yaitu ada imgatt2 dan imglables ), sehingga pada hasilnya akan didapatkan data ciri dan data jawaban atau labelnya sehingga bisa dikategorikan/dikelompokkan sebagai supervised learning. Jadi perintah untuk menggabungkan kedua data, kemudian dilakukan pemisahan antara data set untuk training dan test pada dataset yang dieksekusi.
- \* Code Random Forest 11 :
  - Penjelasan :Pada gambar di atas menghasilkan pemisahan dan pemilihan tabel ( memisahkan dan memilih tabel ).
- \* Code Random Forest 12 :
  - Penjelasan : Pada gambar di atas menunjukkan hasil dari variabel dtatthead. Dimana data nya dapat dilihat pada gambar

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

Figure 3.39: Melakukan perintah plot

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

In [36]: Out[35]: 0.270855332629355850ut[36]: 0.27560718057022177
```

Figure 3.40: Klasifikasi menggunakan decision tree

diatas. Dan dataset nya terdiri dari 5 baris dan 312 kolom.

\* Code Random Forest 13 :

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

\* Code Random Forest 14 :

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

\* Code Random Forest 15 :

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

\* Code Random Forest 16 :

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

\* Code Random Forest 17 :

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

\* Code Random Forest 18 :

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

```

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

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

```

Figure 3.41: Klasifikasi menggunakan SVM

```

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

```

Figure 3.42: Pengecekan cross validation random forest

– Program Aplikasi Confusion Matrix dan Penjelasan Keluarannya :

\* Code Confusion Matrix 1 :

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

\* Code Confusion Matrix 2 :

- Penjelasan : Pada gambar di atas merupakan tampilan dari variabel cm.

\* Code Confusion Matrix 3 :

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

\* Code Confusion Matrix 4 :

- Penjelasan : Pada gambar di atas merupakan kodingan untuk menyesuaikan sumbu dengan nama datanya makanya datset nya di lakukan dengan perintah di atas.

\* Code Confusion Matrix 5 :

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

– Program Klasifikasi SVM dan Decision Tree Beserta Penjelasan Keluarannya :

\* Code SVM :

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

```
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 3.43: Pengecekan cross validation decision tree

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

Figure 3.44: Pengecekan cross validation SVM

- \* Code Decision Tree :
  - Penjelasan : Pada gambar di atas merupakan cara untuk mencoba klasifikasi dengan decision tree dengan dataset yang sama.
- Program Cross Validation dan Penjelasan Keluarannya :
  - \* Code Cross Validation 1 :
    - Penjelasan : Pada gambar di atas merupakan Hasil dari cross validation random forest.
  - \* Code Cross Validation 2 :
    - Penjelasan : Pada gambar di atas merupakan hasil dari cross validation Decision tree.
  - \* Code Cross Validation 3 :
    - Penjelasan : Pada gambar di atas merupakan hasil dari cross validation SVM.
- Program Pengamatan Komponen Informasi dan Penjelasan Keluarannya :
  - \* Code Pengamatan Komponen Informasi 1 :
    - Penjelasan : Pada gambar di atas menunjukkan cara untuk mengetahui berapa banyak tree yang dibuat, berapa banyak atribut yang dipakai dan informasi lainnya menggunakan kode.

```

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

```

Figure 3.45: Pengamatan Komponen

```

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

```

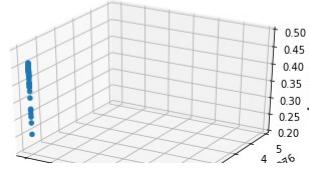


Figure 3.46: Plot informasi

#### \* Code Pengamatan Komponen Informasi 2 :

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

## 2. Penanganan Error

### • Skrinsut Error

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

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

Figure 3.47: Skrinsut Error

```
import pandas as pd
some lines have too many fields (?), so skip bad lines
imgatt = pd.read_csv("CUB_200_2011/attributes/image_attribute_labels.txt",
 sep='\t', header=None, error_bad_lines=False, warn_bad_lines=False,
 usecols=[0,1,2], names=['imgid', 'attid', 'present'])
```

Figure 3.48: Penyelesaian

### 3.3 Fadila/1164072

#### 3.3.1 Teori

Penyelesaian Tugas Harian 5 ( No. 1-6 )

##### 1. Random Forest Dan Ilustrasi Gambarnya

- Pengertian Random Forest:

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

- Ilustrasi Gambar Random Forest :

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

- Cara Membaca Dataset :

Membaca Dataset

- (a) Langkah-langkah cara membaca dataset :

- Pertama-tama kita harus membuka Aplikasi yang digunakan untuk membuka datasetnya
  - Yang saya gunakan ialah spyder dari Anaconda Navigator
  - Selanjutnya import libraries seperti numpy, pandas, matplotlib dll ( sesuai kebutuhan ).
  - Silahkan untuk memasukkan kode python yang digunakan untuk membaca file csv yang berada pada dataset
- ```
dataset = pd.read_csv('Data.csv')
```

```

In [9]: import pandas as pd
...
...: # some lines have too many fields (?), so skip bad lines
...: imgatt = pd.read_csv("CUB_2011/attributes/image_attribute_labels.txt",
...:                     sep='\t', header=None, error_bad_lines=False,
warn_bad_lines=False,
...:                     usecols=[0,1,2], names=['imgid', 'attid', 'present'])
...
...: # description from dataset README:
...
...: # The set of attribute labels as perceived by MTurkers for each image
...: # is contained in the file attributes/image_attribute_labels.txt, with
...: # each line corresponding to one image/attribute/worker triplet:
...
...: #
...: # <image_id> <attribute_id> <is_present> <certainty_id> <time>
...
...: #
...: # where <image_id>, <attribute_id>, <certainty_id> correspond to the IDs
...: # in images.txt, attributes/attributes.txt, and attributes/certainties.txt
...: # respectively. <is_present> is 0 or 1 (1 denotes that the attribute is
...: # present). <time> denotes the time spent by the MTurker in seconds.

```

Figure 3.49: Hasil

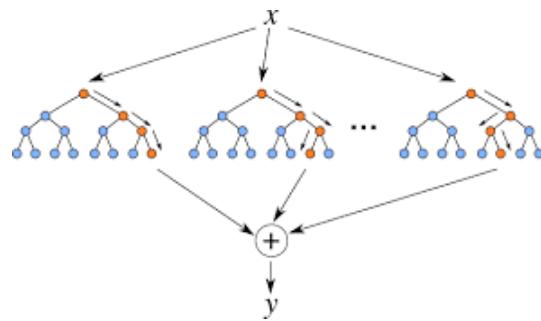


Figure 3.50: Random Forest

- Kemudian jalankan script python tersebut
 - Perhatikan pada bagian ” window consolenya ”, tampilan akan seperti berikut :
 - Dari window ’explorer’ tersebut, kita dapat melihat dataset yang terimport.
 - Akan ada kolom nama, tipe, size dan values
 - Lalu klik dataset cell, maka akan muncul seperti berikut :
 - Selanjutnya seperti yang dapat dilihat pada gambar tersebut untuk dataset ini memiliki Kolom Country, Age, dan Salary sebagai independent variablenya dan kolom Purchased sebagai dependent variablenya.
 - Kemudian pada code buatlah 2 matrix of features yang berisi values / nilai-nilai dari independent variable dan dependent variable yang sesuai.
 - Setelah semuanya selesai, maka silahkan tuliskan perintah berikut :
- ```

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

```

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

Figure 3.51: (b)

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

Figure 3.52: (c)

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

### 3. Cross Validation Dan Ilustrasi Gambar :

- Pengertian Cross Validation :

Cross Validation (validasi silang) biasa disebut dengan estimasi rotasi, atau pengujian out-of-sample dimana merupakan salah satu dari berbagai teknik validasi model yang serupa untuk menilai bagaimana hasil analisis statistik akan digeneralisasikan ke set data independen. Hal ini (terutama) digunakan dalam pengaturan di mana tujuannya adalah prediksi, dan orang ingin memperkirakan seberapa akurat model prediksi akan dilakukan dalam praktek.

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

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

### 4. Penjelasan / Maksud Dari Score pada :

- Random forest ( 44 % )

Maksudnya adalah akurasi. Dimana Hasil dari eksekusi variabel, parameter dll dari perhitungan code dan decision tree terkait akan diproses lagi

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

Figure 3.53: (d)

| Index | Country | Age | Salary | Purchased |
|-------|---------|-----|--------|-----------|
| 0     | France  | 44  | 72000  | No        |
| 1     | Spain   | 27  | 46000  | Yes       |
| 2     | Germany | 38  | 54000  | No        |
| 3     | Spain   | 38  | 61000  | No        |
| 4     | Germany | 48  | nan    | Yes       |
| 5     | France  | 35  | 56000  | Yes       |
| 6     | Spain   | nan | 52000  | No        |
| 7     | France  | 48  | 79000  | Yes       |
| 8     | Germany | 58  | 83000  | No        |
| 9     | France  | 37  | 67000  | Yes       |

Figure 3.54: (e)

untuk menghasilkan nilai akurasi. Dan untuk nilai akurasinya berupa 44 %

- Decision Tree ( 27 % )

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

- SVM ( 29 % )

Untuk nilai 29 % dari SCV merupakan hasil pendekatan jaringan saraf. Jaringan saraf sendiri merupakan komponen jaringan utama dari sistem saraf. Sistem tersebut mengatur dan mengontrol fungsi tubuh dan aktivitas dan terdiri dari dua bagian: (SSP) yang terdiri dari otak dan sumsum tulang belakang, dan percabangan saraf perifer dari sistem saraf tepi (SST) yang terdapat dalam pengolahan dataset terkait.

## 5. Confusion Matrix Dan Ilustrasinya :

- Pengertian Confusion Matrix :

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

- Cara Membaca Confusion Matrix :

Untuk pembacaan dan pemahaman pada confusion matrix dapat memperhatikan penjelasan berikut

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

Figure 3.55: (h)

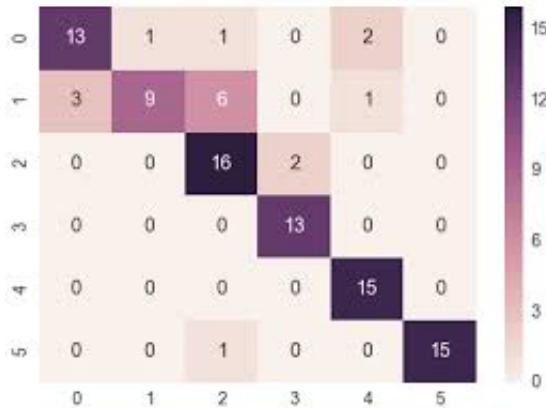


Figure 3.56: Confussion Matrik

- Perhatikan : Apabila hasil prediksi negatif dan data sebenarnya merupakan negatif.
- Perhatikan : Apabila hasil prediksi positif sedangkan nilai sebenarnya merupakan negatif.
- Perhatikan : Apabila hasil prediksi negatif sedangkan nilai sebenarnya merupakan positif.
- Perhatikan : Apabila hasil prediksi positif dan nilai sebenarnya merupakan positif.
- Pembacaan Lanjutan ( lebih jelas ) Untuk Confusion Matrix :
  - (a) Pastikan definisi dari 4 proses klasifikasi yang akan digunakan ada dalam confusion matrix.
  - (b) 4 Istilah tersebut ada TP, TN, FP dan FN
  - (c) Cara bacanya yaitu True Positive ( TP ), True Negative ( TN ), False Positive ( FP ) dan False Negative ( FN ).
  - (d) Ada pendefinisian biner pada klasifikasi confusion matrix
  - (e) Klasifikasi tersebut menghasilkan keluaran berupa 2 Kelas ( Positif dan Negatif ) dan penentuan TP, FP ( 1 klasifikasi positif ) , FN dan TN ( 1 klasifikasi negatif ).
- Ilustrasi Gambar

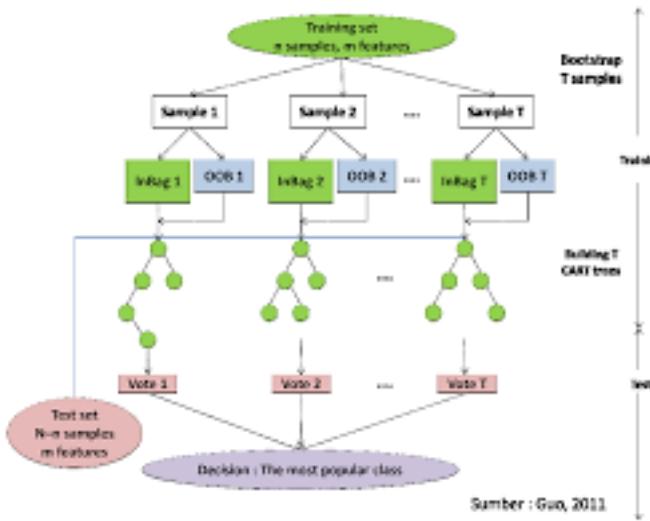


Figure 3.57: Voting Random forest

```

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

```

Figure 3.58: Pandas

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

## 6. Voting Random Forest Dan Ilustrasi Gambarnya.

- Pengertian Voting pada Random Forest :

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

Setelah pohon terbentuk,maka akan dilakukan voting pada setiap kelas dari data sampel. Kemudian, mengkombinasikan vote dari setiap kelas

```

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

```

Figure 3.59: Numpy

kemudian diambil vote yang paling banyak. Dengan menggunakan random forest pada klasifikasi data maka, akan menghasilkan vote yang paling baik.

- Ilustrasi Gambar Voting Random Forest :

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

### 3.3.2 Praktek

Penyelesaian Tugas Harian 6 ( No. 1-8 )

1. Aplikasi Sederhana Pandas dan Penjelasan Code ( Perbaris )

- Code Pandas:

```

import pandas as fadila
np_array = np.array([10, 20, 30, 40, 50])
print("NumPy array:")
print(np_array)

```

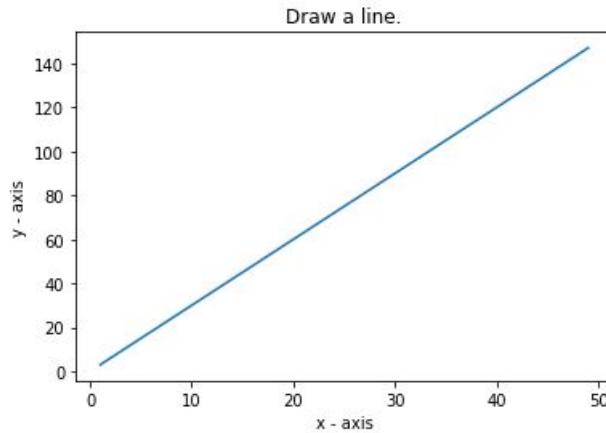


Figure 3.60: Matplotlib

```
In [30]: import pandas as pd
.....
....: # some lines have too many fields (?), so skip bad lines
....: imgatt = pd.read_csv("CUB_200_2011/attributes/image_attribute_labels.txt",
....: sep='\t', header=None, error_bad_lines=False, warn_bad_lines=False,
....: usecols=[0,1,2], names=['imgid', 'attid', 'present'])
.....
....: # description from dataset README:
....: #
....: # The set of attribute labels as perceived by MTurkers for each image
....: # is contained in the file attributes/image_attribute_labels.txt, with
....: # each line corresponding to one image/attribute/worker triplet:
....: #
....: # <image_id> <attribute_id> <is_present> <certainty_id> <time>
....: #
....: # where <image_id>, <attribute_id>, <certainty_id> correspond to the IDs
....: # in images.txt, attributes/attributes.txt, and attributes/certainties.txt
....: # respectively. <is_present> is 0 or 1 (1 denotes that the attribute is
....: # present). <time> denotes the time spent by the MTurker in seconds.
```

Figure 3.61: Gambar1

```
new_series = fadila.Series(np_array)
print("Converted Pandas series dari Fadila:")
print(new_series)
```

- Penjelasan Code Baris 1 :
  - Mengimport library / module pandas sebagai fadila
- Penjelasan Code Baris 2 :
  - variabel np\_array mendefinisikan variabel np.array dimana terdapat beberapa parameter yang akan dieksekusi pada variabel tersebut
- Penjelasan Code Baris 3 :
  - Mendefinisikan perintah print dengan tulisan ” Numpy array ”
- Penjelasan Code Baris 4 :
  - Melakukan perintah print / pencetakan untuk variabel np\_array

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

Figure 3.62: Gambar2

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

Figure 3.63: Gambar3

– Penjelasan Code Baris 5 :

Mendefinisikan variabel baru yaitu new\_series dimana mengeksekusi fadila.series dengan variabel parameternya ialah np\_array

– Penjelasan Code Baris 6 :

Mendefinisikan perintah print dengan tulisan ” Converted pandas series dari fadila ”

– Penjelasan Code Baris 7 :

Melakukan perintah print / pencetakan terhadap variabel new\_series

- Hasil Eksekusi :

2. Aplikasi Sederhana Numpy dan Penjelasan Code ( Perbaris )

- Code Numpy:

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

– Penjelasan Code Baris 1 :

Mengimport library / module numpy sebagai fadila

– Penjelasan Code Baris 2 :

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

– Penjelasan Code Baris 3 :

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

Figure 3.64: Gambar 4

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

1 0 0 0 0 1 0 0 ...
2 0 0 0 0 0 0 0 ...
3 0 0 0 0 1 0 0 ...
4 0 0 0 0 1 0 0 ...
5 0 0 0 0 1 0 0 ...
[5 rows x 312 columns]
```

Figure 3.65: Gambar 5

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

– Penjelasan Code Baris 4 :

Melakukan Perintah print variabel X

– Penjelasan Code Baris 5 :

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

– Penjelasan Code Baris 6 :

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

– Penjelasan Code Baris 7 :

Melakukan perintah print / pencetakan variabel X

- Hasil Eksekusi :

### 3. Aplikasi Sederhana Matplotlib dan Penjelasan Code ( Perbaris )

- Code Matplotlib:

```
import matplotlib.pyplot as fadila
line 1 points
x1 = [10,20,30]
y1 = [20,40,10]
plotting the line 1 points
fadila.plot(x1, y1, label = "line 1")
line 2 points
x2 = [10,20,30]
y2 = [40,10,30]
plotting the line 2 points
```

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

Figure 3.66: 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 3.67: Gambar 7

```
fadila.plot(x2, y2, label = "line 2")
fadila.xlabel('x - axis')
Set the y axis label of the current axis.
fadila.ylabel('y - axis')
Set a title of the current axes.
fadila.title('Dua atau line lebih pada plot yang sama
dengan suitable legends')
show a legend on the plot
fadila.legend()
Display a figure.
fadila.show()
```

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

```
In [39]: imgLabels.isNaive()
Out[39]: False
```

Figure 3.68: Gambar 8

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

Figure 3.69: Gambar 9

Mendefinisikan variabel x1 dengan pengeksekusian 3 parameter yaitu ( 10,20,30 )

– Penjelasan Code Baris 6 :

Mendefinisikan / membuat variabel x2 dengan pengeksekusian 3 parameter yaitu ( 10,20,30 )

– Penjelasan Code Baris 7 :

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

– Penjelasan Code Baris 8 :

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

– Penjelasan Code Baris 9 :

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

– Penjelasan Code Baris 10 :

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

– Penjelasan Code Baris 11 :

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

– Penjelasan Code Baris 12 :

Mendefinisikan perintah fadila.legend tanpa parameter

– Penjelasan Code Baris 13 :

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

- Hasil Eksekusi :

#### 4. Program Aplikasi Random Forest dan Penjelasan Keluarannya :

- Code Random Forest 1 :

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

Figure 3.70: Gambar 10

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

Figure 3.71: Gambar 11

- Penjelasan : Codingan tersebut akan menghasilkan sebuah variabel baru yaitu ”imgatt” ( yang telah dibuat sebelumnya ) dengan tipe dataframe dan berupa matrix 2 dimensi. Terdapat 3 kolom dan 3677856 baris data. Bisa di check isi datanya karena berupa data frame. Codingan tersebut ”Membaca” dan ”Menampilkan” data yang berasal dari file ”image\_attribute\_labels.txt”.
- Code Random Forest 2 :
  - Penjelasan : Hasil dari codingan tersebut berfungsi untuk melihat data awal dari data/dataset yang dibaca dan diproses. Lebih jelasnya hanya untuk melihat isi paling atas dari data dan ditampilkan di console saat codingan dieksekusi.
- Code Random Forest 3 :
  - Penjelasan : Codingan diatas akan menghasilkan dan juga menampilkan dari jumlah data pada dataset yang telah dieksekusi. Jumlah data/ sizenya akan berupa baris dan kolom dari data yang ada.
- Code Random Forest 4 :
  - Penjelasan : Codingan tersebut menghasilkan tampilan variabel baru yaitu ”imgatt2” . Fungsinya yaitu juga untuk merubah atribut menjadi kolom dengan menggunakan pivot layaknya excel. Yang membedakan antara ”imgatt2” dan ”imgatt1” ialah, jumlah dari data yang ditampilkan berbeda.
- Code Random Forest 5 :
  - Penjelasan : Hasil dari codingan tersebut berfungsi untuk melihat data awal dari data/dataset yang dibaca dan diproses pada variabel” imgatt2 ”. Lebih jelasnya hanya untuk melihat isi paling atas dari data dan ditampilkan di console saat codingan dieksekusi.

```

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

```

Figure 3.72: Gambar 12

```

In [45]: df_label.head()
Out[45]:
 label

10779 183
9334 159
10372 177
1554 28
378 8

```

Figure 3.73: Gambar 13

- Code Random Forest 6 :
  - Penjelasan : Codingan diatas akan menghasilkan dan juga menampilkan dari jumlah data pada dataset yang telah dieksekusi pada variabel ” imgatt2 ”. Jumlah data/ sizenya akan berupa baris dan kolom dari data yang ada.
- Code Random Forest 7 :
  - Penjelasan : Hasil dari codingan diatas menampilkan atau menunjukkan load dari jawabannya yang berisi ” apakah burung tersebut ( subjek pada dataset ) termasuk dalam spesies yang mana ?. Kedua kolom yang digunakan adalah imgid dan label, kemudian melakukan pivot yang mana imgid menjadi index yang artinya unik sehubungan dengan dataset yang telah dieksekusi.
- Code Random Forest 8 :
  - Penjelasan : Hasil dari codingan tersebut berfungsi untuk melihat data awal dari data/dataset yang dibaca dan diproses pada variabel” imglabels ”. Lebih jelasnya hanya untuk melihat isi paling atas dari data dan ditampilkan di console saat codingan dieksekusi.
- Code Random Forest 9 :
  - Penjelasan : Codingan diatas akan menghasilkan dan juga menam-

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

Figure 3.74: Gambar 14

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

Figure 3.75: Gambar 15

pilihan dari jumlah data pada dataset yang telah dieksekusi pada variabel ” imglabels ”. Jumlah data/ sizenya akan berupa baris dan kolom dari data yang ada.

- Code Random Forest 10 :

- Penjelasan : Codingan diatas dikarenakan isinya sama, maka bisa melakukan join antara dua data yang diesekusi ( yaitu ada imgatt2 dan imglabels ), sehingga pada hasilnya akan didapatkan data ciri dan data jawaban atau labelnya sehingga bisa dikategorikan/dikelompokkan sebagai supervised learning. Jadi perintah untuk menggabungkan kedua data, kemudian dilakukan pemisahan antara data set untuk training dan test pada dataset yang dieksekusi.

- Code Random Forest 11 :

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

- Code Random Forest 12 :

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

- Code Random Forest 13 :

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

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

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

Figure 3.76: Gambar 16

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

Figure 3.77: Gambar 17

- Code Random Forest 14 :

- Penjelasan : Hasil codingannya menunjukkan pembagian untuk data training dan data testing pada dataset. Dilakukan pembagian data menjadi dua bagian dimana untuk row/baris data pertama sebanyak 8000 dijadikan sebagai data training , kemudian 8000 data sisanya sebagai data testing pada pengeksekusinya.

- Code Random Forest 15 :

- Penjelasan : Hasil dari codingan tersebut yaitu instalasi kelas random forest. Dimana pada pemrosesannya, dilakukan pemanggilan kelas Random Forest Classifier ( Klasifikasi Random Forest ). Pada codingan terdapat max features yang diartikan sebagai berapa banyak kolom pada setiap tree yang dieksekusi pada dataset.

- Code Random Forest 16 :

- Penjelasan : Codingan tersebut menunjukkan fitting random forest dengan dataset training. Dimana pengeksekusinya dilakukan fit untuk membangun random forest yang sudah ditentukan dengan maksimum fitur sebanyak 50 untuk perpohonnya ( dari dataset yang dieksekusi ).

- Code Random Forest 17 :

- Penjelasan : Codingan tersebut menunjukkan hasil dari prediksi dataset yang dieksekusi.

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

Figure 3.78: Gambar 18

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

Figure 3.79: Gambar 19

- Code Random Forest 18 :

- Penjelasan : Codingan tersebut menunjukkan score perolehan dari klasifikasi dataset yang dieksekusi. Terdapat hasil besaran akurasi dari code yang dieksekusi sesuai dengan data yang digunakan.

5. Program Aplikasi Confusion Matrix dan Penjelasan Keluarannya :

- Code Confusion Matrix 1 :

- Penjelasan : Codingan diatas menunjukkan proses dari pembuatan confusion matrix dimana dilakukan pemetaan dan pemetakan confusion matrix terhadap data yang dieksekusi.

- Code Confusion Matrix 2 :

- Penjelasan : Codingan ini menunjukkan hasil dari pembuatan pemetakan data yang dilakukan sebelumnya ( Code confusion matrix 1 ). Hasilnya berupa matrix yang didalamnya terdapat angka dari hasil eksekusi confusion matrix itu sendiri. Memunculkan array dan dtype dari hasil pembuatan confusion matrix.

- Code Confusion Matrix 3 :

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

- Code Confusion Matrix 4 :

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

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

Figure 3.80: Gambar 20

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

Figure 3.81: Gambar 21

- Code Confusion Matrix 5 :

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

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

- Code SVM :

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

- Code Decision Tree :

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

```

In [56]: birds = pd.read_csv("CUB_200_2011/classes.txt",
...: sep='\t', header=None, usecols=[1], names=['birdname'])
...: birds = birds['birdname']
...: birds
Out[56]:
0 001.Black_footed_Albatross
1 002.Laysan_Albatross
2 003.Sooty_Albatross
3 004.Groove_billed_Ani
4 005.Crested_Auklet
5 006.Least_Auklet
6 007.Parakeet_Auklet
7 008.Rhinoceros_Auklet
8 009.Brewer_Blackbird
9 010.Red_winged_Blackbird
10 011.Rusty_Blackbird
11 012.Yellow_headed_Blackbird
12 013.Bobolink
13 014.Indigo_Bunting
14 015.Lazuli_Bunting
15 016.Painted_Bunting
16 017.Cardinal
17 018.Spotted_Catbird
18 019.Gray_Catbird
19 020.Yellow_breasted_Chat
20 021.Eastern_Towhee
21 022.Chuck_will_Widow
22 023.Brandt_Cormorant

```

Figure 3.82: Gambar 22

```

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

```

Figure 3.83: Gambar 23

## 7. Program Cross Validation dan Penjelasan Keluarannya :

- Code Cross Validation 1 :
  - Penjelasan : Codingan tersebut menunjukkan hasil dari Cross Validation Random Forest. Pada pemrosesannya dilakukan pengecekan terhadap Cross Validation untuk Random Forest sesuai dengan data yang dieksekusi.
- Code Cross Validation 2 :
  - Penjelasan : Codingan tersebut menunjukkan hasil dari Cross Validation Decision Tree. Pada pemrosesannya dilakukan pengecekan terhadap Cross Validation untuk Decision Tree sesuai dengan data yang dieksekusi.
- Code Cross Validation 3 :

```

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

```

Figure 3.84: SVM

```

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

```

Figure 3.85: Decision Tree

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

## 8. Program Pengamatan Komponen Informasi dan Penjelasan Keluarannya :

- Code Pengamatan Komponen Informasi 1 :
  - Penjelasan : Codingan ini memberikan hasil pengamatan komponen informasi dimana pada pengeksekusianya dapat diketahui berapa banyak tree yang dibuat, berapa banyak atribut yang dipakai dan informasi lainnya terkait kode yang digunakan.
- Code Pengamatan Komponen Informasi 2 :
  - Penjelasan : Codingan ini menunjukkan plot dari komponen informasi agar bisa dibaca dimana pada pemrosesannya plotting dilakukan pada informasi dengan menggunakan kode.

### 3.3.3 Penanganan Error

#### Penyelesaian Tugas Harian 6 ( Penanganan Error )

- (a) Menyelesaikan dan Membahas Penanganan Error :
  - Error 1 :
    - Penjelasan Error 1 :
 

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

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

Figure 3.86: Cross Validation 1

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

Figure 3.87: Cross Validation 2

FileNotFoundException: File b'/data/CUB\_200\_2011/attributes/  
image\\_attribute\\_labels.txt' does not exist

- Penyelesaian Error 1 :

- i. Pertama-tama perhatikan file yang akan dieksekusi yaitu ”image\_attribute\_labels.txt”
  - ii. Pastikan file tersebut berada satu tempat ( directory ) dengan file bird-identifier.py yang digunakan sebagai codingan untuk pengeksekusian file
  - iii. Selanjutnya hapus codingan dan sesuai seperti contoh codingan berikut :
- ```
imgatt = pd.read_csv('image_attribute_labels.txt')
```
- iv. Maka tidak akan terjadi error lagi

- Penjelasan Error 2 :

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

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

- Penyelesaian Error 2 :

- i. Pertama-tama perhatikan file yang akan dieksekusi yaitu ”image_class_labels.txt”
- ii. Pastikan file tersebut berada satu tempat (directory) dengan file bird-identifier.py yang digunakan sebagai codingan untuk pengeksekusian file
- iii. Selanjutnya hapus codingan dan sesuai seperti contoh codingan berikut :

```

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

```

Figure 3.88: Cross Validation 3

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

iv. Maka tidak akan terjadi error lagi

- Penjelasan Error 3 :

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

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

- Penyelesaian Error 3 :

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

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

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

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

iv. Maka tidak akan terjadi error lagi

```

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

```

Figure 3.89: Program Pengamatan Komponen Informasi 1

```

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

```

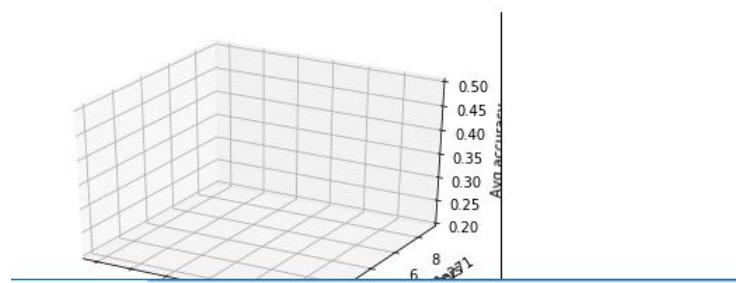


Figure 3.90: Program Pengamatan Komponen Informasi 2

```

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 3.91: Error



Figure 3.92: random forest

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

Figure 3.93: random forest 1

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

Figure 3.94: random forest 2

Name: a
Date: Sat, 04 Mar 2017 10:41
Dataset ID: 1044
Column names: Country, Age, Salary, Purchased

Figure 3.95: random forest 3

| Index | Country | Age | Salary | Purchased |
|-------|---------|-----|--------|-----------|
| 0 | France | 44 | 72000 | No |
| 1 | Spain | 27 | 48000 | Yes |
| 2 | Germany | 38 | 54000 | No |
| 3 | Spain | 38 | 61000 | No |
| 4 | Germany | 48 | 100000 | Yes |
| 5 | France | 35 | 56000 | Yes |

Figure 3.96: random forest 4

| | | | | | | | | | |
|---|---|---|---|---|----------------|---|---|----|----|
| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
| 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | |
| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
| | | | | | Data Pelajaran | | | | |
| | | | | | Data Pelatihan | | | | |

Figure 3.97: cross validation

| Y | Y Pred | Keluaran Untuk (threshold) 0.6 | Recall { pemanggilan/penarikan} | Precision { Presisi} |
|---|--------|-----------------------------------|------------------------------------|-------------------------|
| 0 | 0.5 | 0 | | |
| 1 | 0.9 | 1 | | |
| 0 | 0.7 | 1 | | |
| 1 | 0.7 | 1 | 1 / 2 | |
| 1 | 0.3 | 0 | | |
| 0 | 0.4 | 0 | | |
| 1 | 0.5 | 0 | | |

Figure 3.98: confusion matrix

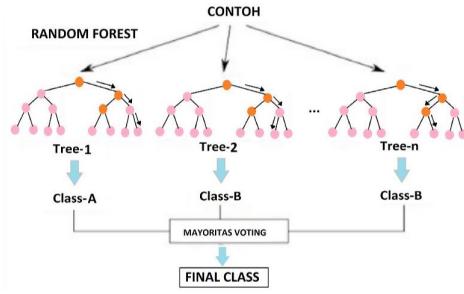


Figure 3.99: voting random forest

```

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

Figure 3.100: pandas

```

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

Figure 3.101: numpy

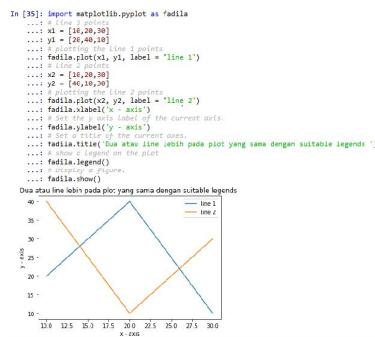


Figure 3.102: matplotlib

```
In [2]: import pandas as pd
...:
...: # give lines description for plotting
...: imgatt = pd.read_csv('imgatt.csv', header=None, names=[ 'imgid', 'attid', 'present'])
...: imgatt.set_index('imgid', inplace=True)
...: # description from dataset (HEAD):
...: # the set of attribute labels as perceived by software for each image
...: # each row corresponds to one image, each column to one attribute
...: # each line corresponding to one image/attribute/label triplet
...: # a single line corresponds to one image/attribute/label triplet
...: # attid, present, imgid are the three columns
...: # in imgatt, attid corresponds to the attribute id
...: # in imgatt, attid corresponds to the attribute id
...: # in imgatt, attid corresponds to the attribute id
...: # present: 0 or 1 denotes if the attribute is present in the image
...: # present: 0 or 1 denotes if the attribute is present in the image
...: # present: other denotes the time spent by the filter in seconds.
```

Figure 3.103: random forest

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

Figure 3.104: random forest 2

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

Figure 3.105: random forest 3

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

Figure 3.106: random forest 4

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

Figure 3.107: random forest 5

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

Figure 3.108: random forest 6

```
In [8]: imglabels = pd.read_csv('image_class_labels.csv',
...:                           sep=',', header=None, names=[ 'imgid', 'label'])
...: imglabels.set_index('imgid', inplace=True)
...: # description from dataset (HEAD):
...: # the set of class labels (first specific labels) for each image are contained
...: # in the file image_class_labels.txt, with each line corresponding to one image
...: # a single line corresponds to one image
...: # in imglabels, imgid corresponds to the imgid in images.txt and classes.txt,
...: # and represents:
```

Figure 3.109: random forest 7

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

Figure 3.110: random forest 8

```
In [10]: imgLabels.shape
Out[10]: (11789, 1)
```

Figure 3.111: random forest 9

```
In [11]: df = imgLabels.groupby(imgLabels)
```

Figure 3.112: random forest 10

```
In [12]: df_ext = df.agg(lambda x: x[0])
... df_mean = df.agg(lambda x: x[1].mean())
... df_std = df.agg(lambda x: x[1].std())
```

Figure 3.113: random forest 11

```
In [13]: df_ext, df_mean, df_std
```

| | | | | | | | | | | | | | | | |
|------|---|---|---|---|---|---|-----|-----|-----|-----|-----|-----|-----|-----|---|
| 1 | 2 | 3 | 4 | 5 | 6 | 7 | ... | 366 | 367 | 368 | 369 | 370 | 371 | 372 | |
| 5902 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| 4432 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| 4430 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| 4770 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |

(8 rows x 32 columns)

Figure 3.114: random forest 12

```
In [14]: df_label.head()
```

| |
|-------|
| label |
| 1 |
| 0 |
| 0 |
| 0 |

Figure 3.115: random forest 13

```
In [15]: df_train_ext = df[0:8000]
... df_train_label = df[0:8000]
... df_train_data = df[0:8000]
... df_train_label = df['train_label'].values
... df_train_data = df['train_data'].values
```

Figure 3.116: random forest 14

```
In [16]: from sklearn.ensemble import RandomForestClassifier
... clt = RandomForestClassifier(max_features=10, random_state=0, n_estimators=100)
```

Figure 3.117: random forest 15

```
In [17]: clt.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 3.118: random forest 16

```
In [18]: pred = clt.predict(df_train_att)
```

Figure 3.119: random forest 17

```
In [19]: clt.score(df_test_att, df_test_label)
```

Figure 3.120: random forest 18

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

Figure 3.121: confusion matrix 1

```
In [24]: np
array([[ 0,  0,  0, ...,  0,  0,  0],
       [ 0,  0,  0, ...,  0,  0,  0],
       [ 0,  0,  0, ...,  0,  0,  0],
       [ 0,  0,  0, ...,  0,  0,  0],
       [ 0,  0,  0, ...,  0,  0,  0],
       [ 0,  0,  0, ...,  0,  0,  0], dtype=int64)
```

Figure 3.122: confusion matrix 2

```
In [25]: import numpy as np
from sklearn.metrics import classification_report, confusion_matrix
from sklearn import metrics
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")
    if normalize:
        print(cm)
    else:
        print("Confusion matrix, without normalization")

    plt.imshow(cm, interpolation='nearest', cmap=cmap)
    plt.title(title)
    plt.colorbar()
    tick_marks = np.arange(len(classes))
    plt.xticks(tick_marks, classes, rotation=45)
    plt.yticks(tick_marks, classes)

    for i in range(len(classes)):
        for j in range(len(classes)):
            plt.text(j, i, cm[i, j])
    plt.tight_layout()
    plt.ylabel('True label')
    plt.xlabel('Predicted label')

def plot_confusion_matrix(cm, classes):
    plt.imshow(cm, interpolation='nearest', cmap=plt.cm.Blues)
    plt.title('Confusion matrix, without normalization')
    plt.colorbar()
    tick_marks = np.arange(len(classes))
    plt.xticks(tick_marks, classes, rotation=45)
    plt.yticks(tick_marks, classes)

    for i in range(len(classes)):
        for j in range(len(classes)):
            plt.text(j, i, cm[i, j])
    plt.tight_layout()
    plt.ylabel('True label')
    plt.xlabel('Predicted label')
```

Figure 3.123: confusion matrix 3

```
In [26]: birds = pd.read_csv("classes.txt",
                         sep='\t', header=None, names=[1, 'birds'])
# birds = birds[['0', '1', '2', '3', '4', '5', '6', '7', '8', '9', '10', '11', '12', '13', '14', '15', '16', '17', '18', '19', '20', '21', '22', '23', '24', '25', '26', '27', '28', '29', '30', '31', '32', '33', '34', '35', '36', '37', '38', '39', '40', '41', '42', '43', '44', '45', '46', '47', '48', '49', '50', '51', '52', '53', '54', '55', '56', '57', '58', '59', '60', '61', '62', '63', '64', '65', '66', '67', '68', '69', '70', '71', '72', '73', '74', '75', '76', '77', '78', '79', '80', '81', '82', '83', '84', '85', '86', '87', '88', '89', '90', '91', '92', '93', '94', '95', '96', '97', '98', '99', '100']]
# birds = birds[[0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49, 50, 51, 52, 53, 54, 55, 56, 57, 58, 59, 60, 61, 62, 63, 64, 65, 66, 67, 68, 69, 70, 71, 72, 73, 74, 75, 76, 77, 78, 79, 80, 81, 82, 83, 84, 85, 86, 87, 88, 89, 90, 91, 92, 93, 94, 95, 96, 97, 98, 99, 100]]
```

Figure 3.124: confusion matrix 4

```
In [27]: from sklearn import svm
clf = svm.SVC(gamma=0.001, C=1000)
clf.fit(birds[0:90], birds[100:100])
y_pred = clf.predict(birds[90:100])
print("Normalized confusion matrix")
cm = metrics.confusion_matrix(birds[90:100], y_pred, normalize=True)
cm = cm.astype('float') / cm.sum(axis=1)[:, np.newaxis]
print(cm)
```

Figure 3.125: confusion matrix 5

```
In [28]: from sklearn import tree
clf = tree.DecisionTreeClassifier(max_depth=2)
clf.fit(birds[0:90], birds[100:100])
y_pred = clf.predict(birds[90:100])
print("Normalized confusion matrix")
cm = metrics.confusion_matrix(birds[90:100], y_pred, normalize=True)
cm = cm.astype('float') / cm.sum(axis=1)[:, np.newaxis]
print(cm)
```

Figure 3.126: svm

```
In [29]: from sklearn import tree
clf = tree.DecisionTreeClassifier(max_depth=2)
clf.fit(birds[0:90], birds[100:100])
y_pred = clf.predict(birds[90:100])
print("Accuracy: %.2f" % metrics.accuracy_score(y_pred, birds[90:100]))
```

Figure 3.127: decision tree

```
In [30]: from sklearn.model_selection import cross_val_score
scores = cross_val_score(tree.DecisionTreeClassifier(),
                         birds[0:90], birds[100:100], cv=5)
print("Accuracy: %.2f" % (scores.mean() * 100))

Accuracy: 8.27
```

Figure 3.128: cross validation 1

```
In [31]: cross_validation = cross_val_score(clf, birds[0:90], birds[100:100], cv=5)
print("Accuracy: %.2f" % (cross_validation.mean() * 100))

Accuracy: 8.26
```

Figure 3.129: cross validation 2

```
In [32]: cross_validation = cross_val_score(clf, birds[0:90], birds[100:100], cv=5)
print("Accuracy: %.2f" % (cross_validation.mean() * 100))

Accuracy: 8.27
```

Figure 3.130: cross validation 3

```
In [28]: new_features.opts = myopt1(50, 2)
In [28]: new_features.opts = myopt1(200, 20)
In [29]: features = np.concatenate((train_attributes, new_features), 1)
In [30]: new_features = np.concatenate((test_attributes, new_features), 1)
```

Figure 3.131: pengamatan komponen informasi 1

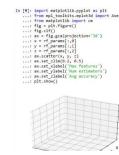


Figure 3.132: pengamatan komponen informasi 2

```
In [29]: import numpy as np
In [29]: from sklearn.svm import SVC
In [29]: from sklearn.metrics import accuracy_score
In [30]: svc = SVC(kernel='linear', C=1000, random_state=42)
In [31]: svc.fit(train_attributes, train_labels)
In [32]: predictions = svc.predict(test_attributes)
In [33]: accuracy = accuracy_score(test_labels, predictions)
In [33]: accuracy
Out[33]: 0.8645833333333333
```

Figure 3.133: error 1

```
File "c:\python34\lib\site-packages\sklearn\svm\classes.py", line 5, in <module>
    from .base import _SVC
File "c:\python34\lib\site-packages\sklearn\base.py", line 47, in <module>
    from .meta import clone, is_classifier, is_regressor
File "c:\python34\lib\site-packages\sklearn\meta.py", line 1, in <module>
    import copy
    import inspect
    import operator
    import sys
    import warnings
    import types
File "c:\python34\lib\copy.py", line 1, in <module>
    import _copy
    from _weakref import _WeakKeyDictionary
File "c:\python34\lib\_copy.py", line 2, in <module>
    import types
    import weakref
File "c:\python34\lib\operator.py", line 1, in <module>
    from _operator import *
    from _operator import __new__, __hash__, __call__
File "c:\python34\lib\weakref.py", line 1, in <module>
    from _weakref import WeakValueDictionary
File "c:\python34\lib\_weakref.py", line 1, in <module>
    from _weakref import WeakReference, WeakList, WeakSet
File "c:\python34\lib\weakref.py", line 2, in <module>
    import collections
    import functools
    import types
File "c:\python34\lib\collections\__init__.py", line 1, in <module>
    from _collections import *  # re-export public API
File "c:\python34\lib\functools.py", line 1, in <module>
    from _functools import *  # re-export public API
File "c:\python34\lib\types.py", line 1, in <module>
    from _types import *
File "c:\python34\lib\_types.py", line 1, in <module>
    from _operator import __new__, __hash__, __call__
File "c:\python34\lib\weakref.py", line 1, in <module>
    __weakref__ = _weakref
File "c:\python34\lib\_weakref.py", line 1, in <module>
    __weakref__ = _weakref
```

Figure 3.134: error 2

```
File "c:\python34\lib\site-packages\sklearn\svm\classes.py", line 1, in <module>
    from .base import _SVC
    from .metrics import classification_report, log_loss
File "c:\python34\lib\site-packages\sklearn\base.py", line 47, in <module>
    from .meta import clone, is_classifier, is_regressor
File "c:\python34\lib\site-packages\sklearn\meta.py", line 1, in <module>
    import copy
    import inspect
    import operator
    import sys
    import warnings
    import types
File "c:\python34\lib\copy.py", line 1, in <module>
    import _copy
    from _weakref import _WeakKeyDictionary
File "c:\python34\lib\_copy.py", line 2, in <module>
    import types
    import weakref
File "c:\python34\lib\operator.py", line 1, in <module>
    from _operator import *
    from _operator import __new__, __hash__, __call__
File "c:\python34\lib\weakref.py", line 1, in <module>
    from _weakref import WeakValueDictionary
File "c:\python34\lib\_weakref.py", line 1, in <module>
    from _weakref import WeakReference, WeakList, WeakSet
File "c:\python34\lib\weakref.py", line 2, in <module>
    import collections
    import functools
    import types
File "c:\python34\lib\collections\__init__.py", line 1, in <module>
    from _collections import *  # re-export public API
File "c:\python34\lib\functools.py", line 1, in <module>
    from _functools import *  # re-export public API
File "c:\python34\lib\types.py", line 1, in <module>
    from _types import *
File "c:\python34\lib\_types.py", line 1, in <module>
    from _operator import __new__, __hash__, __call__
File "c:\python34\lib\weakref.py", line 1, in <module>
    __weakref__ = _weakref
File "c:\python34\lib\_weakref.py", line 1, in <module>
    __weakref__ = _weakref
```

Figure 3.135: error 3

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 Lusia Violita Aprilian/1164080

4.3.1 Teori

1. Klasifikasi teks

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

2. Klasifikasi Bunga tidak dapat menggunakan machine learning

Klasifikasi bunga tidak dapat menggunakan machine learning karena memiliki masalah input yang serupa namun output yang berbeda atau 'noise'. Yang dimaksud dengan noise adalah contoh output yang direkam bukan seperti seharusnya. Misalnya saja kita secara implisit berasumsi bahwa contoh bunga

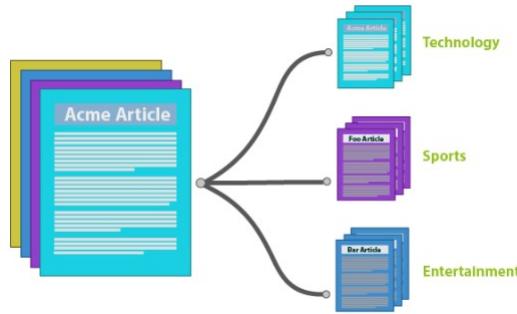


Figure 4.1: Lusia-Klasifikasi teks

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

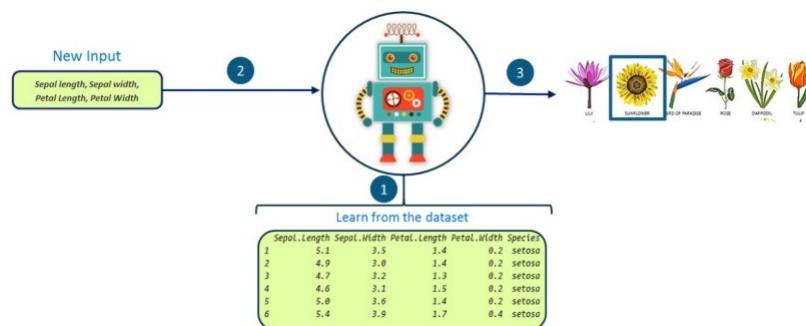


Figure 4.2: Lusia-Klasifikasi bunga

3. Teknik pembelajaran mesin pada teks YouTube

Dengan menggunakan kasus seperti rekomendasi video yang terdapat pada fiturnya, Machine Learning pada YouTube memperhatikan apa saja yang menarik perhatian para penggunanya. Ketika kita sedang menonton di YouTube, pada sebelah kanan terdapat 'Up Next' yang menampilkan beberapa video serupa yang sedang ditonton. Dan ketika mengklik salah satu video dari baris tersebut,

but, maka YouTube akan mengingatnya dan menggunakan kata yang tertera sebagai referensi.

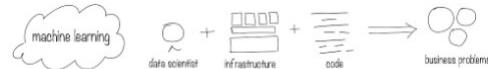


Figure 4.3: Lusia-Teknik YouTube

4. Vectorisasi Data

- Maksud dari Vectorisasi Data merupakan Pemecahan dan Pembagian Data.

5. Bag of word

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

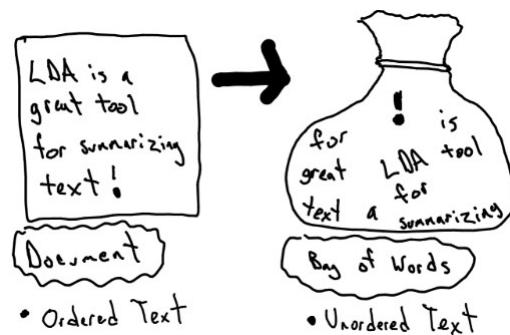


Figure 4.4: Lusia-Bag of Word

6. TF-IDF

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

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

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

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

Inverse document frequency # of documents
↑
 $\log \frac{1 + n_e}{1 + df(a, t)} + 1$

Document frequency of the term t

Figure 4.5: Lusia-TF IDF

4.3.2 Praktek

1. Aplikasi menggunakan pandas

Berikut adalah aplikasi yang dibuat menggunakan pandas :

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

Figure 4.6: Lusia-Pandas

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

Berikut adalah hasilnya :

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

Figure 4.7: Lusia-Hasil Pandas

```

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

```

Figure 4.8: Lusia-Pecah data

2. Memecah data frame

Berikut untuk memecah data frame menjadi dua :

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

Berikut adalah hasilnya :

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

Figure 4.9: Lusia-Hasil Pecah data

3. Vektorisasi dan klasifikasi Decission Tree Katty Perry

- Berikut adalah vektorisasi dan klasifikasi Katty Perry

```

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

```

Figure 4.10: Lusia-Vektorisasi dan klasifikasi

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

- Berikut adalah Decission Tree Katty Perry

Dalam gambar Decission Tree dijelaskan bahwa library tree dari sklearn. Dan mendefinisikan variable untuk memanggil Decission Tree Classifisier

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

Figure 4.11: Lusia-Decission Tree Katty Perry

yang kemudian dilakukan fit atau pengujian. Lalu menggunakan prediksi dengan fungsi predict untuk memprediksi test. Yang terakhir memunculkan akurasi prediksi yaitu 0,86.

4. Klasifikasikan dari data vektorisasi dengan klasifikasi SVM

Berikut adalah klasifikasikan dari data vektorisasi dengan klasifikasi SVM

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

Figure 4.12: Lusia-Hasil klasifikasi SVM

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

5. Klasifikasikan dari data vektorisasi dengan klasifikasi Decission Tree

Maksud dari gambar vektorisasi adalah hasil dari impor dataset

Berikut adalah Decission Tree

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

Figure 4.13: Lusia-Decission Tree

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

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

Figure 4.14: Lusia-ploting confusion matrix

6. Plot confusion matrix

Berikut adalah hasil dari ploting confusion matrix

Dari gambar dijelaskan bahwa library numpy di import sebagai np. Lalu Opsi set printoption untuk menentukan cara angka floating point, array dan objek NumPy lainnya ditampilkan. Selanjutnya matplotlib pyplot figure untuk membuat figur atau gambar baru. Selanjutnya confution matrix dinormalisasikan. Dan yang terakhir hasil ditampilkan.

7. Program cross validation

Berikut adalah hasil dari program cross validation

```
In [24]: from sklearn.model_selection import cross_val_score
...: scores = cross_val_score(clf,d_train_att,d_train_label,cv=5)
...:
...: skorrrata2=scores.mean()
...: skoresd=scores.std()
```

Figure 4.15: Lusia-Program cross validation

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

8. Program pengamatan komponen informasi

Berikut adalah hasil dari program pengamatan komponen informasi

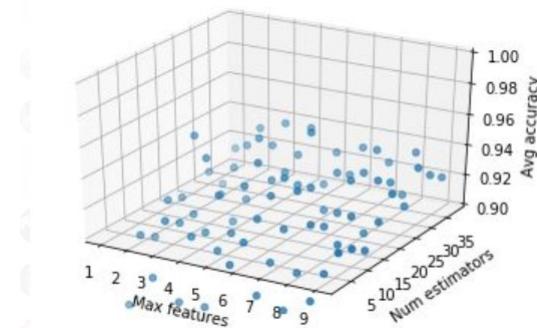


Figure 4.16: Lusia-Program pengamatan komponen informasi

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

4.3.3 Penanganan Error

1. skrinsut error

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

Figure 4.17: Lusia-skrinsut error

2. Tuliskan kode eror dan jenis errornya

- Kode error = KeyError: 'test preparation course'
- Jenis error = KeyError

3. Solusi pemecahan masalah error

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

4.4 Rahmi Roza-1164085

4.4.1 Teori

Penjelasan Tugas Harian 7 (No 1-6)

1. Pengertian Klasifikasi Teks Dan Ilustrasi Gambar

- Pengertian Klasifikasi Teks

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

- Ilustrasi Gambar

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

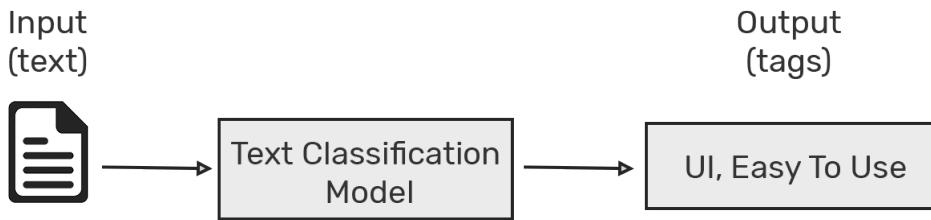


Figure 4.18: Klasifikasi Teks Roza

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

- Mengapa Klasifikasi Bunga Tidak Bisa Menggunakan Machine Learning

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

- Ilustrasi Gambar

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

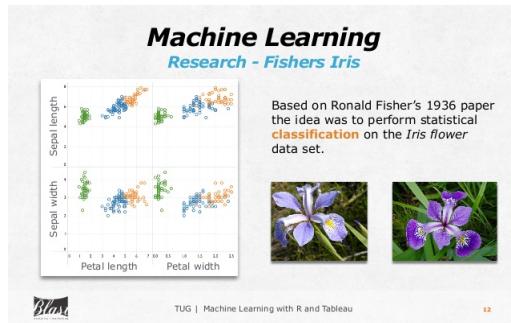


Figure 4.19: Klasifikasi Bunga Roza

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

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

Penjelasan : Penggunaan Machine Learning pada Youtube yaitu contohnya pada saat kita melakukan searching vidio atau pun bahan lainnya di youtube maka yang akan ditampilkan adalah vidio dari keyword yang

kita ketikkan di kotak pencarian. Dengan kata lain Youtube akan mem-filter vidio dengan keyword yang telah digunakan. Dan pada saat kita menonton Youtube pada bagian sebelah kanan tampilan youtuber terdapat vidio yang berkaitan dengan keyword yang kita cari tadi. Dimana disitulah Machine Learning pada Youtube menyimpan data.

- Ilustrasi Gambar

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

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

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

Out[9]:
```

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

Figure 4.20: Youtube Roza

4. Vektorisasi Data

- Maksud Dari Vektorisasi Data

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

5. Pengertian Bag Of Words Dan Ilustrasi Gambar

- Pengertian Bag Of Words

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

- Ilustrasi Gambar

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

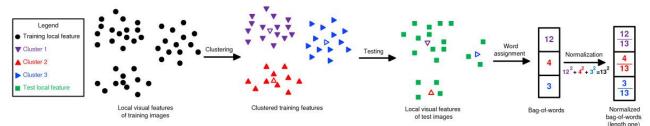


Figure 4.21: Bag of Words Roza

6. Pengertian TF-IDF Dan Ilustrasi Gambar

- Pengertian TF-IDF

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

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

- Ilustrasi Gambar

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

4.4.2 Praktek

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

- Baris 1: Memanggil library pandas sebagai pd
- Baris 2: Membaca dataset Balance Scale

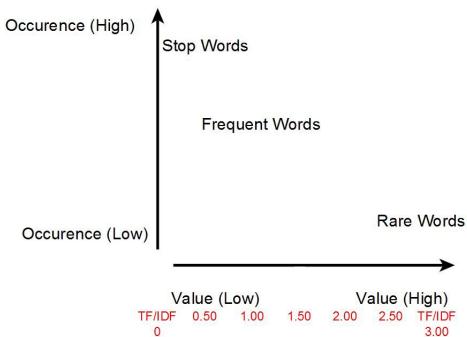


Figure 4.22: TF-IDF Rroza

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

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

Figure 4.23: Nomor 1 Roza

- Baris 3: Mengambil data frame dari library pandas
- Baris 4: Data Dummy digunakan untuk mengubah Categorical menjadi Integer
- Baris 5: Menampilkan 5 data teratas

(b) GAMBAR HASIL No1:

| | | | |
|----|-----------|----------|--|
| b1 | DataFrame | (501, 5) | Column names: class-name, left-weight, left-distance, right-weight, ri ... |
|----|-----------|----------|--|

Figure 4.24: Hasil 1 Roza

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

Figure 4.25: Hasil 1 Roza

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

- Baris 1: Membagi data set balance scale dari 500 data menjadi data train menjadi 450 data.

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

Figure 4.26: Nomor 2 Roza

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

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

Figure 4.27: Hasil 2 Roza

(d) Vektorisasi dan Klasifikasi Data Dengan Decission Tree

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

Figure 4.28: Nomor 3 Roza

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

In [12]: clf.predict(d_test_att)
Out[12]:
array([0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 1, 1, 0, 1, 1, 1,
       0, 1, 0, 1, 0, 1, 1, 1, 0, 1, 1, 1, 0, 0, 0, 0, 1, 1, 0, 1,
       1, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 1, 1, 0,
       1, 1, 0, 1, 1, 1, 0, 1, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 1, 0, 0,
       1, 0, 1, 1, 1, 0, 1, 0, 1, 1, 0, 1, 1, 0, 0, 0, 0, 1, 1, 0,
       1, 0, 1, 1, 1, 0, 1, 0, 0, 1, 1, 0, 1, 1, 0, 0, 0, 0, 1, 1, 0,
       1, 0, 1, 1, 1, 0, 1, 0, 0, 1, 1, 0, 1, 1, 0, 0, 0, 0, 1, 1, 0,
       0, 1, 1, 1, 0, 0, 1, 1, 0, 0, 1, 0, 1, 1, 0, 0, 0, 0, 1, 1, 0,
       0, 1, 1, 1, 0], dtype=int64)

In [13]: clf.score(d_test_att,d_test_label)
Out[13]: 0.9492753623188406
```

Figure 4.29: Hasil 3 Roza

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

```

In [14]: from sklearn import svm
...: clfsvm = svm.SVC()
...: clfsvm.fit(d_train_att, d_train_label)
...:
...: clfsvm.score(d_test_att, d_test_label)
C:\Users\ROZA\Anaconda3\lib\site-packages\sklearn\svm\base.py:196: FutureWarning: The default value of
gamma will change from 'auto' to 'scale' in version 0.22 to account better for unscaled features. Set
gamma explicitly to 'auto' or 'scale' to avoid this warning.
"avoid this warning.", FutureWarning)
Out[14]: 0.6884057971014492

```

Figure 4.30: Nomor 4 Roza

(e) Klasifikasi SVM

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

(f) Klasifikasi Decision Tree

```

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

```

Figure 4.31: Nomor 5 Roza

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

(g) Matplotlib

- Penjelasan:

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

```
In [48]: import matplotlib.pyplot as plt
...: import itertools
...: def plot_confusion_matrix(cm, classes,
...:                         normalize=False,
...:                         title='Confusion matrix',
...:                         cmap=plt.cm.Blues):
...:     """
...:     This function prints and plots the confusion matrix.
...:     Normalization can be applied by setting 'normalize=True'.
...:     """
...:     if normalize:
...:         cm = cm.astype('float') / cm.sum(axis=1)[:, np.newaxis]
...:         print("Normalized confusion matrix")
...:     else:
...:         print('Confusion matrix, without normalization')
...:
...:     print(cm)
...:
...:     plt.imshow(cm, interpolation='nearest', cmap=cmap)
...:     plt.title(title)
...:     #plt.colorbar()
...:     tick_marks = np.arange(len(classes))
...:     plt.xticks(tick_marks, classes, rotation=90)
...:     plt.yticks(tick_marks, classes)
...:
...:     fmt = '.2f' if normalize else 'd'
...:     thresh = cm.max() / 2.
```

Figure 4.32: Nomor 6 Roza

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

Figure 4.33: Nomor 7 Roza

(h) Menjelaskan Program Cross Validation

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

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

Figure 4.34: Hasil 7 Roza

| | | | |
|-----------|---------|---|----------------------|
| skoresd | float64 | 1 | 0.010208097193137975 |
| skorrata2 | float64 | 1 | 0.9500490877095491 |

Figure 4.35: Hasil 7 Roza

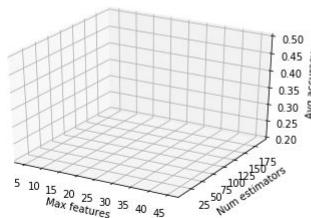
(i) Program Pengamatan Komponen

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

```
In [56]: max_features_opts = range(5, 50, 5)
...: n_estimators_opts = range(10, 200, 20)
...: rf_params = np.empty((len(max_features_opts)*len(n_estimators_opts),4), float)
...: i = 0
...: for max_features in max_features_opts:
...:     for n_estimators in n_estimators_opts:
...:         clf = RandomForestClassifier(max_features=max_features, n_estimators=n_estimators)
...:         scores = cross_val_score(clf, d_train_att, d_train_label, cv=5)
...:         rf_params[i,0] = max_features
...:         rf_params[i,1] = n_estimators
...:         rf_params[i,2] = scores.mean()
...:         rf_params[i,3] = scores.std() * 2
...:         i += 1
...:         print("Max features: %d, num estimators: %d, accuracy: %.2f (+/- %.2f)" % 
(max_features, n_estimators, scores.mean(), scores.std() * 2))
Max features: 5, num estimators: 10, accuracy: 0.91 (+/- 0.11)
Max features: 5, num estimators: 30, accuracy: 0.91 (+/- 0.07)
Max features: 5, num estimators: 50, accuracy: 0.93 (+/- 0.08)
Max features: 5, num estimators: 70, accuracy: 0.93 (+/- 0.07)
Max features: 5, num estimators: 90, accuracy: 0.93 (+/- 0.06)
Max features: 5, num estimators: 110, accuracy: 0.94 (+/- 0.06)
Max features: 5, num estimators: 130, accuracy: 0.94 (+/- 0.06)
Max features: 5, num estimators: 150, accuracy: 0.93 (+/- 0.07)
Max features: 5, num estimators: 170, accuracy: 0.93 (+/- 0.07)
Max features: 5, num estimators: 190, accuracy: 0.94 (+/- 0.07)
Max features: 10, num estimators: 10, accuracy: 0.93 (+/- 0.03)
Max features: 10, num estimators: 30, accuracy: 0.93 (+/- 0.09)
```

Figure 4.36: Nomor 8 Roza

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



[In [58]:

Figure 4.37: Nomor 8 Roza

4.4.3 Penanganan Error

i. Skrinsut Error

```
...: df = pd.DataFrame(bl, columns = ['class-name','left-weight','left-distance','right-weight','right-distance'])
Traceback (most recent call last):
  File "<ipython-input-54-4858d7fef12>", line 2, in <module>
    bl = pd.read_csv('balancee.csv', sep=',')
  File "C:\Users\ROZA\Anaconda3\lib\site-packages\pandas\io\parsers.py", line 678, in parser_f
    return _read(filepath_or_buffer, kwds)
  File "C:\Users\ROZA\Anaconda3\lib\site-packages\pandas\io\parsers.py", line 440, in _read
    parser = TextFileReader(filepath_or_buffer, **kwds)
  File "C:\Users\ROZA\Anaconda3\lib\site-packages\pandas\io\parsers.py", line 787, in __init__
    self._make_engine(self.engine)
  File "C:\Users\ROZA\Anaconda3\lib\site-packages\pandas\io\parsers.py", line 1014, in _make_engine
    self._engine = CParserWrapper(self.f, **self.options)
  File "C:\Users\ROZA\Anaconda3\lib\site-packages\pandas\io\parsers.py", line 1708, in __init__
    self._reader = parsers.TextReader(src, **kwds)
  File "pandas\_libs\parsers.pyx", line 384, in pandas._libs.parsers.TextReader.__cinit__
  File "pandas\_libs\parsers.pyx", line 695, in pandas._libs.parsers.TextReader._setup_parser_source
FileNotFoundError: File b'balancee.csv' does not exist
```

Figure 4.38: Error Roza

ii. Kode Error dan Jenis Errornya

Kode Error: "FileNotFoundException" dan "File b 'balancee.csv' does not exist".

Jenis Error: Import Dataset

iii. Penanganan

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

4.5 Fadila-1164072

4.5.1 Teori

Penjelasan Tugas Harian 7 (No 1-6)

1. Pengertian Klasifikasi Teks Dan Ilustrasi Gambar

- Pengertian Klasifikasi Teks

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

- Ilustrasi Gambar

Penjelasan : Berdasarkan pengertian diatas, ada beberapa contoh yang bisa diterapkan. Untuk salah satu contoh dari klasifikasi data sendiri dapat diliat pada gambar berikut 5.8.

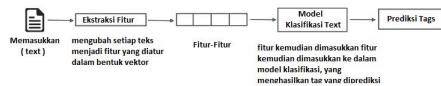


Figure 4.39: text-fadila

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

• Mengapa Klasifikasi Bunga Tidak Bisa Menggunakan Machine Learning

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

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

• Ilustrasi Gambar

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

| 1. Train model using data with known desired output | | | | | |
|---|-------------|-----------------|------------------|--------|--------|
| | Flower 1 | Flower 2 | Labels | Mean 1 | Mean 2 |
| sepal length | 5.4 | 4.3 | fixed acidity | 6.80 | 6.80 |
| sepal width | 3.9 | 3.4 | volatile acidity | 0.350 | 0.270 |
| petal length | 1.7 | 1.4 | citric acid | 0.30 | 0.30 |
| petal width | 0.4 | 0.3 | residual sugar | 5.70 | 13.00 |
| Classification | iris setosa | iris versicolor | Regression | 4 | 7 |

Figure 4.40: bunga-fadila

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

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

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

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

- Ilustrasi Gambar

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

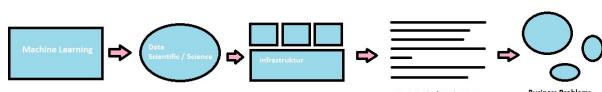


Figure 4.41: youtube-fadila

4. Vektorisasi Data

- Maksud Dari Vektorisasi Data

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

Array angka floating point (Mewakili data) :

- teks
- audio
- gambar

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

5. Pengertian Bag Of Words Dan Ilustrasi Gambar

- Pengertian Bag Of Words

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

- Ilustrasi Gambar

Penjelasan : Berdasarkan pengertian diatas, ada beberapa contoh yang bisa diterapkan. Untuk salah satu contoh dari Bag Of Words sendiri dapat diliat pada gambar berikut 4.42.

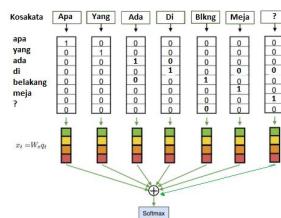


Figure 4.42: bag-fadila

6. Pengertian TF-IDF Dan Ilustrasi Gambar

- Pengertian TF-IDF

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

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

- Ilustrasi Gambar

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

SCORE TF-IDF (RUMUS / PERHITUNGAN)
$$TF - IDF Score = TF_{x,y} * IDF = TF_{x,y} * \log \frac{N}{df} \dots\dots (1)$$

Figure 4.43: tf2-fadila

4.5.2 Praktek Fadila

Pengerjaanya Tugas Harian 8 (No 1-8)

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

- Penjelasan Code Pandas Sederhana

```
import pandas as pd  
fadila_car = pd.read_csv('1.csv', sep=';')  
len(fadila_car)  
  
fadila = pd.DataFrame(fadila_car, columns = [ 'buying' , 'maint' ,  
dummy = pd.get_dummies ( fadila [ 'class' ] )  
dummy.head()
```

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

- Hasil Dari Aplikasi Sederhana Code Pandas

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

```
In[8]:  
# load dataset (student Portuguese scores)  
import pandas as pd  
fادila_car = pd.read_csv('l1.csv', sep=';')  
len(fادila_car)  
  
fادila = pd.DataFrame(fادila_car, columns = ['buying', 'maint', 'doors', 'persons', 'lug-boot', 'safety', 'class'])  
fادila  
  
dummy = pd.get_dummies (fادila['class'])  
dummy.head()
```

Figure 4.44: 1-fadila

| Name | Type | Size | Value |
|--------|-----------|----------|--|
| dummy | DataFrame | (500, 2) | Column names: acc, unacc |
| fادila | DataFrame | (500, 7) | Column names: buying, maint, doors, persons, lug-boot, safety, class |

Figure 4.45: lanjutan1-fadila

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

Figure 4.46: lanjutan1-1-fadila

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

Figure 4.47: lanjutan1-2-fadila

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

- Dataframe 1 : 450 row data pertama
- Dataframe 2 : 50 row data sisanya.
- Codingan Dan Penjelasannya :

```
fادila_car_train = fادila_car [:450]  
fادila_car_test = fادila_car [451:]
```

- (a) Baris 1 : Membuat variabel fadila_car_train dari variabel fadila_car dengan 450 data awal untuk dijadikan data training

- (b) Baris 2 : Membuat variabel fadila_car_test dari variabel fadila_car dengan 50 data sisanya untuk data testing (dimulai dari data ke 451 sampai akhir data)

- Hasil Dari Pemecahan Dataframe

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

```
# In [21]: fadila_car_train = fadila_car[0:450]
          fadila_car_test = fadila_car[451:]
```

Figure 4.48: 2-fadila

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

Figure 4.49: lanjutan2-fadila

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

```
from sklearn import tree
```

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

Penjelasan : Sesuai Hasil.

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

- Hasil Program Menggunakan Decision Tree

Hasilnya dapat diliat pada gambar berikut 4.50.

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

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

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

Figure 4.50: 3-fadila

4. Vektorisasi Dengan Klasifikasi SVM

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

clfsvm.score(d_test_att, d_test_label)
```

Penjelasan :

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

- Hasil Klasifikasi SVM

Hasilnya dapat diliat pada gambar berikut 4.51.

```
In [27]: from sklearn import svm
... clfsvm = svm.SVC()
... clfsvm.fit(d_train_att, d_train_label)
...
clfsvm.score(d_test_att, d_test_label)
C:\ProgramData\Anaconda\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]: 0.96
```

Figure 4.51: 4-fadila

5. Vektorisasi Menggunakan Klasifikasi Decision Tree

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

Penjelasan :

- (a) Dalam [in 31] merupakan hasil dari impor tree dari sklearn. Dan mendefinisikan variabel clf untuk memanggil Decission Tree Classifier dan melakukan fit atau pengujian.
- (b) Di dalamnya juga menjelaskan bahwa digunakan prediksi untk clf dengan function predict untuk memprediksi test. Dan hasilnya muncul dalam bentuk array.

- (c) clf score memunculkan akurasi prediksi yang dilakukan terhadap clf
- (d) Lalu muncul lah hasilnya pada [out 31] seperti pada gambar

- Hasil Program Cross Validation

Hasilnya dapat diliat pada gambar berikut 4.52.

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

Figure 4.52: 5-fadila

6. Plot Confusion Matrix

- Code Plot Confusion Matrix

```
In [40]: import matplotlib.pyplot as plt
from sklearn.metrics import classification_report, confusion_matrix
# This function prints and plots the confusion matrix.
# Normalization can be applied by setting `normalize=True`.
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]
    else:
        print('Confusion matrix, without normalization')
    print(cm)

    plt.imshow(cm, interpolation='nearest')
    plt.title(title)
    plt.colorbar()
    tick_marks = np.arange(len(classes))
    plt.xticks(tick_marks, classes, rotation=45)
    plt.yticks(tick_marks, classes)

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

Figure 4.53: cod6-fadila

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

- Hasil Plot Confusion Matrix

Hasilnya dapat diliat pada gambar berikut 4.54 dan 4.55 .

```
In [40]: import matplotlib.pyplot as plt
from sklearn.metrics import classification_report, confusion_matrix
# This function prints and plots the confusion matrix.
# Normalization can be applied by setting `normalize=True`.
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]
    else:
        print('Confusion matrix, without normalization')
    print(cm)

    plt.imshow(cm, interpolation='nearest')
    plt.title(title)
    plt.colorbar()
    tick_marks = np.arange(len(classes))
    plt.xticks(tick_marks, classes, rotation=45)
    plt.yticks(tick_marks, classes)

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

Figure 4.54: 6-fadila

7. Program Cross Validation

```
In [40]: import numpy as np
...: np.set_printoptions(precision=2)
...: cm = confusion_matrix(y_true, y_pred)
...: print(cm)
...: plt.show()
Normalized confusion matrix
[[1.  0.]
 [0.  0.99]]
```

Figure 4.55: lanjutan6-fadila

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

skorrata2=scores.mean()
skoresd=scores.std()
```

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

- Hasil Program Cross Validation

Hasilnya dapat diliat pada gambar berikut 4.56 dan 4.57 .

```
In [54]: from sklearn.model_selection import cross_val_score
...: scores = cross_val_score(clf,d_train_att,d_train_label, cv=5)
...: skorrata2=scores.mean()
...: skoresd=scores.std()
```

Figure 4.56: 7-fadila

| | | | |
|-----------|---------|---|----------------------|
| skoresd | float64 | 1 | 0.010910084509667118 |
| skorrata2 | float64 | 1 | 0.94993794572566946 |

Figure 4.57: lanjutan7-fadila

8. Program Pengamatan Komponen Informasi

- Hasil Dan Code Pengamatan Komponen Informasi :

Hasilnya dapat diliat pada gambar berikut 4.58 dan 4.59.

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

Figure 4.58: 8-fadila

```
In [14]: import matplotlib.pyplot as plt
...: from mpl_toolkits.mplot3d import Axes3D
...: from matplotlib import cm
...: fig = plt.figure()
...: ax = fig.add_subplot(111, projection='3d')
...: x = rf_params[:,0]
...: y = rf_params[:,1]
...: z = rf_params[:,2]
...: ax.set_xlabel('x')
...: ax.set_ylabel('y')
...: ax.set_zlabel('z')
...: ax.set_xlabel('Max features')
...: ax.set_ylabel('Num estimators')
...: ax.set_zlabel('Avg accuracy')
...: plt.show()
```

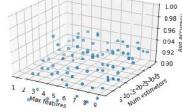


Figure 4.59: lanjutan8-fadila

Penjelasan :

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

4.5.3 Penanganan Error Fadila

Menangani dan Mengatasi Error Pada Praktek

1. Error 1 :

- Code Yang Error :

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

- Peringatan Error :

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

- Gambar Error 1 : (Lebih jelas)

```
File "C:\ProgramData\Anaconda\lib\site-packages\pandas\io\parsers.py", line 787, in  
    _init_  
        self._make_engine(self.engine)  
File "C:\ProgramData\Anaconda\lib\site-packages\pandas\io\parsers.py", line 1014, in  
    _make_engine  
        self._engine = CParserWrapper(self.f, **self.options)  
File "C:\ProgramData\Anaconda\lib\site-packages\pandas\io\parsers.py", line 1706, in  
    _init_  
        self._reader = parsers.TextReader(frc, **kwargs)  
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'dataset/Youtube02-KatyPerry.csv' does not exist
```

Figure 4.60: error1-fadila

- Cara Penanganan :

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

- Silahkan tuliskan codingan berikut untuk mengganti codingan yang bermasalah

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

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

4.5.4 Praktek

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 Rahmi Roza-1164085

5.5.1 Teori

Praktek Tugas Harian

1. Mengapa Kata-Kata Harus di Vektorisasi

Kata harus di vektorisasi dikarenakan mesin hanya mampu membaca data dalam bentuk angka. Oleh karena itu diperlukan vektorisasi kata agar mesin mampu membaca data yang telah di vektorisasi.

- Gambar :

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

Word Vector Representations

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

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

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

Figure 5.1: Vektorisasi Kata Roza

2. Mengapa Dimensi Dari Vektor Dataset Google Bisa Sampai 300

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

- Gambar :

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

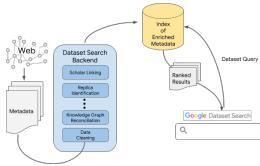


Figure 5.2: Google Dataset Roza

3. Konsep Vektorisasi Kata

Konsep vektorisasi data merupakan kata-kata yang di inputkan pada mesin learning. Dan outputnya berupa kara-kata atau keyword dari pencarian yang telah dilakukan sebelumnya. Contohnya pada saat kita melakukan pencarian di channel youtube kita. Maka akan muncul hasil dari pencarian dari kata-kata yang telah kita cari.

- Gambar :

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

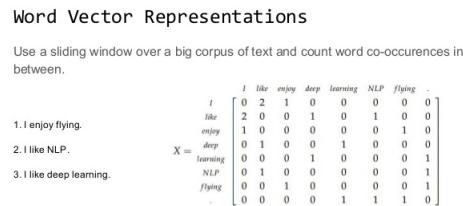


Figure 5.3: Vektorisasi Kata Roza

4. Konsep Vektorisasi Dokumen

Konsep vektorisasi dokumen yaitu mesin akan membaca terlebih dahulu semua kalimat yang ada di dalam dokumen dan nanti kalimat yang ada di dalam dokument tersebut akan diolah menjadi kata-kata.

- Gambar :

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

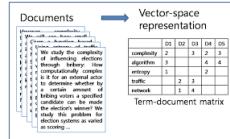


Figure 5.4: Vektorisasi Dokumen Roza

5. Mean dan Standar Deviasi

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

- Gambar :

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

$$\begin{aligned} \text{Mean} &= \frac{\text{Sum of all data values}}{\text{Number of data values}} \\ &= \frac{15+13+18+16+14+17+12}{7} \\ &= \frac{105}{7} \\ &= 15 \end{aligned}$$

Figure 5.5: Mean Roza



Figure 5.6: Standar Deviasi Roza

6. Skip Gram

Skip-Gram mencoba memprediksi vektor kata-kata yang ada di konteks diberikan vektor kata tertentu. Skip-gram membuat sepasang kata target dan konteks sebagai sebuah instance sehingga skip-gram cenderung lebih baik ketika ukuran corpus sangat besar.

- Gambar :

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

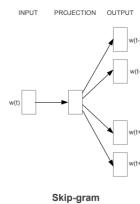


Figure 5.7: Skip-Gram Roza

- Plagiarisme Roza



Figure 5.8: Plagiarisme Roza

5.6 Fadila-1164072

5.6.1 Teori

Penjelasan Tugas Harian 9 (No 1-6). (No. 7 Bukti Plagiarisme)

1. Mengapa Kata-Kata Harus Di Lakukan Vektorisasi Dan Ilustrasi Gambar

- Penjelasan :

Alasan mengapa kata-kata harus dilakukan vektorisasi terlebih dahulu yaitu dikarenakan mesin hanya mampu membaca data dengan bentuk angka. Berdasarkan hal tersebut maka tentunya diperlukan vektorisasi kata atau bisa disebut dengan mengubah kata menjadi bentuk vektor agar mesin seolah-olah paham apa yang kita maksudkan dan dapat memproses aktifitas/perintah dengan benar. Kata juga harus di vektorisas iuntuk mengetahui presentase kata yang sering muncul dalam setiap kalimatnya, yang berguna untuk menetukan kata kunci.

- Ilustrasi Gambar

Penjelasan : Berdasarkan penjelasan diatas, ada beberapa contoh yang bisa diterapkan. Untuk salah satu contoh dari vektorisasi kata dapat diliat pada gambar berikut 5.9.

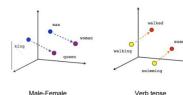


Figure 5.9: Vektorisasi Kata-fadila

2. Mengapa Dimensi Dari Vektor Dataset Google Bisa Mencapai 300 Dan Ilustrasi Serta Contoh Gambar

- Penjelasan :

Dimensi dari Vektor Dataset Google Bisa Mencapai 300 itu dikarenakan pada masing-masing objek yang terdapat pada dataset akan memiliki identitasnya tersendiri, selain itu juga untuk nilai dalam vektor 300 dimensi yang terkait dalam sebuah kata "dioptimalkan" dalam berbagai hal untuk menangkap aspek yang berbeda dari makna dan penggunaan kata itu. secara singkatnya terdapat ada lebih dari 3 miliar kata-kata dan kalimat yang tidak mungkin disimpan dalam 1 dimensi vektor, lalu disimpan menjadi 300 dimensi vektor untuk mengatasi yang namanya kegagalan memori

- Ilustrasi :(berdasarkan identitas tersendiri) Apabila dicontohkan dengan penjelasan yang lebih rinci maka dilakukan perumpamaan sederhana. Misalnya untuk sebuah dataset google yang memiliki 3 buah objek yaitu berat, lebar, dan tinggi. Kemudian dari masing-masing objek tersebut dilakukan perbandingan antara berat dan lebar beserta berat dan tinggi. Hasil yang didapatkan akan memiliki presentasi yang berbeda sehingga dapat diartikan bahwa mesin dapat membedakan objek yang hampir serupa namun tak sama.

- Contoh Gambar

Penjelasan : Berdasarkan penjelasan diatas, ada beberapa contoh yang lain bisa diterapkan. Untuk salah satu contoh dari vektor dataset googlei dapat diliat pada gambar berikut 5.10 dan 5.11.

3. Konsep Vektorisasi Untuk Kata Dan Ilustrasi Gambar

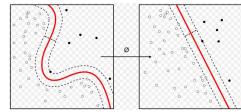


Figure 5.10: Dimensi Vektor Dataset Wikipedia-fadila

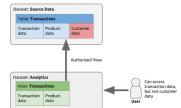


Figure 5.11: Dimensi Vektor Dataset -fadila

- Penjelasan :

Konsep untuk vektorisasi kata sebenarnya sama dengan ketika dilakukan input suatu kata pada mesin pencarian. Kemudian untuk hasilnya akan mengeluarkan (berupa) referensi mengenai kata tersebut. Jadi data kata tersebut didapatkan dari hasil pengolahan pada kalimat-kalimat sebelumnya yang telah diolah. Contoh sederhananya pada kalimat berikut (Please click the alarm icon for more notifications about my channel), pada kalimat tersebut terdapat konteks yakni channel, kata tersebut akan dijadikan data latih untuk mesin yang akan dipelajari dan diproses. Jadi ketika kita inputkan kata channel, maka mesin akan menampilkan keterkaitannya dengan kata tersebut sehingga akan lebih efisien dan lebih mudah.

- Ilustrasi Gambar

Penjelasan : Berdasarkan konsep diatas, ada beberapa contoh yang bisa diterapkan. Untuk salah satu contoh dari vektorisasi kata dapat diliat pada gambar berikut 5.12.

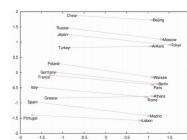


Figure 5.12: Vektorisasi Untuk Kata-fadila

4. Konsep Vektorisasi Untuk Dokumen Dan Ilustrasi Gambar

- Penjelasan :

Untuk vektorisasi dokumen sebenarnya terbilang sama dengan konsep vektorisasi kata, yang membedakan hanya pada proses awalnya (pada eksekusi

awal). Untuk vektorisasi dokumen ini, mesin akan membaca semua kalimat yang terdapat pada dokumen tersebut, kemudian kalimat yang terdapat pada dokumen tersebut akan di pecah menjadi kata-kata. Seperti itulah konsep vektorisasi dokumen.

- Ilustrasi Gambar

Penjelasan : Berdasarkan konsep diatas, ada beberapa contoh yang bisa diterapkan. Untuk salah satu contoh dari vektorisasi dokumen dapat diliat pada gambar berikut 5.13.



Figure 5.13: Vektorisasi Untuk Dokumen-fadila

5. Pengertian Mean Dan Standar Deviasi Beserta Ilustrasi Gambar

- Pengertian Mean :

Mean merupakan nilai rata-rata dari suatu data. Mean sendiri dapat dicari dengan cara membagi jumlah data dengan banyak data sehingga diperoleh lah nilai rata-rata dari suatu data yang dicari / tersebut.

- Pengertian Standar Deviasi :

Untuk standar deviasi sendiri merupakan sebuah teknik statistik yang digunakan dalam menjelaskan homogenitas kelompok ataupun dapat diartikan dengan nilai statistik dimana dimanfaatkan untuk menentukan bagaimana sebaran data dalam sampel, serta seberapa dekat titik data individu ke mean atau rata-rata nilai sampel yang ada.

- Ilustrasi Gambar

Penjelasan : Berdasarkan penjelasan diatas, ada beberapa contoh yang bisa diterapkan. Untuk salah satu contoh dari mean dan standar deviasi sendiri dapat diliat pada gambar berikut 5.14 , 5.15 dan 5.16.

| Population Mean | Sample Mean |
|--------------------------|--------------------------------|
| $\mu = \frac{\sum X}{N}$ | $\bar{X} = \frac{\sum x_i}{n}$ |

N = number of items in the population

n = number of items in the sample

Figure 5.14: Mean-fadila



Figure 5.15: Mean Lanjutan-fadila

$$\text{Rumus Deviasi Standar}$$

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

$$SD = \sqrt{\frac{\sum (x - \bar{x})^2}{N}}$$

Keterangan:
 (1) \bar{x} = Rata-rata
 (2) x = Setiap angka pada data set
 (3) Rumus untuk standar deviasi negatif atau positif

Figure 5.16: Standar Deviasi-fadila

6. Penjelasan Skip-gram Dan Ilustrasi Gambar

- Penjelasan :

Sebuah teknik yang digunakan di area speech processing, dimana n-gram yang dibentuk kemudian ditambahkan juga dengan tindakan “skip” pada token-tokennya.

Untuk membentuk k-skip-n-grams, ada dua nilai yang harus didefinisikan, dimana kedua nilai tersebut yaitu k (jumlah kata yang di-skip) dan n (banyak kata dalam n-gram, e.g. bigram (2-gram), trigram (3-gram), dll.).

- Ilustrasi Gambar

Penjelasan : Berdasarkan penjelasan diatas, ada beberapa contoh yang bisa diterapkan. Untuk salah satu contoh dari skip-gram sendiri dapat diliat pada gambar-gambar berikut 5.18 .

Continuous Skip-gram Model

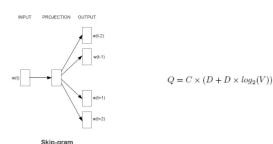


Figure 5.17: Skip Gram - fadila

7. Plagiarisme Fadila :

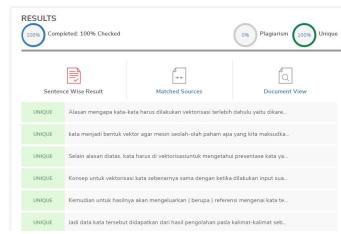


Figure 5.18: Plagiarisme - fadila

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) |
| TOTAL | | | 36 | |
| Catatan : Nilai minimal untuk diterima 25 | | | | |

Figure A.2: form nilai bagian 2.

Appendix B

FAQ

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

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

M : Pa saya sudah cari jurnal terindeks scopus tapi ga nemu. D : Kamu punya mata de? coba dicolok dulu. Kamu udah lakuin apa aja? tolong di list laporkan ke grup Tingkat Akhir. Tinggal buka google scholar klik dari tahun 2014, cek nama jurnalnya di scimagojr.com beres.

M : Pa saya belum dapat tempat intership, jadi ga tau mau presentasi apa? D : kamu kok ga nyambung, yang dipresentasikan itu yang kamu baca bukan yang akan kamu lakukan.

M : Pa ini jurnal harus yang terindex scopus ga bisa yang lain ? D : Index scopus menandakan artikel tersebut dalam standar semantik yang mudah dipahami dan dibaca serta bukan artikel asal jadi. Jika diluar scopus biasanya lebih sukar untuk dibaca dan dipahami karena tidak adanya proses review yang baik dan benar terhadap artikel.

M : Pa saya tidak mengerti D : Coba lihat standar alasan

M : Pa saya bingung D : Coba lihat standar alasan

M : Pa saya sibuk D : Mbahmu....

M : Pa saya ganteng D : Ndasmu....

M : Pa saya kece D : wes karepmu lah....

Biasanya anda memiliki alasan tertentu jika menghadapi kendala saat proses bimbingan, disini saya akan melakukan standar alasan agar persepsi yang diterima sama dan tidak salah kaprah. Penggunaan kata alasan tersebut antara lain :

1. Tidak Mengerti : anda boleh menggunakan alasan ini jika anda sudah melakukan tahapan membaca dan meresumekan 15 jurnal. Sudah mencoba dan mempraktekkan teorinya dengan mencari di youtube dan google minimal 6 jam sehari selama 3 hari berturut-turut.
2. Bingung : anda boleh mengatakan alasan bingung setelah maksimal dalam berusaha menyelesaikan tugas bimbingan dari dosen(sudah dilakukan semua). Anda belum bisa mengatakan alasan bingung jika anda masih belum menyelesaikan tugas bimbingan dan poin nomor 1 diatas. Setelah anda menyelesaikan tugas bimbingan secara maksimal dan tahap 1 poin diatas, tapi anda masih tetap bingung maka anda boleh memakai alasan ini.

Bibliography

- [1] Joshua Eckroth. *Python Artificial Intelligence Projects for Beginners: Get up and running with Artificial Intelligence using 8 smart and exciting AI applications.* Packt Publishing Ltd, 2018.
- [2] Stuart J Russell and Peter Norvig. *Artificial intelligence: a modern approach.* Malaysia; Pearson Education Limited,, 2016.