

CERDAS MENGUASAI GIT

CERDAS MENGUASAI GIT

Dalam 24 Jam

Rolly M. Awangga
Informatics Research Center



Kreatif Industri Nusantara

Penulis:

Rolly Maulana Awangga

ISBN : 978-602-53897-0-2

Editor:

M. Yusril Helmi Setyawan

Penyunting:

Syafril Fachrie Pane

Khaera Tunnisa

Diana Asri Wijayanti

Desain sampul dan Tata letak:

Deza Martha Akbar

Penerbit:

Kreatif Industri Nusantara

Redaksi:

Jl. Ligar Nyawang No. 2

Bandung 40191

Tel. 022 2045-8529

Email : awangga@kreatif.co.id

Distributor:

Informatics Research Center

Jl. Sariasih No. 54

Bandung 40151

Email : irc@poltekpos.ac.id

Cetakan Pertama, 2019

Hak cipta dilindungi undang-undang

Dilarang memperbanyak karya tulis ini dalam bentuk dan dengan cara
apapun tanpa ijin tertulis dari penerbit

*'Jika Kamu tidak dapat
menahan lelahnya
belajar, Maka kamu
harus sanggup menahan
perihnya Kebodohan.'*

Imam Syafi'i

CONTRIBUTORS

ROLLY MAULANA AWANGGA, Informatics Research Center., Politeknik Pos Indonesia, Bandung, Indonesia

CONTENTS IN BRIEF

| | |
|--------------------|------------|
| 1 Chapter 1 | 1 |
| 2 Chapter 2 | 57 |
| 3 Chapter 3 | 123 |
| 4 Chapter 4 | 137 |
| 5 Chapter 5 | 139 |
| 6 Chapter 6 | 141 |
| 7 Chapter 7 | 143 |

DAFTAR ISI

| | |
|--|----------|
| Daftar Gambar | xiii |
| Daftar Tabel | xxiii |
| Foreword | xxxii |
| Kata Pengantar | xxxiii |
| Acknowledgments | xxxv |
| Acronyms | xxxvii |
| Glossary | xxxix |
| List of Symbols | xli |
| Introduction | xliii |
| <i>Rolly Maulana Awangga, S.T., M.T.</i> | |
| 1 Chapter 1 | 1 |
| 1.1 1174035 - Luthfi Muhammad Nabil | 1 |
| 1.1.1 Sejarah dan perkembangan Kecerdasan Buatan | 1 |
| 1.1.2 Supervised Learning | 2 |
| 1.1.3 Unsupervised Learning | 2 |

| | | |
|--------|---|----|
| 1.1.4 | Jenis - Jenis Dataset | 3 |
| 1.1.5 | Instalasi dan Percobaan Kompilasi dari Library Scikit-learn | 3 |
| 1.1.6 | Mencoba Loading and example dataset | 5 |
| 1.1.7 | Learning and Predicting | 7 |
| 1.1.8 | Model Persistence | 8 |
| 1.1.9 | Conventions | 10 |
| 1.1.10 | Skrinsut Error | 14 |
| 1.1.11 | Kode error dan jenis error tersebut | 14 |
| 1.1.12 | Penanganan Error | 15 |
| 1.1.13 | Plagiarisme | 16 |
| 1.2 | 1174040 - Hagan Rowlenstino A. S | 16 |
| 1.2.1 | Teori | 16 |
| 1.2.2 | Instalasi | 17 |
| 1.2.3 | Penanganan Error | 20 |
| 1.2.4 | Cek Plagiarism | 21 |
| 1.3 | 1174042 Faisal Najib Abdullah | 21 |
| 1.3.1 | Teori | 21 |
| 1.3.2 | Instalasi | 22 |
| 1.3.3 | Penanganan eror | 27 |
| 1.3.4 | Plagiat | 28 |
| 1.3.5 | Link | 28 |
| 1.4 | 1174043 - Irvan Rizkiansyah | 29 |
| 1.4.1 | Definisi Kecerdasan Buatan | 29 |
| 1.4.2 | Sejarah dan Perkembangan | 29 |
| 1.4.3 | Supervised Learning | 29 |
| 1.4.4 | Unsupervised Learning | 30 |
| 1.4.5 | Teknik Klasifikasi | 30 |
| 1.4.6 | Regresi | 30 |
| 1.4.7 | Training Set | 30 |
| 1.4.8 | Testing Set | 30 |
| 1.4.9 | Instalasi dan Percobaan Kompilasi dari Library Scikit-learn | 30 |
| 1.4.10 | Mencoba Loading an example dataset | 32 |
| 1.4.11 | Mencoba Learning and Predicting | 33 |
| 1.4.12 | Mencoba Model Persistence | 33 |
| 1.4.13 | Mencoba Conventions | 34 |
| 1.5 | 1174050 Dika Sukma Pradana | 34 |

| | | |
|----------|--------------------------------|-----------|
| 1.5.1 | Teori | 34 |
| 1.5.2 | Instalasi | 36 |
| 1.5.3 | Percobaan | 36 |
| 1.5.4 | Penanganan eror | 41 |
| 1.5.5 | Plagiarism | 42 |
| 1.6 | 1174057 Alit Fajar Kurniawan | 42 |
| 1.6.1 | Teori | 42 |
| 1.6.2 | Praktek | 43 |
| 1.6.3 | Penanganan Error | 50 |
| 1.6.4 | Bukti Tidak Plagiat | 51 |
| 1.7 | 1174039 - Liyana Majdah Rahma | 51 |
| 1.7.1 | Teori | 51 |
| 1.7.2 | Instalasi | 52 |
| 1.7.3 | Penanganan Error | 56 |
| 1.7.4 | Cek Plagiarism | 56 |
| 2 | Chapter 2 | 57 |
| 2.1 | 1174042 Faisal Najib Abdullah | 57 |
| 2.1.1 | Teori | 57 |
| 2.1.2 | Sikic-Learn | 62 |
| 2.1.3 | Penanganan Error | 68 |
| 2.1.4 | Plagiat | 69 |
| 2.2 | 1174035 Luthfi Muhammad Nabil | 69 |
| 2.2.1 | Teori | 69 |
| 2.2.2 | Scikit-Learn | 74 |
| 2.2.3 | Skrinsut Error | 79 |
| 2.2.4 | Penanganan Error | 79 |
| 2.2.5 | Plagiarisme | 79 |
| 2.3 | 1174057 - Alit Fajar Kurniawan | 80 |
| 2.3.1 | Teori | 80 |
| 2.3.2 | Praktek | 81 |
| 2.3.3 | Penanganan Error | 81 |
| 2.3.4 | Bukti Tidak Plagiat | 81 |
| 2.4 | 1174050 Dika Sukma Pradana | 81 |
| 2.4.1 | Teori | 81 |
| 2.4.2 | Sikic-Learn | 85 |
| 2.4.3 | Penanganan Error | 93 |
| 2.4.4 | Plagiat | 94 |

| | | |
|----------|----------------------------------|------------|
| 2.5 | 1174039 Liyana Majdah Rahma | 94 |
| 2.5.1 | Teori | 94 |
| 2.5.2 | Sikic-Learn | 98 |
| 2.5.3 | Penanganan Error | 105 |
| 2.5.4 | Plagiat | 106 |
| 2.6 | 1174043 Irvan Rizkiansyah | 106 |
| 2.6.1 | Teori | 106 |
| 2.6.2 | Scikit-Learn | 109 |
| 2.7 | 1174040 - Hagan Rowlenstino A. S | 113 |
| 2.7.1 | Teori | 113 |
| 2.7.2 | scikit-learn | 118 |
| 2.7.3 | Penanganan Error | 122 |
| 3 | Chapter 3 | 123 |
| 3.0.1 | Soal Teori | 123 |
| 3.1 | Faisal Najib Abdullah / 1174042 | 127 |
| 3.1.1 | Teori | 127 |
| 3.1.2 | Praktikum | 129 |
| 3.1.3 | Penanganan Error / cokro | 135 |
| 4 | Chapter 4 | 137 |
| 5 | Chapter 5 | 139 |
| 6 | Chapter 6 | 141 |
| 7 | Chapter 7 | 143 |

DAFTAR GAMBAR

| | | |
|------|------------------------------|---|
| 1.1 | Instalasi Scikit Learn | 3 |
| 1.2 | Daftar Example | 3 |
| 1.3 | Variable Explorer | 4 |
| 1.4 | Hasil Percobaan 1 | 5 |
| 1.5 | Hasil Percobaan 2 | 5 |
| 1.6 | Hasil Percobaan 3 | 6 |
| 1.7 | Hasil Percobaan 4 | 6 |
| 1.8 | Hasil pada variable explorer | 6 |
| 1.9 | Hasil Percobaan 1 | 7 |
| 1.10 | Hasil Percobaan 2 | 7 |
| 1.11 | Hasil Percobaan 3 | 7 |
| 1.12 | Hasil pada variable explorer | 8 |
| 1.13 | Hasil Percobaan 1 | 8 |

| | | |
|------|------------------------------------|----|
| 1.14 | Hasil Percobaan 2 | 9 |
| 1.15 | Hasil Percobaan 3 | 9 |
| 1.16 | Hasil Percobaan 4 | 9 |
| 1.17 | Hasil Percobaan 5 | 9 |
| 1.18 | Hasil pada variable explorer | 10 |
| 1.19 | Hasil Percobaan 1 | 11 |
| 1.20 | Hasil Percobaan 2 | 11 |
| 1.21 | Hasil Percobaan 3 | 11 |
| 1.22 | Hasil Percobaan 4 | 12 |
| 1.23 | Hasil Percobaan 5 | 12 |
| 1.24 | Hasil Percobaan 6 | 12 |
| 1.25 | Hasil pada variable explorer | 13 |
| 1.26 | Hasil Percobaan 6 | 14 |
| 1.27 | Hasil pada variable explorer | 16 |
| 1.28 | Install Library Scikit | 17 |
| 1.29 | Variable Exploler | 18 |
| 1.30 | Data Digits | 18 |
| 1.31 | Digits Target | 19 |
| 1.32 | Data 2D | 19 |
| 1.33 | Data 2D | 20 |
| 1.34 | Cek Plagiarism | 21 |
| 1.35 | Installasi | 23 |
| 1.36 | Mencoba Loading an example Dataset | 24 |
| 1.37 | Learning and Predicting | 24 |
| 1.38 | Model Presistence | 25 |
| 1.39 | Model Presistence | 25 |
| 1.40 | Model Presistence | 26 |
| 1.41 | Error | 28 |

| | | |
|------|--------------------------------|----|
| 1.42 | Error | 28 |
| 1.43 | Instalasi Scikit Learn | 31 |
| 1.44 | Variable Explorer | 32 |
| 1.45 | Dataset | 33 |
| 1.46 | Predicting | 33 |
| 1.47 | Instalasi | 36 |
| 1.48 | Variabel Explore | 37 |
| 1.49 | Datasets | 37 |
| 1.50 | Error | 41 |
| 1.51 | Plagiarism | 42 |
| 1.52 | Instalasi Scikit Learn | 44 |
| 1.53 | Example | 44 |
| 1.54 | Example | 45 |
| 1.55 | Result Data Digits | 46 |
| 1.56 | Result digits.target | 46 |
| 1.57 | Result digits.image | 46 |
| 1.58 | Result Learning and predicting | 47 |
| 1.59 | Result Model persistence | 48 |
| 1.60 | Result Conventions | 50 |
| 1.61 | Error | 50 |
| 1.62 | Error | 50 |
| 1.63 | Plagiarisme | 51 |
| 1.64 | Instalasi | 53 |
| 1.65 | Variable Exploler | 53 |
| 1.66 | Variable Exploler | 54 |
| 1.67 | Data Digits | 54 |
| 1.68 | Digits Target | 54 |
| 1.69 | Data 2D | 55 |

| | | |
|------|--------------------------------|----|
| 1.70 | Data 2D | 56 |
| 1.71 | Cek Plagiarism | 56 |
| 2.1 | contoh binari calssification | 58 |
| 2.2 | contoh supervised learning | 58 |
| 2.3 | contoh unsupervised learning | 59 |
| 2.4 | contoh clusterring | 59 |
| 2.5 | contoh evaluasi dan akurasi | 60 |
| 2.6 | contoh Confusion Matrix | 60 |
| 2.7 | contoh K-fold cross validation | 61 |
| 2.8 | contoh decision tree | 62 |
| 2.9 | contoh information gain | 62 |
| 2.10 | hasil | 63 |
| 2.11 | hasil | 63 |
| 2.12 | hasil | 64 |
| 2.13 | hasil | 65 |
| 2.14 | hasil | 65 |
| 2.15 | hasil | 66 |
| 2.16 | hasil | 66 |
| 2.17 | hasil | 67 |
| 2.18 | hasil | 67 |
| 2.19 | hasil | 68 |
| 2.20 | hasil | 69 |
| 2.21 | Error | 69 |
| 2.22 | Error | 70 |
| 2.23 | contoh binary classification | 70 |
| 2.24 | contoh clustering | 71 |
| 2.25 | contoh K-fold cross validation | 72 |
| 2.26 | contoh information gain | 73 |

| | | |
|------|--------------------------------|----|
| 2.27 | contoh information gain | 74 |
| 2.28 | Hasil Percobaan 1 | 74 |
| 2.29 | Hasil Percobaan 2 | 75 |
| 2.30 | Hasil Percobaan 3 | 75 |
| 2.31 | Hasil Percobaan 4 | 76 |
| 2.32 | Hasil Percobaan 5 | 76 |
| 2.33 | Hasil Percobaan 6 | 76 |
| 2.34 | Hasil Percobaan 7 | 77 |
| 2.35 | Hasil Percobaan 8 | 77 |
| 2.36 | Hasil Percobaan 9 | 77 |
| 2.37 | Hasil Percobaan 10 | 78 |
| 2.38 | Hasil Percobaan 11 | 78 |
| 2.39 | Hasil Percobaan 12 | 79 |
| 2.40 | Error | 79 |
| 2.41 | Error | 79 |
| 2.42 | Hasil Pengecekan Plagiarisme | 79 |
| 2.43 | Klasifikasi Binari | 80 |
| 2.44 | Plagiarisme | 81 |
| 2.45 | contoh binari classification | 81 |
| 2.46 | contoh supervised learning | 82 |
| 2.47 | contoh unsupervised learning | 82 |
| 2.48 | contoh clustering | 82 |
| 2.49 | contoh evaluasi dan akurasi | 83 |
| 2.50 | contoh Confusion Matrix | 83 |
| 2.51 | contoh K-fold cross validation | 84 |
| 2.52 | contoh decision tree | 84 |
| 2.53 | contoh information gain | 84 |
| 2.54 | hasil | 85 |

| | | |
|------|--------------------------------|-----|
| 2.55 | hasil | 86 |
| 2.56 | hasil | 86 |
| 2.57 | hasil | 87 |
| 2.58 | hasil | 88 |
| 2.59 | hasil | 89 |
| 2.60 | hasil | 90 |
| 2.61 | hasil | 91 |
| 2.62 | hasil | 91 |
| 2.63 | hasil | 92 |
| 2.64 | hasil | 93 |
| 2.65 | Error | 93 |
| 2.66 | Error | 94 |
| 2.67 | contoh K-fold cross validation | 94 |
| 2.68 | contoh supervised learning | 95 |
| 2.69 | contoh unsupervised learning | 95 |
| 2.70 | contoh clusterring | 95 |
| 2.71 | contoh evaluasi dan akurasi | 96 |
| 2.72 | contoh Confusion Matrix | 96 |
| 2.73 | contoh K-fold cross validation | 96 |
| 2.74 | contoh decision tree | 97 |
| 2.75 | contoh information gain | 97 |
| 2.76 | hasil | 98 |
| 2.77 | hasil | 99 |
| 2.78 | hasil | 99 |
| 2.79 | hasil | 100 |
| 2.80 | hasil | 101 |
| 2.81 | hasil | 101 |
| 2.82 | hasil | 102 |

| | | |
|-------|--------------------------------|-----|
| 2.83 | hasil | 103 |
| 2.84 | hasil | 104 |
| 2.85 | hasil | 105 |
| 2.86 | hasil | 105 |
| 2.87 | Error | 105 |
| 2.88 | plagiat | 106 |
| 2.89 | contoh Binary Classification | 106 |
| 2.90 | contoh supervised learning | 107 |
| 2.91 | contoh unsupervised learning | 107 |
| 2.92 | contoh clustering | 107 |
| 2.93 | contoh Evaluasi | 108 |
| 2.94 | contoh Confusion Matrix | 108 |
| 2.95 | contoh K-fold cross validation | 108 |
| 2.96 | contoh decision tree | 109 |
| 2.97 | contoh information gain | 109 |
| 2.98 | Hasil Percobaan 1 | 109 |
| 2.99 | Hasil Percobaan 2 | 110 |
| 2.100 | Hasil Percobaan 3 | 110 |
| 2.101 | Hasil Percobaan 4 | 110 |
| 2.102 | Hasil Percobaan 5 | 111 |
| 2.103 | Hasil Percobaan 7 | 111 |
| 2.104 | Hasil Percobaan 9 | 112 |
| 2.105 | Hasil Percobaan 10 | 112 |
| 2.106 | Hasil Percobaan 11 | 113 |
| 2.107 | Hasil Percobaan 12 | 113 |
| 2.108 | Binary Classification | 114 |
| 2.109 | Supervised | 114 |
| 2.110 | Unsupervised | 115 |

| | | |
|-------|--|-----|
| 2.111 | lustering | 115 |
| 2.112 | Evaluasi | 116 |
| 2.113 | Confussion Matrix | 116 |
| 2.114 | K-Fold | 117 |
| 2.115 | Decision Tree | 117 |
| 2.116 | Information Gain | 118 |
| 2.117 | No 1 | 118 |
| 2.118 | No 2 | 119 |
| 2.119 | No 3 | 119 |
| 2.120 | No 4 | 119 |
| 2.121 | No 5 | 120 |
| 2.122 | No 7 | 120 |
| 2.123 | No 9 | 121 |
| 2.124 | No 10 | 121 |
| 2.125 | No 11 | 122 |
| 2.126 | No 12 | 122 |
| 2.127 | Screenshot Error | 122 |
| 3.1 | Contoh Confusion Matrix | 124 |
| 3.2 | Contoh Confusion Matrix | 126 |
| 3.3 | Contoh Random Forest yang sudah Divote | 127 |
| 3.4 | contoh binari calssification | 127 |
| 3.5 | contoh binari calssification | 129 |
| 3.6 | contoh binari calssification | 129 |
| 3.7 | hasil | 130 |
| 3.8 | hasil | 131 |
| 3.9 | hasil | 132 |
| 3.10 | hasil | 133 |
| 3.11 | hasil | 133 |

| | | |
|------|-------|-----|
| 3.12 | hasil | 134 |
| 3.13 | hasil | 134 |
| 3.14 | hasil | 135 |
| 3.15 | hasil | 136 |
| 3.16 | hasil | 136 |

DAFTAR TABEL

Listings

| | |
|---------------------------------|----|
| src/1174035/chapter1/sample1.py | 4 |
| src/1174035/chapter1/sample2.py | 5 |
| src/1174035/chapter1/sample2.py | 5 |
| src/1174035/chapter1/sample2.py | 5 |
| src/1174035/chapter1/sample2.py | 6 |
| src/1174035/chapter1/sample2.py | 6 |
| src/1174035/chapter1/sample3.py | 7 |
| src/1174035/chapter1/sample3.py | 7 |
| src/1174035/chapter1/sample3.py | 7 |
| src/1174035/chapter1/sample3.py | 8 |
| src/1174035/chapter1/sample4.py | 8 |
| src/1174035/chapter1/sample4.py | 9 |
| src/1174035/chapter1/sample4.py | 9 |
| src/1174035/chapter1/sample4.py | 9 |
| src/1174035/chapter1/sample4.py | 10 |
| src/1174035/chapter1/sample5.py | 10 |
| src/1174035/chapter1/sample5.py | 11 |

| | |
|--|----|
| src/1174035/chapter1/sample5.py | 11 |
| src/1174035/chapter1/sample5.py | 12 |
| src/1174035/chapter1/sample5.py | 14 |
| src/1174035/chapter1/sample3.py | 15 |
| src/1174040/chap1/ex1.py | 17 |
| src/1174040/chap1/ex2.py | 18 |
| src/1174040/chap1/ex2.py | 18 |
| src/1174040/chap1/ex2.py | 18 |
| src/1174040/chap1/ex2.py | 19 |
| src/1174040/chap1/no3.py | 19 |
| src/1174040/chap1/no4.py | 20 |
| src/1174040/chap1/no5.py | 20 |
| src/1174040/chap1/no3.py | 21 |
| src/1174042/chapter1/2,1.py | 22 |
| src/1174042/chapter1/2,2.py | 22 |
| src/1174042/chapter1/2,3.py | 23 |
| src/1174042/chapter1/2,4.py | 24 |
| src/1174042/chapter1/2,5.py | 25 |
| src/1174043/chapter1/sample1.py | 31 |
| src/1174043/chapter1/sample2.py | 32 |
| src/1174043/chapter1/sample3.py | 33 |
| src/1174043/chapter1/sample4.py | 33 |
| src/1174043/chapter1/sample5.py | 34 |
| src/1174050/chapter1/VAR.py | 36 |
| src/1174050/chapter1/dataset.py | 37 |
| src/1174050/chapter1/learning.py | 37 |
| src/1174050/chapter1/modelpersistance.py | 38 |
| src/1174050/chapter1/typecasting.py | 39 |
| src/1174050/chapter1/Multiclass.py | 40 |
| src/1174050/chapter1/Refitting.py | 40 |
| src/1174057/chapter1/example.py | 44 |
| src/1174057/chapter1/dataset.py | 45 |
| src/1174057/chapter1/learning.py | 47 |
| src/1174057/chapter1/modelpersistence.py | 47 |
| src/1174057/chapter1/conventions.py | 48 |

| | |
|---------------------------------|----|
| src/1174039/chapter1/no1.py | 53 |
| src/1174039/chapter1/no1.py | 53 |
| src/1174039/chapter1/no2.py | 54 |
| src/1174039/chapter1/no3.py | 55 |
| src/1174039/chapter1/no4.py | 55 |
| src/1174039/chapter1/no5.py | 55 |
| src/1174039/chapter1/no3.py | 56 |
| src/1174042/chapter2/2,1.py | 62 |
| src/1174042/chapter2/2,2.py | 63 |
| src/1174042/chapter2/2,3.py | 64 |
| src/1174042/chapter2/2,4.py | 64 |
| src/1174042/chapter2/2,5.py | 65 |
| src/1174042/chapter2/2,6.py | 65 |
| src/1174042/chapter2/2,7.py | 65 |
| src/1174042/chapter2/2,8.py | 66 |
| src/1174042/chapter2/2,9.py | 66 |
| src/1174042/chapter2/2,10.py | 67 |
| src/1174042/chapter2/2,11.py | 67 |
| src/1174042/chapter2/2,12.py | 68 |
| src/1174035/chapter2/sample1.py | 74 |
| src/1174035/chapter2/sample1.py | 74 |
| src/1174035/chapter2/sample1.py | 75 |
| src/1174035/chapter2/sample1.py | 75 |
| src/1174035/chapter2/sample1.py | 76 |
| src/1174035/chapter2/sample1.py | 76 |
| src/1174035/chapter2/sample1.py | 76 |
| src/1174035/chapter2/sample1.py | 77 |
| src/1174035/chapter2/sample1.py | 77 |
| src/1174035/chapter2/sample1.py | 77 |
| src/1174035/chapter2/sample1.py | 78 |
| src/1174035/chapter2/sample1.py | 78 |
| src/1174050/chapter2/1.py | 85 |
| src/1174050/chapter2/2.py | 85 |
| src/1174050/chapter2/3.py | 86 |
| src/1174050/chapter2/4.py | 87 |

| | |
|---------------------------------|-----|
| src/1174050/chapter2/5.py | 87 |
| src/1174050/chapter2/6.py | 88 |
| src/1174050/chapter2/7.py | 88 |
| src/1174050/chapter2/8.py | 89 |
| src/1174050/chapter2/9.py | 90 |
| src/1174050/chapter2/10.py | 91 |
| src/1174050/chapter2/11.py | 92 |
| src/1174050/chapter2/12.py | 92 |
| src/1174039/chapter2/1.py | 98 |
| src/1174039/chapter2/2.py | 98 |
| src/1174039/chapter2/3.py | 99 |
| src/1174039/chapter2/4.py | 100 |
| src/1174039/chapter2/5.py | 100 |
| src/1174039/chapter2/6.py | 101 |
| src/1174039/chapter2/7.py | 102 |
| src/1174039/chapter2/8.py | 102 |
| src/1174039/chapter2/9.py | 103 |
| src/1174039/chapter2/10.py | 103 |
| src/1174039/chapter2/11.py | 104 |
| src/1174039/chapter2/12.py | 105 |
| src/1174043/chapter2/1.py | 109 |
| src/1174043/chapter2/1.py | 109 |
| src/1174043/chapter2/1.py | 110 |
| src/1174043/chapter2/1.py | 110 |
| src/1174043/chapter2/1.py | 111 |
| src/1174043/chapter2/1.py | 112 |
| src/1174043/chapter2/1.py | 112 |
| src/1174043/chapter2/1.py | 113 |
| src/1174040/chapter2/1174040.py | 118 |
| src/1174040/chapter2/1174040.py | 118 |
| src/1174040/chapter2/1174040.py | 119 |
| src/1174040/chapter2/1174040.py | 119 |
| src/1174040/chapter2/1174040.py | 120 |
| src/1174040/chapter2/1174040.py | 120 |

| | |
|---------------------------------|-----|
| src/1174040/chapter2/1174040.py | 120 |
| src/1174040/chapter2/1174040.py | 120 |
| src/1174040/chapter2/1174040.py | 120 |
| src/1174040/chapter2/1174040.py | 121 |
| src/1174040/chapter2/1174040.py | 121 |
| src/1174040/chapter2/1174040.py | 122 |
| src/1174040/chapter2/err.py | 122 |
| src/1174042/chapter3/2,1.py | 129 |
| src/1174042/chapter3/2,2.py | 130 |
| src/1174042/chapter3/2,3.py | 131 |
| src/1174042/chapter3/2,4.py | 132 |
| src/1174042/chapter3/2,5.py | 133 |
| src/1174042/chapter3/2,6.py | 134 |
| src/1174042/chapter3/2,7.py | 134 |
| src/1174042/chapter3/2,8.py | 135 |

FOREWORD

Sepatah kata dari Kaprodi, Kabag Kemahasiswaan dan Mahasiswa

KATA PENGANTAR

Buku ini diciptakan bagi yang awam dengan git sekalipun.

R. M. AWANGGA

*Bandung, Jawa Barat
Februari, 2019*

ACKNOWLEDGMENTS

Terima kasih atas semua masukan dari para mahasiswa agar bisa membuat buku ini lebih baik dan lebih mudah dimengerti.

Terima kasih ini juga ditujukan khusus untuk team IRC yang telah fokus untuk belajar dan memahami bagaimana buku ini mendampingi proses Internship.

R. M. A.

ACRONYMS

| | |
|-------|---|
| ACGIH | American Conference of Governmental Industrial Hygienists |
| AEC | Atomic Energy Commission |
| OSHA | Occupational Health and Safety Commission |
| SAMA | Scientific Apparatus Makers Association |

GLOSSARY

| | |
|-------|--|
| git | Merupakan manajemen sumber kode yang dibuat oleh linus torvald. |
| bash | Merupakan bahasa sistem operasi berbasiskan *NIX. |
| linux | Sistem operasi berbasis sumber kode terbuka yang dibuat oleh Linus Torvald |

SYMBOLS

- A Amplitude
 - $\&$ Propositional logic symbol
 - a Filter Coefficient
- B Number of Beats

INTRODUCTION

ROLLY MAULANA AWANGGA, S.T., M.T.

Informatics Research Center
Bandung, Jawa Barat, Indonesia

Pada era disruptif saat ini. git merupakan sebuah kebutuhan dalam sebuah organisasi pengembangan perangkat lunak. Buku ini diharapkan bisa menjadi pengantar para programmer, analis, IT Operation dan Project Manajer. Dalam melakukan implementasi git pada diri dan organisasinya.

Rumusnya cuman sebagai contoh aja biar keren[?].

$$ABC\mathcal{DEF}\alpha\beta\Gamma\Delta \sum_{def}^{abc} \quad (I.1)$$

BAB 1

CHAPTER 1

1.1 1174035 - Luthfi Muhammad Nabil

Kecerdasan buatan merupakan kecerdasan yang dimasukkan ke sistem yang dapat diatur untuk kepentingan ilmiah. Kecerdasan buatan biasa disebut AI (Artificial Intelligence) yang didefinisikan sebagai kecerdasan ilmiah. AI memiliki kemampuan untuk menerjemahkan data dari luar, dan mempelajari data tersebut untuk dipelajari demi mencapai tujuan dan melakukan tugas tertentu sesuai hasil adaptasi berdasarkan data yang didapat.

1.1.1 Sejarah dan perkembangan Kecerdasan Buatan

AI mulai berkembang sesuai dengan konsep yang dikemukakan pada awal abad 17, René Descartes menyebutkan bahwa tubuh hewan bukanlah apa-apa melainkan mesin-mesin yang rumit. Lalu Blaise Pascal menciptakan mesin perhitungan digital mekanis pertama pada 1642. Selanjutnya pada abad ke 19, Charles Babbage dan Ada Lovelace menciptakan sebuah mesin penghitung mekanis yang dapat diprogram.

Pada tahun 1950-an, Program AI pertama yang sudah dapat difungsikan telah ditulis pada 1951 untuk menjalankan mesin Ferranti Mark I di University of Manchester yang merupakan sebuah program permainan naskah yang ditulis oleh Christopher Strachey. John McCarthy menyebutkan istilah kecerdasan buatan pada konferensi pertama yang disediakan untuk persoalan ini. Dilanjut pada tahun 1956, Beliau menemukan bahasa pemrograman yang bernama Lisp.

Jaringan saraf mulai digunakan secara luas pada tahun 1980-an, dimana algoritma perambatan balik pertama kali dijelaskan oleh Paul John Werbos pada tahun 1974. Selanjutnya di tahun 1982, para ahli fisika menganalisis sifat dari penyimpanan dan optimasi pada jaringan saraf menggunakan sistem statistika. Lalu dilanjutkan pada tahun 1985 sedikitnya empat kelompok riset menemukan algoritma pembelajaran propagansi balik. Algoritma ini berhasil diimplementasikan ke ilmu komputer dan psikologi. Dan pada tahun 1990, ditandai perolehan besar dalam berbagai bidang AI dan demonstrasi dari berbagai aplikasi yang sudah mengimplementasi. Seperti Deep Blue, sebuah komputer dari permainan catur yang dapat mengalahkan Garry Kasparov dalam sebuah pertandingan 6 game yang terkenal pada 1997.

1.1.2 Supervised Learning

Supervised learning adalah kondisi yang menggunakan variabel input dan output untuk dapat dilakukan pemetaan input output yang sudah didapat. Disebut Supervised Learning karena proses dari pembelajaran algoritma dari pembelajaran yang disumberkan dengan dataset dapat dipikirkan seperti seorang guru yang mengawasi proses pembelajaran. Proses pembelajaran dari algoritma akan berhenti saat algoritma sudah mendapatkan level dari performansi yang dapat diterima.

Masalah dari Supervised learning dapat dikelompokkan menjadi masalah dengan regresi dan klasifikasi

- Klasifikasi : Masalah dalam klasifikasi yang dimana output dari variable itu adalah kategori, seperti "Laki - laki" atau "Perempuan, dan "Muda" dan "Tua"
- Regresi : Masalah dalam regresi adalah jika pengeluaran dari variabel adalah sebuah nilai asli, seperti "suhu", dan "tinggi"

1.1.3 Unsupervised Learning

Unsupervised learning adalah kondisi dimana kamu hanya memiliki input data tanpa memiliki variabel output yang sesuai. Tujuan dari unsupervised learning adalah untuk memodelkan distribusi pada data untuk mengetahui lebih lanjut mengenai data. Disebut unsupervised learning karena pada metode ini, tidak ada jawaban yang tepat dan tidak ada pengarah. Sehingga algoritma

akan ditinggalkan sesuai rancangan demi menemukan dan dapat mengolah data yang menarik pada saat yang akan datang.

1.1.4 Jenis - Jenis Dataset

Dataset merupakan objek yang merepresentasikan data dan relasinya di memor. Strukturnya dapat mirip sesuai dengan struktur yang ada pada database namun bisa diubah sesuai dengan kebutuhan. Dataset juga berisi koleksi dari tabel data dan relasi data.

- Training set : merupakan sebuah dataset yang digunakan untuk kepentingan pembelajaran. Kepentingan tersebut akan disesuaikan dengan parameter yang ada.
- Test dataset : adalah sebuah dataset yang bersifat independen dibandingkan dengan training dataset, namun mengikuti probabilitas distribusi yang sama dengan training dataset. Jika model sudah sesuai dengan training dataset maka dataset sudah dapat disesuaikan dengan test dataset. Penyesuaian dari training dataset .

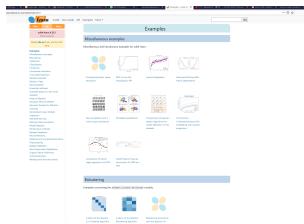
1.1.5 Instalasi dan Percobaan Kompilasi dari Library Scikit-learn

1. Buka anaconda prompt
2. Ketik di anaconda prompt yaitu : "pip install -U scikit-learn" untuk instalasi

```
(base) D:\Wallah\Python\KecerdasanBuatAnda>pip install -U scikit-learn
Collecting scikit-learn
  Downloading scikit-learn-0.22.1-pypi_0-py3-none-any.whl (6.3 MB)
Collecting scipy==1.7.0
  Downloading scipy-1.7.0-cp37-cp37m-win_amd64.whl (30.9 MB)
Collecting numpy==1.18.1
  Downloading numpy-1.18.1-cp37-cp37m-win_amd64.whl (12.8 MB)
Collecting joblib==0.14.1
  Downloading joblib-0.14.1-py3-none-any.whl (294 kB)
Collecting scikit-learn[scipy,numpy]
  Downloading scikit-learn-0.22.1-pypi_0-py3-none-any.whl (204 kB)
Successfully installed joblib-0.14.1 numpy-1.18.1 scikit-learn-0.22.1 scipy-1.4.1
(base) D:\Wallah\Python\KecerdasanBuatAnda>
```

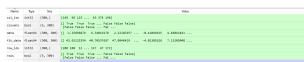
Gambar 1.1 Instalasi Scikit Learn

3. Setelah selesai instalasi, pilih salah satu example dari website Scikit (Contoh :)



Gambar 1.2 Daftar Example

4. Lalu coba jalankan aplikasi tersebut, bisa dicek hasil dari Variable explorernya



Gambar 1.3 Variable Explorer

5. Sample kode

```

1 print(__doc__)
2
3 # Author: Kemal Eren <kemal@kemaleren.com>
4 # License: BSD 3 clause
5
6 import numpy as np
7 from matplotlib import pyplot as plt
8
9 from sklearn.datasets import make_biclusters
10 from sklearn.cluster import SpectralCoclustering
11 from sklearn.metrics import consensus_score
12
13 data, rows, columns = make_biclusters(
14     shape=(300, 300), n_clusters=5, noise=5,
15     shuffle=False, random_state=0)
16
17 plt.matshow(data, cmap=plt.cm.Blues)
18 plt.title("Original dataset")
19
20 # shuffle clusters
21 rng = np.random.RandomState(0)
22 row_idx = rng.permutation(data.shape[0])
23 col_idx = rng.permutation(data.shape[1])
24 data = data[row_idx][:, col_idx]
25
26 plt.matshow(data, cmap=plt.cm.Blues)
27 plt.title("Shuffled dataset")
28
29 model = SpectralCoclustering(n_clusters=5, random_state=0)
30 model.fit(data)
31 score = consensus_score(model.biclusters_,
32                         (rows[:, row_idx], columns[:, col_idx]))
33
34 print("consensus score: {:.3f}".format(score))
35
36 fit_data = data[np.argsort(model.row_labels_)]
37 fit_data = fit_data[:, np.argsort(model.column_labels_)]
38
39 plt.matshow(fit_data, cmap=plt.cm.Blues)
40 plt.title("After biclustering; rearranged to show biclusters")
41

```

```
42 plt.show()
```

1.1.6 Mencoba Loading and example dataset

Disini akan dilakukan percobaan dengan menggunakan beberapa datasets seperti digits dan iris untuk bisa digunakan sebagai training set yang akan dipakai seluruh metode.

- Percobaan 1 (Memuat data iris dan digits dari datasets)

```
1 from sklearn import datasets #Untuk import class/fungsi
  dataset dari scikit-learn library
2 iris = datasets.load_iris() #Untuk memuat dan memasukkan
  dataset iris ke variabel bernama iris
3 digits = datasets.load_digits() #Untuk memuat dan memasukkan
  dataset digits ke variabel digits
```

```
In [18]: """
...: Created on Wed Feb 26 15:55:58 2020
...:
...: @author: Luthfi Muhammad Nabil
...: """
...:
...: from sklearn import datasets
...: iris = datasets.load_iris()
...: digits = datasets.load_digits()
```

Gambar 1.4 Hasil Percobaan 1

- Percobaan 2 (Menampilkan data dari digits)

```
1 print(digits.data) #Menampilkan object berformat Dictionary-
  like yang nanti akan ditampilkan pada console
```

```
In [19]: 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.]]
```

Gambar 1.5 Hasil Percobaan 2

- Percobaan 3 (Menampilkan digits.target)

```
1 digits.target #Menunjukkan data angka yang berhubungan dengan
  setiap digit gambar yang sedang dipelajari
```

```
In [20]: digits.target
Out[20]: array([0, 1, 2, ..., 8, 9, 8])
```

Gambar 1.6 Hasil Percobaan 3

- Percobaan 4 (Menampilkan data 2 dimensi)

```
1 digits.images[0] #Akan mengambil data dengan berformat array
2 Dimensi, dengan bentuk parameter (n_samples, n_features)
```

```
In [22]: digits.images[0]
Out[22]:
array([[ 0.,  0.,  5., 13.,  9.,  1.,  0.,  0.],
       [ 0.,  0., 13., 15., 10., 15.,  5.,  0.],
       [ 0.,  3., 15.,  2.,  0., 11.,  8.,  0.],
       [ 0.,  4., 12.,  0.,  0.,  8.,  8.,  0.],
       [ 0.,  5.,  8.,  0.,  0.,  9.,  8.,  0.],
       [ 0.,  4., 11.,  0.,  1., 12.,  7.,  0.],
       [ 0.,  2., 14.,  5., 10., 12.,  0.,  0.],
       [ 0.,  0.,  6., 13., 10.,  0.,  0.,  0.]])
```

Gambar 1.7 Hasil Percobaan 4

- Full sample

```
1 # -*- coding: utf-8 -*-
2 """
3 Created on Wed Feb 26 15:55:58 2020
4
5 @author: Luthfi Muhammad Nabil
6 """
7
8 from sklearn import datasets #Untuk import class/fungsi
9      dataset dari scikit-learn library
10 iris = datasets.load_iris() #Untuk memuat dan memasukkan
11      dataset iris ke variabel bernama iris
12 digits = datasets.load_digits() #Untuk memuat dan memasukkan
13      dataset digits ke variabel digits
14 #%%
15 print(digits.data) #Menampilkan object berformat Dictionary-
16      like yang nanti akan ditampilkan pada console
17 #%%
18 digits.target #Menunjukkan data angka yang berhubungan dengan
19      setiap digit gambar yang sedang dipelajari
20 #%%
21 digits.images[0] #Akan mengambil data dengan berformat array
22 Dimensi, dengan bentuk parameter (n_samples, n_features)
```

**Gambar 1.8** Hasil pada variable explorer

1.1.7 Learning and Predicting

Disini akan dicoba untuk melakukan prediksi berupa angka yang inputnya berupa gambaran dataset.

- Percobaan 1

```

1 from sklearn import svm #Untuk mengimport class sv dari
   library sklearn
2 from sklearn import datasets #Untuk import class/fungsi
   dataset dari scikit-learn library
3 clf = svm.SVC(gamma=0.001, C=100) #Memasukkan implementasi
   dari "Support Vector Classification" ke variabel clf
4 iris = datasets.load_iris() #Untuk memuat dan memasukkan
   dataset iris ke variabel bernama iris

```

Gambar 1.9 Hasil Percobaan 1

- Percobaan 2

```

1 digits = datasets.load_digits() #Untuk memuat dan memasukkan
   dataset digits ke variabel digits
2 clf.fit(digits.data[:-1], digits.target[:-1]) #Untuk
   melakukan pengiriman data training set ke method fit

```

Gambar 1.10 Hasil Percobaan 2

- Percobaan 3

```

1 clf.predict(digits.data[-1:]) #Untuk melakukan prediksi nilai
   yang baru berdasarkan gambar terakhir dari digits.data

```

Gambar 1.11 Hasil Percobaan 3

- Full sample

```

1 # -*- coding: utf-8 -*-
2 """
3 Created on Thu Feb 27 10:32:23 2020
4
5 @author: Luthfi Muhammad Nabil
6 """
7
8 from sklearn import svm #Untuk mengimport class sv dari
9     library sklearn
10 from sklearn import datasets #Untuk import class/fungsi
11     dataset dari scikit-learn library
12 clf = svm.SVC(gamma=0.001, C=100) #Memasukkan implementasi
13     dari "Support Vector Classification" ke variabel clf
14 iris = datasets.load_iris() #Untuk memuat dan memasukkan
15     dataset iris ke variabel bernama iris
16 #%%
17 digits = datasets.load_digits() #Untuk memuat dan memasukkan
18     dataset digits ke variabel digits
19 clf.fit(digits.data[:-1], digits.target[:-1]) #Untuk
20     melakukan pengiriman data training set ke method fit
21 #%%
22 clf.predict(digits.data[-1:]) #Untuk melakukan prediksi nilai
23     yang baru berdasarkan gambar terakhir dari digits.data

```



Gambar 1.12 Hasil pada variable explorer

1.1.8 Model Persistence

Disini akan dilakukan persistensi model menggunakan built-in dari Python

- Percobaan 1

```

1 from sklearn import svm #Mengimport class SVM dari library
2     sklearn
3 from sklearn import datasets #Mengimport datasets dari
4     library sklearn
5 clf = svm.SVC() #Memasukkan implementasi dari "Support Vector
6     Classification" ke variabel clf
7 X, y = datasets.load_iris(return_X_y=True) #Untuk memuat dan
8     memasukkan dataset iris ke variabel bernama iris
9 clf.fit(X, y) #Untuk melakukan pengiriman data training set
10    ke method fit

```



Gambar 1.13 Hasil Percobaan 1

- Percobaan 2

```

1 import pickle #Mengimport pickle untuk implementasi protokol
   serializing dan de-serializing
2 s = pickle.dumps(clf) #Untuk serialize hirarki dari object
3 clf2 = pickle.loads(s) #Untuk memuat dan men de-serialize
   hirarki dari object
4 clf2.predict(X[0:1]) #Untuk melakukan prediksi nilai yang
   baru berdasarkan gambar terakhir dari digits.data

```

Gambar 1.14 Hasil Percobaan 2

- Percobaan 3

```
1 y[0] #Untuk menampilkan data iris koordinat y
```

Gambar 1.15 Hasil Percobaan 3

- Percobaan 4

```

1 from joblib import dump, load #Mengambil class dump dan load
   dari library joblib
2 dump(clf, 'filename.joblib') #Untuk memasukkan data ke dalam
   sebuah file

```

Gambar 1.16 Hasil Percobaan 4

- Percobaan 5

```
1 clf = load('filename.joblib') #Untuk memuat data dari file
```

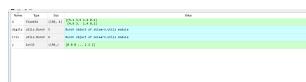
Gambar 1.17 Hasil Percobaan 5

- Full sample

```

1 # -*- coding: utf-8 -*-
2 """
3 Created on Thu Feb 27 10:37:49 2020
4
5 @author:Luthfi Muhammad Nabil
6 """
7
8 from sklearn import svm #Mengimport class SVM dari library
9      sklearn
10 from sklearn import datasets #Mengimport datasets dari
11      library sklearn
12 clf = svm.SVC() #Memasukkan implementasi dari "Support Vector
13      Classification" ke variabel clf
14 X, y = datasets.load_iris(return_X_y=True) #Untuk memuat dan
15      memasukkan dataset iris ke variabel bernama iris
16 clf.fit(X, y) #Untuk melakukan pengiriman data training set
17      ke method fit
18 #%%
19 import pickle #Mengimport pickle untuk implementasi protokol
20      serializing dan de-serializing
21 s = pickle.dumps(clf) #Untuk serialize hirarki dari object
22 clf2 = pickle.loads(s) #Untuk memuat dan men de-serialize
23      hirarki dari object
24 clf2.predict(X[0:1]) #Untuk melakukan prediksi nilai yang
25      baru berdasarkan gambar terakhir dari digits.data
26 #%%
27 y[0] #Untuk menampilkan data iris koordinat y
28 #%%
29 from joblib import dump, load #Mengambil class dump dan load
30      dari library joblib
31 dump(clf, 'filename.joblib') #Untuk memasukkan data ke dalam
32      sebuah file
33 #%%
34 clf = load('filename.joblib') #Untuk memuat data dari file

```



Gambar 1.18 Hasil pada variable explorer

1.1.9 Conventions

Seluruh metode akan dilakukan pengaturan untuk membuat tingkah laku lebih dapat diprediksi

- Percobaan 1

```

1 import numpy as np #Memuat library numpy yang akan
2      diinisialisasikan menjadi np
3 from sklearn import random_projection #Memuat class
4      random_projection dari library sklearn

```

```
4 rng = np.random.RandomState(0) #Memuat angka random dan  
mengekspos angka untuk menghasilkan dari berbagai  
probabilitas distribusi, angka tersebut dimasukkan ke  
variabel rng  
5 X = rng.rand(10, 2000) #Membatasi jarak angka random  
6 X = np.array(X, dtype='float32') #Memuat angka menjadi ke  
dalam array  
7 X.dtype #Mengambil tipe data dari variabel X
```

```
In [1]: ***
       Created on Thu Feb 27 16:16:10 2020
***
*** Author: Luthfi Hulhamel Hellii
***
***
import numpy as np # import library numpy yang akan digunakan untuk menjalankan operasi matematika
from sklearn import linear_model # import class linear_regression dari library sklearn
reg = linear_model.LinearRegression() # membuat objek reg dari kelas linear_regression
reg.fit(X,y) # fit atau membangun model dengan data X dan y
X = np.random.rand(100,1) # membuat array X dengan 100 data dan 1 variable
y = np.sin(X) + np.random.randn(100,1) # membuat array y dengan 100 data dan 1 variable
reg.predict(X) # prediksi nilai y dengan data X
```

Gambar 1.19 Hasil Percobaan 1

▪ Percobaan 2

```
1 transformer = random_projection.GaussianRandomProjection() #  
    Mengimplementasikan modul yang simpel dan efisien secara  
    komputasi untuk mengurangi dimensi dari data  
2 X_new = transformer.fit_transform(X) #Untuk membuat  
    penengahan data  
3 X_new.dtype #Mengambil tipe data dari variabel X
```

```
In [20]: transformer = random_projection.GaussianRandomProjection() #lengkap
data
...: X_new = transformer.fit_transform(X) #untuk membuat pengekangan data
...: X_new.dtype #mengambil tipe data dari variabel X
Out[20]: dtype('float64')
```

Gambar 1.20 Hasil Percobaan 2

▪ Percobaan 3

```
1 from sklearn import datasets #Mengimport datasets dari  
2 library sklearn  
3 from sklearn.svm import SVC  
4 iris = datasets.load_iris() #Untuk memuat dan memasukkan  
5 dataset iris ke variabel bernama iris  
6 clf = SVC() #Memasukkan implementasi dari "Support Vector  
7 Classification" ke variabel clf  
8 clf.fit(iris.data, iris.target) #Untuk melakukan pengiriman  
9 data training set ke method fit
```

```
[In 23]: from sklearn import datasets
iris=datasets.load_iris()
X=iris.data
y=iris.target
clf = SVC()
clf.fit(X,y)
print("Support Vector Classifiers")
print("Training set accuracy: ",clf.score(X,y))
print("Decision function shape: ",clf.decision_function(X).shape)
print("Decision function shape: ",clf.support_.shape)
print("Decision function shape: ",clf.support_vectors_.shape)
print("Decision function shape: ",clf.n_support_)
print("Decision function shape: ",clf.coef_.shape)
print("Decision function shape: ",clf.intercept_.shape)
print("Decision function shape: ",clf.cache_size)
print("Decision function shape: ",clf.degree_, "gamma=",clf.gamma,"coef0=",clf.coef_,"alpha_deprecated",clf.kernel,"rho", "max_iter=",clf.probability,"values", "random_state", "None", "shrinking", "True", "tol", "0.001", "verbose", "False")
```

Gambar 1.21 Hasil Percobaan 3

- Percobaan 4

```
1 list(clf.predict(iris.data[:3])) #Untuk memuat dan men de-
    serialize hirarki dari object , data tersebut akan
    dimasukkan ke fungsi list
```

Gambar 1.22 Hasil Percobaan 4

- Percobaan 5

```
1 clf.fit(iris.data, iris.target_names[iris.target]) #Untuk
    melakukan pengiriman data training set ke method fit
```

Gambar 1.23 Hasil Percobaan 5

- Percobaan 6

```
1 list(clf.predict(iris.data[:3])) #Untuk memuat dan men de-
    serialize hirarki dari object , data tersebut akan
    dimasukkan ke fungsi list
```

Gambar 1.24 Hasil Percobaan 6

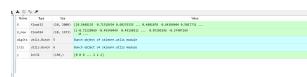
- Full sample

```
1 # -*- coding: utf-8 -*-
2 """
3 Created on Thu Feb 27 10:56:18 2020
4
5 @author: Luthfi Muhammad Nabil
```

```

6 """
7
8 import numpy as np #Memuat library numpy yang akan
9     diinisialisasikan menjadi np
9 from sklearn import random_projection #Memuat class
10    random_projection dari library sklearn
11
11 rng = np.random.RandomState(0) #Memuat angka random dan
12    mengeksplosi angka untuk menghasilkan dari berbagai
13    probabilitas distribusi, angka tersebut dimasukkan ke
14    variabel rng
12 X = rng.rand(10, 2000) #Membatasi jarak angka random
13 X = np.array(X, dtype='float32') #Memuat angka menjadi ke
14    dalam array
14 X.dtype #Mengambil tipe data dari variabel X
15 #%%
16 transformer = random_projection.GaussianRandomProjection() #
17     Mengimplementasikan modul yang simpel dan efisien secara
18     komputasi untuk mengurangi dimensi dari data
17 X_new = transformer.fit_transform(X) #Untuk membuat
18     penengahan data
18 X_new.dtype #Mengambil tipe data dari variabel X
19
20 #%%
21 from sklearn import datasets #Mengimport datasets dari
22     library sklearn
22 from sklearn.svm import SVC
23 iris = datasets.load_iris() #Untuk memuat dan memasukkan
24     dataset iris ke variabel bernama iris
24 clf = SVC() #Memasukkan implementasi dari "Support Vector
25     Classification" ke variabel clf
25 clf.fit(iris.data, iris.target) #Untuk melakukan pengiriman
26     data training set ke method fit
26 #%%
27 list(clf.predict(iris.data[:3])) #Untuk memuat dan men de-
28     serialize hirarki dari object, data tersebut akan
29     dimasukkan ke fungsi list
28 #%%
29 clf.fit(iris.data, iris.target_names[iris.target]) #Untuk
30     melakukan pengiriman data training set ke method fit
30 #%%
31 list(clf.predict(iris.data[:3])) #Untuk memuat dan de-
32     serialize hirarki dari object, data tersebut akan
33     dimasukkan ke fumgsi list

```



Gambar 1.25 Hasil pada variable explorer

1.1.10 Skrinsut Error

Gambar 1.26 Hasil Percobaan 6

1.1.11 Kode error dan jenis error tersebut

Error yang didapat berjenis name error, karena sebuah variabel tidak didefinisikan. Yaitu digits

```
1 # -*- coding: utf-8 -*-
2 """
3 Created on Thu Feb 27 10:56:18 2020
4
5 @author: Luthfi Muhammad Nabil
6 """
7
8 import numpy as np #Memuat library numpy yang akan
9     diinisialisasikan menjadi np
10 from sklearn import random_projection #Memuat class
11     random_projection dari library sklearn
12
13 rng = np.random.RandomState(0) #Memuat angka random dan
14     mengekspos angka untuk menghasilkan dari berbagai
15     probabilitas distribusi, angka tersebut dimasukkan ke
16     variabel rng
17 X = rng.rand(10, 2000) #Membatasi jarak angka random
18 X = np.array(X, dtype='float32') #Memuat angka menjadi ke dalam
19     array
20 X.dtype #Mengambil tipe data dari variabel X
21 #%%
22 transformer = random_projection.GaussianRandomProjection() #
23     Mengimplementasikan modul yang simpel dan efisien secara
24     komputasi untuk mengurangi dimensi dari data
25 X_new = transformer.fit_transform(X) #Untuk membuat penengahan
26     data
27 X_new.dtype #Mengambil tipe data dari variabel X
28
29 #%%
30 from sklearn import datasets #Mengimport datasets dari library
31     sklearn
32 from sklearn.svm import SVC
33 iris = datasets.load_iris() #Untuk memuat dan memasukkan dataset
34     iris ke variabel bernama iris
35 clf = SVC() #Memasukkan implementasi dari "Support Vector
36     Classification" ke variabel clf
37 clf.fit(iris.data, iris.target) #Untuk melakukan pengiriman data
38     training set ke method fit
39 #%%
```

```

27 list(clf.predict(iris.data[:3])) #Untuk memuat dan men de-
    serialize hirarki dari object, data tersebut akan dimasukkan
    ke fungsi list
28 #%%
29 clf.fit(iris.data, iris.target_names[iris.target]) #Untuk
    melakukan pengiriman data training set ke method fit
30 #%%
31 list(clf.predict(iris.data[:3])) #Untuk memuat dan men de-
    serialize hirarki dari object, data tersebut akan dimasukkan
    ke fungsi list

```

1.1.12 Penanganan Error

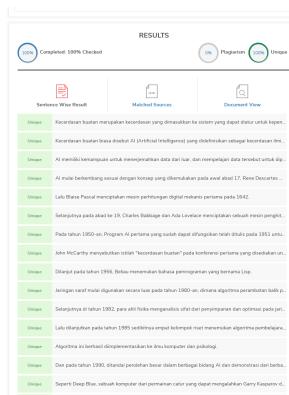
Untuk menangani error tersebut, bisa ditambahkan sesuai instruksi. Yaitu menambahkan sebuah variabel bernama digits. Selain itu, digits harus dapat bekerja sebagaimana mestinya. Berikut full kodingnya :

```

1 # -*- coding: utf-8 -*-
2 """
3 Created on Thu Feb 27 10:32:23 2020
4
5 @author: Luthfi Muhammad Nabil
6 """
7
8 from sklearn import svm #Untuk mengimport class sv dari library
    sklearn
9 from sklearn import datasets #Untuk import class/fungsi dataset
    dari scikit-learn library
10 clf = svm.SVC(gamma=0.001, C=100) #Memasukkan implementasi dari "
        Support Vector Classification" ke variabel clf
11 iris = datasets.load_iris() #Untuk memuat dan memasukkan dataset
    iris ke variabel bernama iris
12 #%%
13 digits = datasets.load_digits() #Untuk memuat dan memasukkan
    dataset digits ke variabel digits
14 clf.fit(digits.data[:-1], digits.target[:-1]) #Untuk melakukan
    pengiriman data training set ke method fit
15 #%%
16 clf.predict(digits.data[-1:]) #Untuk melakukan prediksi nilai
    yang baru berdasarkan gambar terakhir dari digits.data

```

1.1.13 Plagiarisme



Gambar 1.27 Hasil pada variable explorer

1.2 1174040 - Hagan Rowlenstino A. S

1.2.1 Teori

1.2.1.1 Definisi Kecerdasan Buatan Artificial Intelligence atau dapat disebut juga dengan kecerdasan buatan merupakan kecerdasan yang ditambahkan kepada suatu sistem yang dapat diatur dalam konteks ilmiah. Michael Haenlein dan Andreas Kaplan mendefinisikan bahwa AI adalah “sistem yang mempunyai kemampuan untuk menguraikan, belajar agar dapat digunakan untuk mencapai tujuan dengan melalui adaptasi yang fleksibel”. Kecerdasan ini dibuat dan dimasukkan ke dalam mesin agar dapat melakukan pekerjaan seperti yang dapat dilakukan manusia dengan cepat dan tepat. Bidang-bidang yang menggunakan kecerdasan buatan antara lain logika fuzzy, games, sistem pakar, jaringan saraf tiruan dan robotika.

1.2.1.2 Sejarah Kecerdasan Buatan Sejarah kecerdasan buatan ini dimulai dari zaman kuno, mitos ataupun cerita dan desas-desus tentang sebuah makhluk yang mempunyai kecerdasan serta kesadaran yang diberikan oleh pengrajin. Benih - benih nya mulai ditanam oleh para filsuf klasik yang mencari cara untuk menggambarkan proses berfikir manusia sebagai manipulasi simbol secara mekanis yang memuncak pada penemuan komputer digital di tahun 1940-an, yaitu sebuah mesin yang didasarkan penalaran matematika. Istilah kecerdasan buatan sendiri baru muncul pada tahun 1956, dan teori -teori nya sudah muncul sejak tahun 1941.

1.2.1.3 Perkembangan Kecerdasan Buatan

1. Perkembangan kecerdasan buatan dimulai dari Era Komputer Elektronik pada tahun 1941. dimana ditemukannya alat penyimpanan dan pemros-

esan informasi. Dilanjutkan pada tahun 1949, berhasilnya pembuatan komputer yang mampu menyimpan program yang memudahkan pekerjaan dalam memasukkan program menjadi lebih mudah.

2. Masa - masa persiapan AI terjadi pada tahun 1943 - 1956.
3. Awal Perkembangan AI terjadi pada 1952 - 1969
4. Perkembangan kecerdasan buatan melambat pada tahun 1966-1974
5. Sistem Berbasis Pengetahuan pada tahun 1969-1979
6. Kecerdasan Buatan menjadi sebuah industri pada tahun 1980-1988. Kembarinya Jaringan Syaraf tiruan pada tahun 1986-sekarang.

1.2.2 Instalasi

Membuka <https://scikit-learn.org/stable/tutorial/basic/tutorial.html> lalu mencobanya.

1.2.2.1 Instalasi library scikit, mencoba kompilasi dan ujicoba contoh kode

1. Buka anaconda prompt lalu ketikkan "pip install -U scikit-learn" untuk menginstall library scikit



Gambar 1.28 Install Library Scikit

2. Pilih salah satu example dari website tersebut lalu jalankan

```

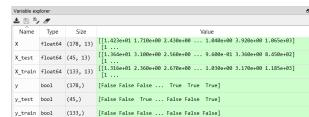
1 # -*- coding: utf-8 -*-
2 """
3 Created on Wed Feb 26 18:16:44 2020
4
5 @author: User
6 """
7
8 import matplotlib.pyplot as plt
9 from sklearn.svm import SVC
10 from sklearn.ensemble import RandomForestClassifier
11 from sklearn.metrics import plot_roc_curve
12 from sklearn.datasets import load_wine
13 from sklearn.model_selection import train_test_split
  
```

```

14
15 X, y = load_wine(return_X_y=True)
16 y = y == 2
17
18 X_train, X_test, y_train, y_test = train_test_split(X, y,
19 random_state=42)
20 svc = SVC(random_state=42)
21 svc.fit(X_train, y_train)

```

3. buka variable explolernya



Gambar 1.29 Variable Exploler

1.2.2.2 Mencoba loading an example dataset

1. mengambil data iris dan digit dari dataset

```

1 from sklearn import datasets #untuk mengimport dataset dari
    library learn-scikit
2 iris = datasets.load_iris() #membuat sebuah variable iris
    yang mempunyai isi yaitu dataset iris
3 digits = datasets.load_digits() #membuat sebuah variable
    digits yang mempunyai isi yaitu dataset digits

```

2. Menampilkan data digits

```

1 print(digits.data) #memberikan akses ke fitur yang dapat
    digunakan untuk mengklasifikasikan sampel digit dan
    menampilkannya di console

```

```

In [21]: runfile('D:/Semester6/AI/Chapter1/no2.py', wdir='D:/Semester6/AI/Chapter1')
[[ 0.,  0.,  5., ...,  0.,  0.]
 [ 0.,  0.,  10., ...,  0.,  0.]
 [ 0.,  0.,  15., ...,  0.,  0.]
 ...
 [ 0.,  0.,  1., ...,  5.,  0.]
 [ 0.,  0.,  2., ..., 12.,  0.]
 [ 0.,  0.,  0., ..., 12.,  1.]]

```

Gambar 1.30 Data Digits

3. menampilkan digits.target

```

1 digits.target #memberikan informasi tentang data yang
    berhubungan atau juga dapat dijadikan sebagai label

```

```
In [16]: digits.target
Out[16]: array([0, 1, 2, ..., 8, 9, 8])
```

Gambar 1.31 Digits Target

4. menampilkan data bentuk 2D.

```
1 digits.images[0] #Data selalu berupa array 2D, shape (
    n_samples, n_features), meskipun data aslinya mungkin
    memiliki bentuk yang berbeda.
```

```
In [17]: digits.images[0]
Out[17]:
array([[0., 0., 5., 13., 9., 1., 0., 0.],
       [0., 0., 13., 15., 10., 15., 5., 0.],
       [0., 3., 15., 2., 0., 11., 8., 0.],
       [0., 1., 11., 13., 10., 12., 7., 0.],
       [0., 5., 8., 0., 0., 9., 8., 0.],
       [0., 4., 11., 0., 1., 12., 7., 0.],
       [0., 2., 14., 5., 10., 12., 0., 0.],
       [0., 0., 6., 13., 10., 0., 0., 0.]])
```

Gambar 1.32 Data 2D

1.2.2.3 Mencoba Learning and Predicting

```
1 # -*- coding: utf-8 -*-
2 """
3 Created on Thu Feb 27 22:20:55 2020
4
5 @author: User
6 """
7
8 from sklearn.linear_model import LogisticRegression #untuk
    mengimport linear-model dari library sklearn
9 from sklearn.datasets import make_blobs #untuk mengimport library
    datasets dari sklearn
10
11 X, y = make_blobs(n_samples=100, centers=2, n_features=2,
    random_state=1) # untuk generate dataset dengan klasifikasi 2
    D
12 model = LogisticRegression() #menggunakan metode loginstic
    regression
13 model.fit(X, y)
14
15 Xnew, _ = make_blobs(n_samples=3, centers=2, n_features=2,
    random_state=1) # menentukan 1 buah contoh baru dimana
    jawabannya tidak diketahui
16
17 ynew = model.predict_proba(Xnew) # membuat sebuah prediksi dan
    memasukkan nya kedalam variable ynew
18 for i in range(len(Xnew)):
19     print("X=%s, Predicted=%s" % (Xnew[i], ynew[i])) #menampilkan
        hasil prediksi
```

1.2.2.4 Mencoba Model Persistence

```

1 # -*- coding: utf-8 -*-
2 """
3 Created on Thu Feb 27 22:27:13 2020
4
5 @author: User
6 """
7
8 from sklearn import svm #menigimport svm dari library sklearn
9 from sklearn import datasets #menigimport datasets dari library
10    sklearn
11 clf = svm.SVC() #menggunakan method SVC
12 iris = datasets.load_iris() #menggunakan dataset iris
13 X, y = iris.data, iris.target #memasukkan x sebagai iris data ,
14      dan y sebagai iris target
15 clf.fit(X, y) #laalu menggunakan metod fit .

```

1.2.2.5 Mencoba Conventions

```

1 # -*- coding: utf-8 -*-
2 """
3 Created on Thu Feb 27 23:24:21 2020
4
5 @author: User
6 """
7
8 import numpy as npy #mengimport numpy sebagai npy
9 from sklearn import random_projection #mengimport
10    random_projection dari library sklearn
11 rng = npy.random.RandomState(0) #Menggunakan fungsi random dari
12      numpy
13 X = rng.rand(10, 2000) #membuat range random diantara 10 sampai
14      2000
15 X = npy.array(X, dtype='float32') #yang dijadikan array dengan
16      tipe data float32
17 X.dtype

```

1.2.3 Penanganan Error



ValueError: Found array with 0 sample(s) (shape=(0, 2)) while a minimum of 1 is required.

Gambar 1.33 Data 2D

1.2.3.1 Screenshot Error

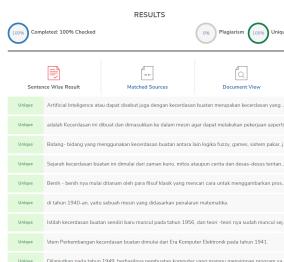
1.2.3.2 Kode error dan jenis error

Jenis errornya adalah value error

```
1 Xnew, _ = make_blobs(n_samples=3, centers=2, n_features=2,
    random_state=1) # menentukan 1 buah contoh baru dimana
    jawabannya tidak diketahui
```

1.2.3.3 Solusi Error Solusinya adalah dengan mengganti nilai n_samples nya agar tidak 0

1.2.4 Cek Plagiarism



Gambar 1.34 Cek Plagiarism

1.3 1174042 Faisal Najib Abdullah

1.3.1 Teori

- Definisi, sejarah, dan perkembangan kecerdasan buatan.

Definisi kecerdasan buatan adalah suatu pengetahuan yang dapat membuat komputer untuk meniru kecerdasan manusia yang berhubungan dengan penangkapan, pemodelan, dan penyimpanan kecerdasan manusia dalam sebuah sistem teknologi. Contohnya yaitu melakukan analisa penalaran untuk mengambil suatu kesimpulan atau penerjemahan atau keputusan dari satu bahasa satu ke bahasa lain.

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

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

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

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

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

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

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

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

1.3.2 Instalasi

1.3.2.1 Instalasi Library Scikit dari Pycharm

Masuk pada menu settings terus pilih Project Interpreter kemudian tambah library lalu cari dan install scikit

Mencoba Library

```

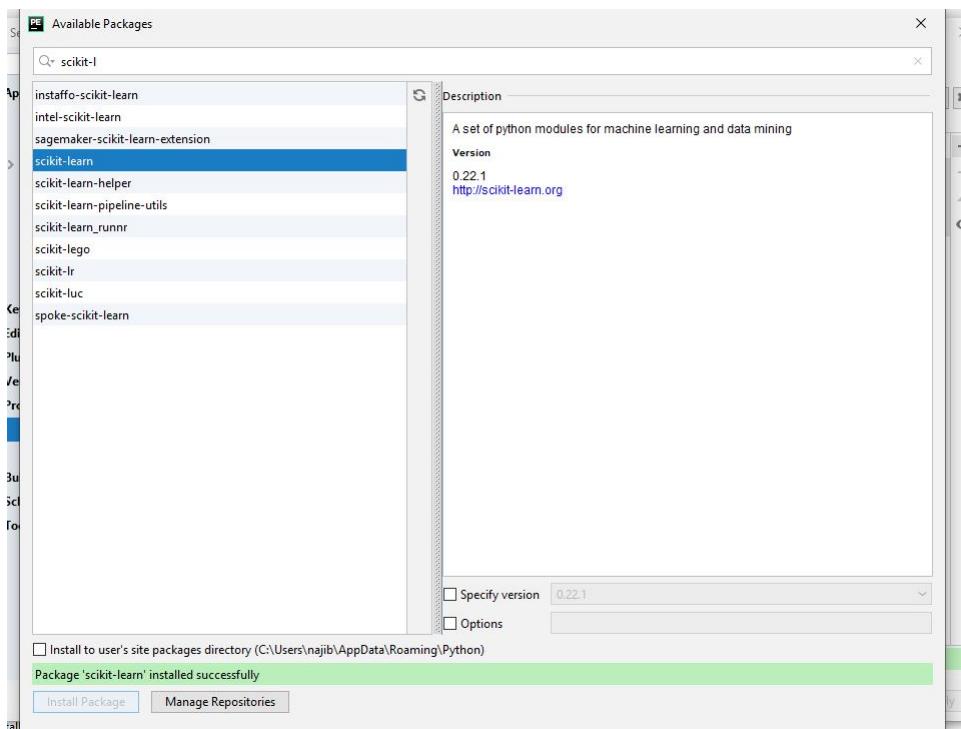
1 from sklearn import datasets
2 #pada baris ini merupakan sebuah perintah untuk mengimport class
   datasets dari packaged sklearn
3 iris = datasets.load_iris()
4 #pada baris kedua ini dimana iris merupakan suatu estimator/
   parameter yang berfungsi untuk mengambil data pada item
   datasets.load_iris
5 digits = datasets.load_digits()
6 #pada baris ketiga ini dimana digits merupakan suatu estimator/
   parameter yang berfungsi untuk mengambil data pada item
   datasets.load_digits

```

```

1 from sklearn import datasets
2 #pada baris ini merupakan sebuah perintah untuk mengimport class
   datasets dari packaged sklearn
3 iris = datasets.load_iris()
4 #pada baris kedua ini dimana iris merupakan suatu estimator/
   parameter yang berfungsi untuk mengambil data pada item
   datasets.load_iris

```



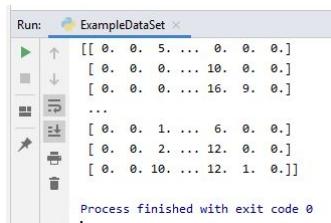
Gambar 1.35 Installasi

```

5 digits = datasets.load_digits()
6 #pada baris ketiga ini dimana digits merupakan suatu estimator/
  # parameter yang berfungsi untuk mengambil data pada item
  # datasets.load_digits
7 print(digits.data)
8 #pada baris keempat ini merupakan perintah yang berfungsi untuk
  # menampilkan estimator/parameter yang dipanggil pada item
  # digits.data dan menampilkan outputannya
9 digits.target
10 #barisan ini untuk mengambil target pada estimator/parameter
    # digits dan menampilkan outputannya
11 digits.images[0]
12 #barisan ini untuk mengambil images[0] pada estimator/parameter
    # digits dan menampilkan outputannya

from sklearn import svm
#pada baris ini merupakan sebuah perintah untuk mengimport class
# svm dari packaged sklearn
clf = svm.SVC(gamma=0.001, C=100.)

```

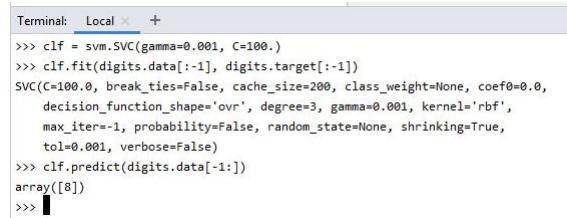


Gambar 1.36 Mencoba Loading an example Dataset

```

4 #pada baris kedua ini clf sebagai estimator/parameter , svm.SVC
   sebagai class , gamma sebagai parameter untuk menetapkan nilai
   secara manual
5 clf.fit(digits.data[:-1], digits.target[:-1])
6 #pada baris ketiga ini clf sebagai estimator/parameter , fit
   sebagai metode , digits.data sebagai item , [-1] sebagai
   syntax pythonnya dan menampilkan outputannya
7 clf.predict(digits.data[-1:])
8 #pada baris terakhir ini clf sebagai estimator/parameter , predict
   sebagai metode lainnya , digits.data sebagai item dan
   menampilkan outputannya

```



Gambar 1.37 Learning and Predicting

```

1 from sklearn import svm
2 #pada baris ini merupakan sebuah perintah untuk mengimport class
   svm dari packaged sklearn
3 from sklearn import datasets
4 #pada baris ini merupakan sebuah perintah untuk mengimport class
   datasets dari packaged sklearn
5 clf = svm.SVC(gamma='scale')
6 #pada baris ketiga ini clf sebagai estimator/parameter , svm.SVC
   sebagai class , gamma sebagai parameter untuk menetapkan nilai
   secara manual dengan nilai scale
7 iris = datasets.load_iris()
8 #pada baris keempat ini iris sebagai estimator/parameter ,
   datasets.load_iris() sebagai item dari suatu nilai
9 X, y = iris.data, iris.target
10 #pada baris kelima ini X, y sebagai estimator/parameter , iris.
    data, iris.target sebagai item dari 2 nilai yang ada

```

```

11 clf.fit(X, y)
12 #pada baris keenam ini clf sebagai estimator/parameter dengan
   menggunakan metode fit untuk memanggil estimator X, y dengan
   outputannya
13
14
15 import pickle
16 #pickle merupakan sebuah class yang di import
17 s = pickle.dumps(clf)
18 #pada baris ini s sebagai estimator/parameter dengan pickle.dumps
   merupakan suatu nilai/item dari estimator/parameter clf
19 clf2 = pickle.loads(s)
20 #pada baris ini clf2 sebagai estimator/parameter, pickle.loads
   sebagai suatu item, dan s sebagai estimator/parameter yang
   dipanggil
21 clf2.predict(X[0:1])
22 #pada baris ini clf2.predict sebagai suatu item dengan
   menggunakan metode predict untuk menentukan suatu nilai dari
   (X[0:1])
23 y[0]
24 #pada estimator/parameter y berapapun angka yang diganti nilainya
   akan selalu konstan yaitu 0
25
26
27 from joblib import dump, load
28 #pada baris berikut ini merupakan sebuah perintah untuk
   mengimport class dump, load dari packaged joblib
29 dump(clf, 'filename.joblib')
30 #pada baris berikutnya dump di sini sebagai class yang didalamnya
   terdapat nilai dari suatu item clf dan data joblib
31 clf = load('filename.joblib')
32 #pada baris terakhir clf sebagai estimator/parameter dengan suatu
   nilai load berfungsi untuk mengulang data sebelumnya

```

```

... clf.fit(X, y)
SVC(C=100.0, break_ties=False, 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)

```

Gambar 1.38 Model Presistence

```

>>> clf2.predict(X[0:1])
array([0])
>>> #pada baris ini clf2.predict
... y[0]
0

```

Gambar 1.39 Model Presistence

```

1 #Type Casting
2 from sklearn import svm

```

The screenshot shows a Jupyter Notebook environment. On the left, there's a file tree with files like 2,4,1.JPG, 2,4.JPG, 2,4.py, 2.JPG, 3.JPG, ExampleDataSet.py, filename.joblib, Teori.tex, and External Libraries. The code cell on the right contains Python code for model persistence:

```

2,4,1.JPG 26 from joblib import dump, load
2,4.JPG 27 #pada baris berikut ini merupakan sel
2,4.py 28 dump(clf, 'filename.joblib')
2.JPG 29 #pada baris berikutnya dump di sini :
3.JPG 30 clf = load('filename.joblib')
ExampleDataSet.py 31 #pada baris terakhir clf sebagai est:
filename.joblib 32 #pada baris terakhir clf sebagai est:
Teori.tex 33 |

```

Gambar 1.40 Model Presistence

```

3 #pada baris ini merupakan sebuah perintah untuk mengimport class
   svm dari packaged sklearn
4 from sklearn import random_projection
5 #pada baris ini merupakan sebuah perintah untuk mengimport class
   random_projection dari packaged sklearn
6 rng = np.random.RandomState(0)
7 #rng sebagai estimator/parameter dengan nilai suatu itemnya yaitu
   np.random.RandomState(0)
8 X = rng.rand(10, 2000)
9 #X sebagai estimator/parameter dengan nilai item rng.rand
10 X = np.array(X, dtype='float32')
11 #X sebagai estimator/parameter dengan nilai item np.array
12 X.dtype
13 #X.dtype sebagai item pemanggil
14 transformer = random_projection.GaussianRandomProjection()
15 #transformer sebagai estimator/parameter dengan memanggil class
   random_projection
16 X_new = transformer.fit_transform(X)
17 #X new di sini sebagai estimator/parameter dan menggunakan metode
   fit
18 X_new.dtype
19 #X new.dtype sebagai item
20
21
22 from sklearn import datasets
23 #pada baris ini merupakan sebuah perintah untuk mengimport class
   datasets dari packaged sklearn
24 from sklearn.svm import SVC
25 #pada baris ini merupakan sebuah perintah untuk mengimport class
   SVC dari packaged sklearn.svm
26 iris = datasets.load_iris()
27 #iris sebagai estimator/parameter dengan item datasets.load_iris
   ()
28 clf = SVC(gamma='scale')
29 #clf sebagai estimator/parameter dengan nilai class SVC pada
   parameter gamma sebagai set penilaian
30 clf.fit(iris.data, iris.target)
31 #estimator/parameter clf menggunakan metode fit dengan itemnya
32 list(clf.predict(iris.data[:3]))
33 #menambahkan item list dengan metode predict
34 clf.fit(iris.data, iris.target_names[iris.target])
35 #estimator/parameter clf menggunakan metode fit dengan itemnya
36 list(clf.predict(iris.data[:3]))
37 #menambahkan item list dengan metode predict
38

```

```
39
40
41 #Refitting and Updating Parameters
42 import numpy as np
43 #pada baris ini merupakan sebuah perintah untuk mengimport class
44     svm dari np
45 from sklearn.svm import SVC
46 #pada baris ini merupakan sebuah perintah untuk mengimport class
47     SVC dari packaged sklearn.svm
48 rng = np.random.RandomState(0)
49 #rng sebagai estimator/parameter dengan nilai suatu itemnya yaitu
50     np.random.RandomState(0)
51 X = rng.rand(100, 10)
52 #X sebagai estimator/parameter dengan nilai item rng.rand
53 y = rng.binomial(1, 0.5, 100)
54 #y sebagai estimator/parameter dengan nilai item rng.binomial
55 X_test = rng.rand(5, 10)
56 #X test sebagai estimator/parameter dengan nilai item rng.rand
57 clf = SVC()
58 #clf sebagai estimator/parameter dan class SVC
59 clf.set_params(kernel='linear').fit(X, y)
60 #set params sebagai item
61 clf.predict(X_test)
62
63
64 #Multiclass vs. Multilabel Fitting
65 from sklearn.svm import SVC
66 #pada baris ini merupakan sebuah perintah untuk mengimport class
67     SVC dari packaged sklearn.svm
68 from sklearn.multiclass import OneVsRestClassifier
69 #pada baris ini merupakan sebuah perintah untuk mengimport class
70     OneVsRestClassifier dari packaged sklearn.multiclass
71 from sklearn.preprocessing import LabelBinarizer
72 #pada baris ini merupakan sebuah perintah untuk mengimport class
73     LabelBinarizer dari packaged sklearn.preprocessing
74 X = [[1, 2], [2, 4], [4, 5], [3, 2], [3, 1]]
75 y = [0, 0, 1, 1, 2]
76 classif = OneVsRestClassifier(estimator=SVC(gamma='scale',
77                                         random_state=0))
78 classif.fit(X, y).predict(X)
79 y = LabelBinarizer().fit_transform(y)
80 classif.fit(X, y).predict(X)
81
82
83 from sklearn.preprocessing import MultiLabelBinarizer
84 y = [[0, 1], [0, 2], [1, 3], [0, 2, 3], [2, 4]]
85 y = MultiLabelBinarizer().fit_transform(y)
86 classif.fit(X, y).predict(X)
```

1.3.3 Penanganan eror

```
>>> from joblib import dump, load
>>> #pada baris berikut ini merupakan sebuah perintah untuk mengimport class dump, load dari packaged joblib
... dump(clf, 'filename.joblib')
Traceback (most recent call last):
  File "<stdin>", line 2, in <module>
NameError: name 'clf' is not defined
>>> #pada baris berikutnya dump di sini sebagai class yang didalamnya terdapat nilai dari suatu item clf dan data joblib
... clf = load('filename.joblib')
```

Gambar 1.41 Error

1.3.3.1 ScreenShoot Eror

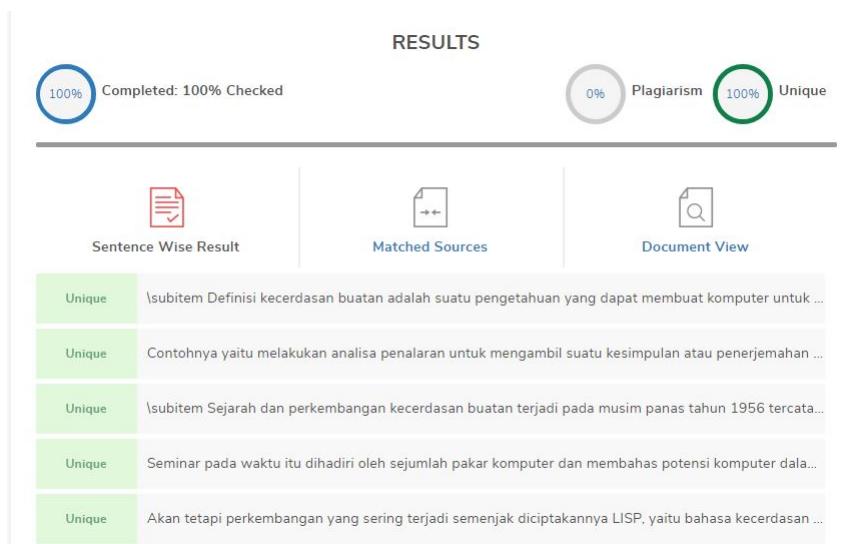
1.3.3.2 Tuliskan Kode Eror dan Jenis Erornya

File "<stdin>", line 2, in <module>
NameError: name 'clf' is not defined

1.3.3.3 Solusi Pemecahan Masalah Error

Ini karna kode di jalankan perbaris perbaris, jika kode dijanlankan bersamaan makan kode berjan sesuai prosedur.

1.3.4 Plagiat



Gambar 1.42 Error

1.3.5 Link

[Youtube](#)

1.4 1174043 - Irwan Rizkiansyah

1.4.1 Definisi Kecerdasan Buatan

Kecerdasan buatan atau biasa disebut AI (Artificial Intelligence) merupakan kecerdasan yang dibuat dan ditambahkan oleh manusia ke suatu sistem teknologi, yang diatur dan dikembangkan dalam konteks ilmiah, yang merupakan bentukan dari kecerdasan entitas ilmiah yang ada. Jadi pada intinya definisi AI dapat terus dikembangkan, namun yang menjadi poin utamanya adalah bagaimana manusia menciptakan sebuah teknologi yang dapat berpikir seperti selayaknya manusia itu sendiri.

1.4.2 Sejarah dan Perkembangan

- Program kecerdasan buatan atau Artificial Intelligence pertama kali diceritakan pada kisaran tahun 1951. Tidak bisa dipungkiri bahwa di tahun tersebut memang sedang gencar-gencarnya pembuatan cikal bakal, konsep, hingga teknologi berbasis AI. Dan, AI sendiri pertama kali digunakan di University of Manchester untuk menjalankan sebuah mesin bernama Ferranti Mark 1.
- Beberapa waktu kemudian, Christopher Strachey melanjutkan konsep kecerdasan buatan untuk menjalankan sebuah permainan catur, dimana bidak catur tersebut dapat berjalan secara otomatis dan mampu bermain melawan manusia sungguhan.
- Berlanjut pada tahun 1956, kecerdasan buatan tidak hanya dibuat untuk memudahkan bermain catur saja. Melainkan pada saat konferensi pertamanya, John McCharty menamai algoritma teknologi tersebut dengan sebutan “Artificial Intelligence”. Istilah tersebut masih digunakan hingga sekarang oleh para pakar teknologi.
- Terakhir, konsep dan teknologi kecerdasan buatan disempurnakan oleh seorang ahli yang namanya masih diingat sampai sekarang sebagai seorang pakar kecerdasan buatan, yaitu Alan Turin. Pada saat itu, Alan Turin meneliti dan menguji coba algoritma AI yang diberi nama dengan “Turing Test”. Hingga seiring berkembangnya waktu, konsep teknologi AI banyak digunakan di berbagai teknologi baik itu multimedia, search engine, dan masih banyak lainnya.

1.4.3 Supervised Learning

Supervised learning (Pembelajaran terawasi), dalam konteks kecerdasan buatan (AI) dan Machine Learning, adalah jenis sistem di mana input dan output data yang diinginkan disediakan. Input dan output data diberi label

untuk klasifikasi untuk memberikan dasar pembelajaran untuk pemrosesan data di masa depan.

1.4.4 Unsupervised Learning

Unsupervised learning merupakan pembelajaran yang tidak terawasi dimana tidak memerlukan target output. Pada metode ini tidak dapat ditentukan hasil seperti apa yang diharapkan selama proses pembelajaran, nilai bobot yang disusun dalam proses range tertentu tergantung pada nilai output yang diberikan. Tujuan metode unsupervised learning ini agar kita dapat mengelompokkan unit-unit yang hampir sama dalam satu area tertentu.

1.4.5 Teknik Klasifikasi

Teknik klasifikasi memprediksi respons diskrit, misalnya seperti apakah email itu asli atau spam, atau apakah tumor itu kanker atau tidak. Model klasifikasi mengklasifikasikan data masukan ke dalam kategori tersebut.

1.4.6 Regresi

Regresi yaitu pengeluaran nilai output yang konstan jika dipicu dengan parameter tertentu biasanya regresi disini berbentuk regresi linier. Regresi linier yaitu metode statistika yang digunakan untuk membentuk model hubungan antara variabel terikat(dependen,respon,Y) dengan satu atau lebih variabel bebas(independent, prdiktor, X). Apabila banyaknya variabel bebas hanya ada satu, disebut sebagai regresi linier sederhana, sedangkan apabila terdapat lebih dari satu variabel bebas, disebut sebagai regresi linier berganda.

1.4.7 Training Set

Training set adalah bagian dari dataset itu sendiri yang dilatih untuk membuat prediksi atau algoritma mesin learning lainnya sesuai keinginan atau tujuan data itu dibuat.

1.4.8 Testing Set

Testing set adalah bagian dari dataset yang di tes atau diujicoba untuk melihat keakuratannya dengan katalain melihat peformanya.

1.4.9 Instalasi dan Percobaan Kompilasi dari Library Scikit-learn

1. Buka anaconda prompt
2. kemudian Ketik di anaconda prompt yaitu : "pip install -U scikit-learn"



Gambar 1.43 Instalasi Scikit Learn

3. Setelah selesai instalasi, pilih salah satu example dari website Scikit
 4. Sample kode

```
1 print(__doc__)
2
3 # Author: Nelle Varoquaux <nelle.varoquaux@gmail.com>
4 #          Alexandre Gramfort <alexandre.gramfort@inria.fr>
5 # License: BSD
6
7 import numpy as np
8 import matplotlib.pyplot as plt
9 from matplotlib.collections import LineCollection
10
11 from sklearn.linear_model import LinearRegression
12 from sklearn.isotonic import IsotonicRegression
13 from sklearn.utils import check_random_state
14
15 n = 100
16 x = np.arange(n)
17 rs = check_random_state(0)
18 y = rs.randint(-50, 50, size=(n,)) + 50. * np.log1p(np.arange(
19     n))
20 #
21 ######
22
23 # Fit IsotonicRegression and LinearRegression models
24
25 ir = IsotonicRegression()
26
27 y_ = ir.fit_transform(x, y)
28
29 lr = LinearRegression()
30 lr.fit(x[:, np.newaxis], y) # x needs to be 2d for
31 # LinearRegression
32 #
33 ######
34
35 # Plot result
36
37 segments = [[i, y[i]], [i, y_[i]]] for i in range(n)]
38 lc = LineCollection(segments, zorder=0)
39 lc.set_array(np.ones(len(y)))
40 lc.set_linewidths(np.full(n, 0.5))
41
42 fig = plt.figure()
```

```
39 plt.plot(x, y, 'r.', markersize=12)
40 plt.plot(x, y_, 'b.-', markersize=12)
41 plt.plot(x, lr.predict(x[:, np.newaxis]), 'b-')
42 plt.gca().add_collection(lc)
43 plt.legend(['Data', 'Isotonic Fit', 'Linear Fit'], loc='lower
        right')
44 plt.title('Isotonic regression')
45 plt.show()
```

5. Kemudian jalankan aplikasi tersebut, dan bisa dicek hasil dari Variable explorernya

Gambar 1.44 Variable Explorer

1.4.10 Mencoba Loading an example dataset


```
1 # -*- coding: utf-8 -*-
2 """
3 Created on Wed Feb 26 19:32:36 2020
4
5 @author: lenovo
6 """
7
8 from sklearn import datasets #import class/fungsi dataset
9         dari scikit-learn library
10
11 import matplotlib.pyplot as plt #import matplotlib
12
13 #Load the digits dataset
14 digits = datasets.load_digits() #memuat dan memasukkan
15         dataset digits ke variabel digits
16
17 #menampilkan digit pertama
18 plt.figure(1, figsize=(3, 3))
19 plt.imshow(digits.images[-1], cmap=plt.cm.gray_r ,
20             interpolation='nearest')
21 plt.show()
```

2. Kemudian jalankan aplikasi tersebut



Gambar 1.45 Dataset

1.4.11 Mencoba Learning and Predicting

| | |
|-----------|------|
| 1. Sample | kode |
|-----------|------|

```

1 # -*- coding: utf-8 -*-
2 """
3 Created on Thu Feb 27 19:07:08 2020
4
5 @author: lenovo
6 """
7
8 from sklearn.linear_model import LogisticRegression #import
9         dari library sklearn
10 from sklearn.datasets import make_blobs #import dari library
11         sklearn
12
13 X, y = make_blobs(n_samples=100, centers=2, n_features=2,
14                     random_state=1) # generate dataset klasifikasi 2d
15
16 # fit final model
17 model = LogisticRegression()
18 model.fit(X, y)
19
20 Xnew, _ = make_blobs(n_samples=3, centers=2, n_features=2,
21                     random_state=1) # menentukan 1 buah contoh baru dimana
22                     jawabannya tidak diketahui
23
24 ynew = model.predict_proba(Xnew) # membuat prediksi
25 for i in range(len(Xnew)):
26     print("X=%s, Predicted=%s" % (Xnew[i], ynew[i])) #
27             menampilkan hasil prediksi

```

| | | | |
|-------------|----------|----------|----------|
| 2. Kemudian | jalankan | aplikasi | tersebut |
|-------------|----------|----------|----------|

```

In [18]: runfile('C:/Users/lenovo/Desktop/simple_3.py', wdir='C:/Users/lenovo/Desktop')
X=[-0.3520228  2.38095117], Predicted:[0.9997159  0.0002840]
X=[-0.2529000  2.38095117], Predicted:[0.9997159  0.0002840]
X=[-2.18771168  3.3332125], Predicted:[0.90389073  0.09610927]

```

Gambar 1.46 Predicting

1.4.12 Mencoba Model Persistence

| | |
|--------|------|
| Sample | kode |
|--------|------|

```

1 # -*- coding: utf-8 -*-

```

```

2 """
3 Created on Thu Feb 27 19:36:22 2020
4
5 @author: lenovo
6 """

7
8 from sklearn import svm #import dari library sklearn
9 from sklearn import datasets #import dari library sklearn
10 clf = svm.SVC() #menggunakan Support Vector Classifier
11 iris = datasets.load_iris() #menggunakan iris dataset
12 X, y = iris.data, iris.target
13 clf.fit(X, y)

```

1.4.13 Mencoba Conventions

Sample kode

```

1 # -*- coding: utf-8 -*-
2 """
3 Created on Thu Feb 27 20:04:46 2020
4
5 @author: lenovo
6 """

7
8 import numpy as np #import dari library sklearn
9 from sklearn import random_projection #import dari library
   sklearn
10
11 rng = np.random.RandomState(0) #Menggunakan fungsi random
12 X = rng.rand(10, 2000) #mengambil nilai random
13 X = np.array(X, dtype='float32') #membuat array bertipe float32
14 X.dtype

```

1.5 1174050 Dika Sukma Pradana

1.5.1 Teori

1. Definisi, sejarah, dan perkembangan kecerdasan buatan.

Intelejenji buatan (AI) mengacu pada simulasi kecerdasan manusia dalam mesin yang diprogram untuk berpikir seperti manusia dan meniru tindakan mereka. Istilah ini juga dapat diterapkan pada mesin apa pun yang menunjukkan sifat-sifat yang terkait dengan pikiran manusia seperti pembelajaran dan pemecahan masalah.

Sejarah Kecerdasan Buatan (AI) bermula pada saat zaman kuno, dengan sejuta mitos, cerita dan desas-desus tentang makhluk buatan yang diberkahii dengan kecerdasan oleh pengrajin ahli. Benih-benih AI modern ditanam oleh para filsuf klasik yang mencoba menggam-

barkan proses pemikiran manusia sebagai manipulasi simbol secara mekanis. Karya ini memuncak dalam penemuan komputer digital yang dapat diprogram pada tahun 1940-an, sebuah mesin yang didasarkan pada esensi abstrak penalaran matematika. Perangkat ini dan ide-ide di belakangnya menginspirasi segenap ilmuwan untuk mulai serius membahas kemungkinan membangun otak elektronik. Bidang penelitian AI didirikan pada lokakarya yang diadakan di kampus Dartmouth College selama musim panas 1956. Mereka yang hadir akan menjadi pemimpin penelitian AI selama beberapa dekade. Banyak dari mereka meramalkan bahwa sebuah mesin yang secerdas manusia akan hidup tidak lebih dari satu generasi dan mereka diberi jutaan dolar untuk mewujudkan visi atau tujuan ini. Akhirnya, menjadi jelas bahwa mereka meremehkan kesulitan proyek. Pada tahun 1973, sebagai tanggapan atas kritik dari James Lighthill dan tekanan terus-menerus dari kongres, AS dan Pemerintah Inggris menghentikan pendanaan penelitian yang tidak diarahkan pada kecerdasan buatan, dan tahun-tahun sulit berikutnya akan dikenal sebagai musim dingin AI. Tujuh tahun kemudian, sebuah inisiatif visioner oleh Pemerintah Jepang mengilhami pemerintah dan industri untuk menyediakan miliaran dolar dalam AI, tetapi pada akhir 80-an investor menjadi kecewa dengan kurangnya daya komputer (perangkat keras) yang dibutuhkan dan menarik lebih banyak dana. Investasi dan minat pada AI tumbuh pesat pada dekade pertama abad ke-21, ketika pembelajaran mesin berhasil diterapkan pada banyak masalah di dunia akademis dan industri karena metode baru, penerapan perangkat keras komputer yang kuat, dan kumpulan data yang sangat besar.

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

Supervised learning adalah sebuah metode pendekatan yang mana sudah terdapat data yang dilatih, dan terdapat variabel yang ditargetkan atau yang menjadi tujuan sehingga tujuan dari pendekatan ini adalah mengelompokan suatu data ke data yang sudah ada. Sedangkan unsupervised learning tidak memiliki data latih, sehingga dari data yang ada, kita mengelompokan data tersebut menjadi dua bagian atau bahkan tiga bagian dan seterusnya.

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

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

Data set merupakan sebuah cabang aplikasi dari Artificial Intelligence(AI)/Kecerdasan Buatan yang terfokus kepada pengembangan se-

b. buah sistem yang mampu belajar sendiri tanpa harus berulang kali di program oleh manusia(programmer).

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

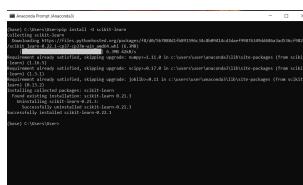
Testing set digunakan untuk mengukur sejauh mana classifier berhasil melakukan klasifikasi dengan benar[?].

1.5.2 Instalasi

1.5.2.1 Instalasi Library Scikit dari Anaconda

1. Download aplikasi Anaconda terlebih dahulu.
 2. Install aplikasi Anaconda yang sudah di download tadi.
 3. Simpan aplikasi sesuai folder yang kita pilih lalu next.
 4. Centang Keduanya lalu tekan tombol install.
 5. Setelah itu tunggu sampai proses instalasi selesai lalu jika sudah tekan tombol finish.
 6. Lalu buka command prompt anda dan tuliskan perintah berikut ini untuk mengecek apakah aplikasinya sudah terinstall.

```
pip install -U scikit-learn
```
 7. Kemudian ketikkan perinta pip install -U scikit-learn seperti gambar berikut.



Gambar 1.47 Instalasi

1.5.3 Percobaan

Mencoba Library

```
1 # -*- coding: utf-8 -*-
2 """
3 Created on Wed Feb 26 22:51:53 2020
4
5 @author: User
6 """
```

```

7
8 import matplotlib.pyplot as plt
9 from sklearn.svm import SVC
10 from sklearn.ensemble import RandomForestClassifier
11 from sklearn.metrics import plot_roc_curve
12 from sklearn.datasets import load_wine
13 from sklearn.model_selection import train_test_split
14
15 X, y = load_wine(return_X_y=True)
16 y = y == 2
17
18 X_train, X_test, y_train, y_test = train_test_split(X, y,
19 random_state=42)
20 svc = SVC(random_state=42)
21 svc.fit(X_train, y_train)
```

SVC(random_state=42)

Gambar 1.48 Variabel Explore

```

1 from sklearn import datasets
2 #perintah untuk mengimport class datasets dari packaged sklearn
3 iris = datasets.load_iris()
4 #iris merupakan suatu estimator/parameter yang berfungsi untuk
5 #mengambil data pada item datasets.load_iris
5 digits = datasets.load_digits()
6 #digits merupakan suatu estimator/parameter yang berfungsi untuk
7 #mengambil data pada item datasets.load_digits
7 print(digits.data)
8 #perintah yang berfungsi untuk menampilkan estimator/parameter
9 #yang dipanggil pada item digits.data dan menampilkan
10 #outputannya
10 digits.target
11 #untuk mengambil target pada estimator/parameter digits dan
12 #menampilkan outputannya
11 digits.images[0]
12 #untuk mengambil images[0] pada estimator/parameter digits dan
13 #menampilkan outputannya
```

```
In [13]: runfile('C:/Users/User/Documents/Sayler/1.py', wdir='C:/Users/User/Documents/')
Spicer
[[ 0.,  0.,  5., ...,  0.,  0.,  0.],
 [ 0.,  0.,  0., ..., 10.,  0.,  0.],
 [ 0.,  0.,  0., ...,  0.,  0.,  0.],
 ...,
 [ 0.,  0.,  1., ...,  6.,  0.,  0.],
 [ 0.,  0.,  2., ..., 12.,  0.,  0.],
 [ 0.,  0.,  0., ..., 12.,  1.,  0.]]
```

Gambar 1.49 Datasets

```

1 # -*- coding: utf-8 -*-
2 """
3 Created on Wed Feb 26 22:17:01 2020
```

```
4
5 @author: User
6 """
7 from sklearn import svm
8 #merupakan sebuah perintah untuk mengimport class svm dari
9     packaged sklearn
10 # merupakan sebuah perintah untuk mengimport class datasets dari
11     packaged sklearn
12 clf = svm.SVC(gamma='scale')
13 #clf sebagai estimator/parameter, svm.SVC sebagai class , gamma
14     sebagai parameter untuk menetapkan nilai secara manual dengan
15     nilai scale
16 iris = datasets.load_iris()
17 #iris sebagai estimator/parameter, datasets.load_iris() sebagai
18     item dari suatu nilai
19 X, y = iris.data, iris.target
20 #X, y sebagai estimator/parameter, iris.data, iris.target sebagai
21     item dari 2 nilai yang ada
22 clf.fit(X, y)
23 #clf sebagai estimator/parameter dengan menggunakan metode fit
24     untuk memanggil estimator X, y dengan outputannya
```

```
1 # -*- coding: utf-8 -*-
2 """
3 Created on Wed Feb 26 23:06:05 2020
4
5 @author: User
6 """
7
8 import pickle
9 #pickle merupakan sebuah class yang di import
10 s = pickle.dumps(clf)
11 #s sebagai estimator/parameter dengan pickle.dumps merupakan
12     suatu nilai/item dari estimator/parameter clf
13 clf2 = pickle.loads(s)
14 #clf2 sebagai estimator/parameter, pickle.loads sebagai suatu
15     item, dan s sebagai estimator/parameter yang dipanggil
16 clf2.predict(X[0:1])
17 #clf2.predict sebagai suatu item dengan menggunakan metode
18     predict untuk menentukan suatu nilai dari (X[0:1])
19 y[0]
20 #pada estimator/parameter y berapapun angka yang diganti nilainya
21     akan selalu konstan yaitu 0
22
23
24 from joblib import dump, load
25 #sebuah perintah untuk mengimport class dump, load dari packaged
26     joblib
27 dump(clf, 'filename.joblib')
28 #dump di sini sebagai class yang didalamnya terdapat nilai dari
29     suatu item clf dan data joblib
30 clf = load('filename.joblib')
31 #clf sebagai estimato/parameter dengan suatu nilai load berfungsi
32     untuk mengulang data sebelumnya
```

1.5.3.4 Conventions Type Casting

```
1 # -*- coding: utf-8 -*-
2 """
3 Created on Wed Feb 26 22:45:29 2020
4
5 @author: User
6 """
7
8 #Type Casting
9 from sklearn import svm
10 #pada baris ini merupakan sebuah perintah untuk mengimport class
11 #svm dari packaged sklearn
12 from sklearn import random_projection
13 #pada baris ini merupakan sebuah perintah untuk mengimport class
14 #random projection dari packaged sklearn
15 rng = np.random.RandomState(0)
16 #rng sebagai estimator/parameter dengan nilai suatu itemnya yaitu
17 #np.random.RandomState(0)
18 X = rng.rand(10, 2000)
19 #X sebagai estimator/parameter dengan nilai item rng.rand
20 X = np.array(X, dtype='float32')
21 #X sebagai estimator/parameter dengan nilai item np.array
22 X.dtype
23 #X.dtype sebagai item pemanggil
24 transformer = random_projection.GaussianRandomProjection()
25 #transformer sebagai estimator/parameter dengan memanggil class
26 #random projection
27 X_new = transformer.fit_transform(X)
28 #X new di sini sebagai estimator/parameter dan menggunakan metode
29 #fit
30 X_new.dtype
31 #X new.dtype sebagai item
32
33
34 from sklearn import datasets
35 #pada baris ini merupakan sebuah perintah untuk mengimport class
36 #datasets dari packaged sklearn
37 from sklearn.svm import SVC
38 #pada baris ini merupakan sebuah perintah untuk mengimport class
39 #SVC dari packaged sklearn.svm
40 iris = datasets.load_iris()
41 #iris sebagai estimator/parameter dengan item datasets.load_iris
42 #()
43 clf = SVC(gamma='scale')
44 #clf sebagai estimator/parameter dengan nilai class SVC pada
45 #parameter gamma sebagai set penilaian
46 clf.fit(iris.data, iris.target)
47 #estimator/parameter clf menggunakan metode fit dengan itemnya
48 list(clf.predict(iris.data[:3]))
49 #menambahkan item list dengan metode predict
50 clf.fit(iris.data, iris.target_names[iris.target])
51 #estimator/parameter clf menggunakan metode fit dengan itemnya
52 list(clf.predict(iris.data[:3]))
53 #menambahkan item list dengan metode predict
```

Refitting and updating parameters

```

1 # -*- coding: utf-8 -*-
2 """
3 Created on Wed Feb 26 23:08:34 2020
4
5 @author: User
6 """
7
8 #Multiclass vs. Multilabel Fitting
9 from sklearn.svm import SVC
10 #pada baris ini merupakan sebuah perintah untuk mengimport class
11 #SVC dari packaged sklearn.svm
12 from sklearn.multiclass import OneVsRestClassifier
13 #pada baris ini merupakan sebuah perintah untuk mengimport class
14 #OneVsRestClassifier dari packaged sklearn.multiclass
15 from sklearn.preprocessing import LabelBinarizer
16 #pada baris ini merupakan sebuah perintah untuk mengimport class
17 #LabelBinarizer dari packaged sklearn.preprocessing
18 X = [[1, 2], [2, 4], [4, 5], [3, 2], [3, 1]]
19 y = [0, 0, 1, 1, 2]
20 classif = OneVsRestClassifier(estimator=SVC(gamma='scale',
21                                     random_state=0))
22 classif.fit(X, y).predict(X)
23 y = LabelBinarizer().fit_transform(y)
24 classif.fit(X, y).predict(X)
25
26 from sklearn.preprocessing import MultiLabelBinarizer
27 y = [[0, 1], [0, 2], [1, 3], [0, 2, 3], [2, 4]]
28 y = MultiLabelBinarizer().fit_transform(y)
29 classif.fit(X, y).predict(X)

```

Multiclass vs. multilabel fitting

```

1 # -*- coding: utf-8 -*-
2 """
3 Created on Wed Feb 26 23:10:03 2020
4
5 @author: User
6 """
7
8 #Refitting and Updating Parameters
9 import numpy as np
10 #pada baris ini merupakan sebuah perintah untuk mengimport class
11 #svm dari np
12 from sklearn.svm import SVC
13 #pada baris ini merupakan sebuah perintah untuk mengimport class
14 #SVC dari packaged sklearn.svm
15 rng = np.random.RandomState(0)
16 #rng sebagai estimator/parameter dengan nilai suatu itemnya yaitu
17 #np.random.RandomState(0)
18 X = rng.rand(100, 10)
19 #X sebagai estimator/parameter dengan nilai item rng.rand
20 y = rng.binomial(1, 0.5, 100)
21 #y sebagai estimator/parameter dengan nilai item rng.binomial
22 X_test = rng.rand(5, 10)
23 #X test sebagai estimator/parameter dengan nilai item rng.rand

```

```

21 clf = SVC()
22 #clf sebagai estimator/parameter dan class SVC
23 clf.set_params(kernel='linear').fit(X, y)
24 #set params sebagai item
25 clf.predict(X_test)
26 #menggunakan metode predict
27 clf.set_params(kernel='rbf', gamma='scale').fit(X, y)
28 clf.predict(X_test)

```

1.5.4 Penanganan eror

```

File "C:\Users\User\Anaconda3\lib\site-packages\spyder_kernels\customize
\spydercustomize.py", line 827, in runfile
execfile(filename, namespace)

File "C:\Users\User\Anaconda3\lib\site-packages\spyder_kernels\customize
\spydercustomize.py", line 110, in execfile
exec(compile(f.read(), filename, 'exec'), namespace)

File "C:/Users/User/Documents/Spyder/2.py", line 14
X, y = iris.data, iris.target
^
IndentationError: unexpected indent

```

Gambar 1.50 Error

1.5.4.1 ScreenShoot Eror

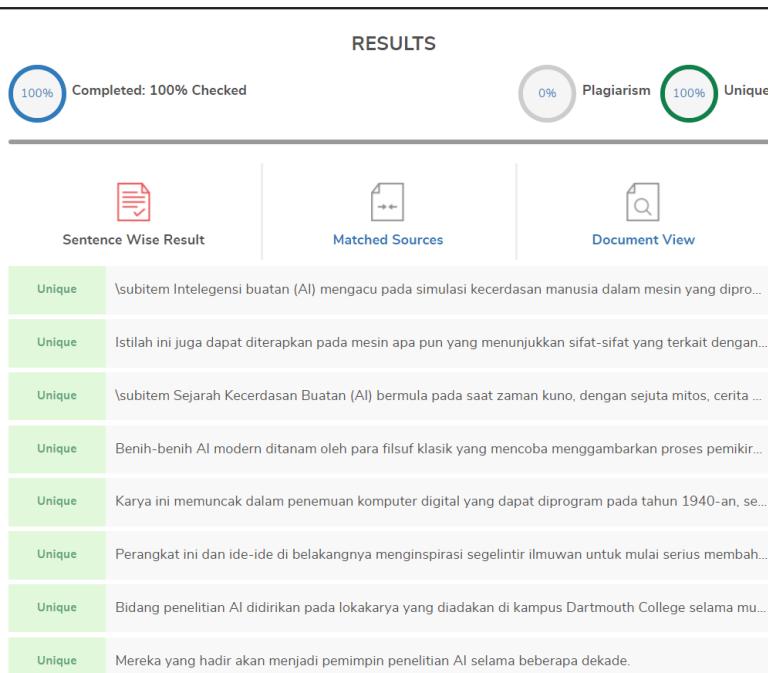
1.5.4.2 Tuliskan Kode Eror dan Jenis Erornya

IndentationError: unexpected indent

1.5.4.3 Solusi Pemecahan Masalah Error

Pastikan semua spasi pada koding sama. Menggunakan spasi atau tab.

1.5.5 Plagiarism



Gambar 1.51 Plagiarism

1.6 1174057 Alit Fajar Kurniawan

1.6.1 Teori

1.6.1.1 Sejarah Perkembangan dan Definisi Artificial Intelligence Kecerdasan buatan merupakan sebuah bidang dalam ilmu computer yang begitu penting di zaman ini dan masa yang akan datang guna mewujudkan sebuah sistem computer yang begitu cerdas. Artificial Intelligence atau biasa disingkat dengan AI berasal dari bahasa latin yang dimana intelligence berarti saya paham.

Pada tahun 1955, Newell dan juga Simon telah mengembangkan The Logic Theorist, yaitu program AI pertama. Dimana program tersebut mempresentasikan sebuah masalah sebagai model pohon, lalu diselesaikan dengan cara memilih cabang yang akan mewujudkan kesimpulan terbenar dan tepat. Program AI tersebut berdampak sangat besar dan dapat mendjadi batu loncatan yang cukup penting dalam mengembangkan bidang AI [?].

Masa Perkembangan AI dimulai pada awal era komputer elektronik pada tahun 1941. dimana ditemukannya alat penyimpanan dan pemrosesan in-

formasi. kemudian dilanjutkan pada masa-masa persiapan AI yang terjadi pada tahun 1943-1956. Pada sekitaran tahun 1952-1969 merupakan masa awal perkembangan AI terjadi, dan pada tahun 1966-1974 perkembangan AI mengalami penurunan atau melambatnya proses dalam melakukan pengembangan. pada tahun 1969 sampai 10 tahun kedepan kembali terjadi perkembangan yang menciptakan inovasi sistem berbasis pengetahuan. dan sekitaran tahun 1980-an AI kembali menjadi sebuah industri yang terus berkembang sampai sekarang ini.

1.6.1.2 learning, klasifikasi, regresi dan unsupervised learning. Data set, training set dan testing set

1. Definisi Supervised Learning Dan Unsupervised Learning

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

2. Definisi Klasifikasi Dan Regresi

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

3. Devinisi Dataset, Training Set, Dan Testing Set

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

1.6.2 Praktek

1.6.2.1 Instalasi

1. Melakukan instalasi library scikit pada anaconda, ketik kan pip install -U scikit-learn pada terminal anaconda.

```
(base) C:\Users\alitf>pip install -U scikit-learn
Collecting scikit-learn
  Downloading https://files.pythonhosted.org/packages/f8/d0/5b7088d1fb891596c34c8b09414cd3daef99876349dd686a3ad536cf9820
    /scikit_learn-0.22.1-cp37-cp37m-win_amd64.whl (6.3MB)
      |████████| 6.3MB 297KB/s
Requirement already satisfied, skipping upgrade: numpy>=1.11.0 in c:\users\alitf\anaconda3\lib\site-packages (from scikit-learn) (1.16.4)
Requirement already satisfied, skipping upgrade: joblib>=0.11 in c:\users\alitf\anaconda3\lib\site-packages (from scikit-learn) (0.13.2)
Requirement already satisfied, skipping upgrade: scipy>=0.17.0 in c:\users\alitf\anaconda3\lib\site-packages (from scikit-learn) (1.2.1)
Installing collected packages: scikit-learn
  Found existing installation: scikit-learn 0.21.2
    Uninstalling scikit-learn-0.21.2:
      Successfully uninstalled scikit-learn-0.21.2
Successfully installed scikit-learn-0.22.1

(base) C:\Users\alitf>
```

Gambar 1.52 Instalasi Scikit Learn

- Setelah selesai instalasi, pilih salah satu example dari website Scikit.

Examples

Miscellaneous examples

Miscellaneous and introductory examples for scikit-learn.

Compact estimator representations

ROC Curve with Visualization API

Isotonic Regression

Advanced Plotting With Partial Dependence

Logistic regression coefficients

Decision regions for multiple classifiers

Cross-validation curves

Partial dependence

Gambar 1.53 Example

```

1 print(__doc__)
2
3 from sklearn.linear_model import LogisticRegression
4 from sklearn import set_config
5
6
7 lr = LogisticRegression(penalty='l1')
8 print('Default representation:')
```

```

9 print(lr)
10 # LogisticRegression(C=1.0, class_weight=None, dual=False ,
11 #                         fit_intercept=True,
12 #                         max_iter=100,
13 #                         multi_class='auto', n_jobs=None, penalty
14 #                         ='l1',
15 #                         random_state=None, solver='warn', tol
16 #                         =0.0001, verbose=0,
17 #                         warm_start=False)
18
19 set_config(print_changed_only=True)
20 print('\nWith changed_only option:')
21 print(lr)
22 # LogisticRegression(penalty='l1')

```

kemudian coba jalankan, lihat hasilnya

@author: alitf

Default representation:

`LogisticRegression(penalty='l1')`

With changed_only option:

`LogisticRegression(penalty='l1')`

Gambar 1.54 Example

3. latihan 2 Mencoba Loading an example dataset

```

1 from sklearn import datasets #mengimport class dataset dari
2     scikit learn library
3 iris = datasets.load_iris() # memuat dan memasukkan dataset
4     iris ke variabel bernama iris
5 digits = datasets.load_digits() #memuat dan memasukkan
6     dataset digits ke variabel digits
7
8 print(digits.data) #memberikan akses ke fitur yang dapat
9     digunakan untuk mengklasifikasikan sampel digit dan
10    menampilkannya di console
11
12 digits.target #memberikan informasi tentang data yang
13    berhubungan atau juga dapat dijadikan sebagai label
14
15 digits.images[0] #Data selalu berupa array 2D, shape (
16     n_samples, n_features), meskipun data aslinya mungkin
17    memiliki bentuk yang berbeda.

```

hasil dari data digits

```
In [17]: runfile('C:/Users/alitf/OneDrive/Desktop/AI/src/1174057'
              wdir='C:/Users/alitf/OneDrive/Desktop/AI/src/1174057/chapter1')
[[ 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.]]
```

Gambar 1.55 Result Data Digits

hasil dari digits.target

```
In [19]: digits.target #memberikan informasi
          dijadikan sebagai Label
Out[19]: array([0, 1, 2, ..., 8, 9, 8])
```

Gambar 1.56 Result digits.target

hasil dari digits.image

```
In [20]: digits.images[0] #Data selalu berupa array 2D, s
          meskipun data aslinya mungkin memiliki bentuk yang berbeda
Out[20]:
array([[ 0.,  0.,  5., 13.,  9.,  1.,  0.,  0.],
       [ 0.,  0., 13., 15., 10., 15.,  5.,  0.],
       [ 0.,  3., 15.,  2.,  0., 11.,  8.,  0.],
       [ 0.,  4., 12.,  0.,  0.,  8.,  8.,  0.],
       [ 0.,  5.,  8.,  0.,  0.,  9.,  8.,  0.],
       [ 0.,  4., 11.,  0.,  1., 12.,  7.,  0.],
       [ 0.,  2., 14.,  5., 10., 12.,  0.,  0.],
       [ 0.,  0.,  6., 13., 10.,  0.,  0.,  0.]])
```

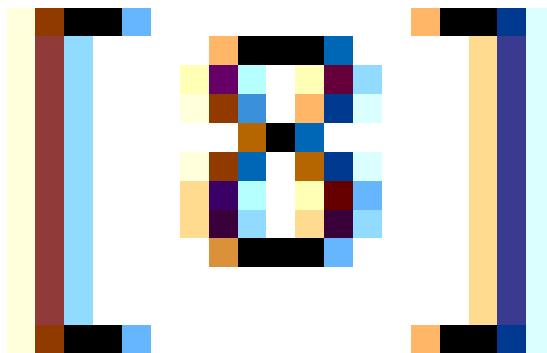
Gambar 1.57 Result digits.image

```

1 from sklearn import svm # perintah untuk mengimport class svm
    dari packaged sklearn
2
3 digits = datasets.load_digits() #memuat dan memasukkan
    dataset digits ke variabel digits
4
5 clf = svm.SVC(gamma=0.001, C=100.) #clf sebagai estimator/
    parameter , svm.SVC sebagai class , gamma sebagai
    parameter untuk menetapkan nilai secara manual
6
7 clf.fit(digits.data[:-1], digits.target[:-1]) # clf sebagai
    estimator/parameter , f i t sebagai metode , digits . data
    sebagai item , [:1] sebagai syntax pythonnya dan
    menampilkan outputannya
8
9 print(clf.predict(digits.data[-1])) #clf sebagai estimator/
    parameter , predict sebagai metode lainnya , digits . data
    sebagai item dan menampilkan outputannya

```

kemudian coba jalankan, lihat hasilnya



Gambar 1.58 Result Learning and predicting

5. latihan 4 Mencoba Model persistence

```

1 from sklearn import svm, datasets #mengimport class dataset
    dari scikit learn library

```

```

2 clf = svm.SVC(gamma=0.001, C=100.) #memanggil class SVC dan
   menyet argument constructor SVC serta ditampung di
   variable clf
3 X, y = datasets.load_iris(return_X_y=True) #meload datasets
   iris dan ditampung di variable x untuk data dan y untuk
   target
4 clf.fit(X, y) #memanggil method fit untuk melakukan training
   data dengan argumen data dan target dari datasets iris
5
6 #Pickle
7 import pickle #mengimport pickle
8 s = pickle.dumps(clf) #memanggil method dumps dengan argumen
   clf dan ditampung di variable s
9 clf2 = pickle.loads(s) #memanggil method loads dengan argumen
   s dan ditampung di variable clf2
10 print(clf2.predict(X[0:1])) #menampilkan hasil dari method
    predict dengan argumen data variable X pertama
11
12 #Joblib
13 from joblib import dump, load #mengimport dump dan load dari
   library joblib
14 dump(clf, '1174057.joblib') #memanggil method dumps dengan
   argumen clf dan nama file joblibnya
15 clf3 = load('1174057.joblib')#memanggil method loads dengan
   argumen nama file joblibnya dan ditampung di variable clf3
16 print(clf3.predict(X[0:1])) #menampilkan hasil dari method
    predict dengan argumen data variable X pertama

```

kemudian coba jalankan, lihat hasilnya

```

In [13]: runfile('D:/Data Alit/Kuliah,
modelpersistence.py', wdir='D:/Data A:
[0]
[0]

```

Gambar 1.59 Result Model persistence

6. latihan 5 Mencoba Conventions

```

1 #Type casting
2 import numpy as np
3 from sklearn import random_projection
4 rng = np.random.RandomState(0)
5 X = rng.rand(10, 2000)
6 X = np.array(X, dtype='float32')
7 print(X.dtype)
8 transformer = random_projection.GaussianRandomProjection()
9 X_new = transformer.fit_transform(X)

```

```
10 print(X_new.dtype)
11
12 from sklearn import datasets
13 from sklearn.svm import SVC
14 iris = datasets.load_iris()
15 clf = SVC(gamma=0.001, C=100.)
16 clf.fit(iris.data, iris.target)
17 print(list(clf.predict(iris.data[:3])))
18 clf.fit(iris.data, iris.target_names[iris.target])
19 print(list(clf.predict(iris.data[:3])))
20
21 #Refitting and updating parameters
22 import numpy as np
23 from sklearn.datasets import load_iris
24 from sklearn.svm import SVC
25 X, y = load_iris(return_X_y=True)
26 clf = SVC(gamma=0.001, C=100.)
27 clf.set_params(kernel='linear').fit(X, y)
28 print(clf.predict(X[:5]))
29 clf.set_params(kernel='rbf').fit(X, y)
30 print(clf.predict(X[:5]))
31
32 #Multiclass vs. Multilabel Fitting
33 from sklearn.svm import SVC
34 from sklearn.multiclass import OneVsRestClassifier
35 from sklearn.preprocessing import LabelBinarizer
36 X = [[1, 2], [2, 4], [4, 5], [3, 2], [3, 1]]
37 y = [0, 0, 1, 1, 2]
38 classif = OneVsRestClassifier(estimator=SVC(random_state=0,
                                              gamma=0.001, C=100.))
39 print(classif.fit(X, y).predict(X))
40 y = LabelBinarizer().fit_transform(y)
41 print(classif.fit(X, y).predict(X))
42
43 from sklearn.preprocessing import MultiLabelBinarizer
44 y = [[0, 1], [0, 2], [1, 3], [0, 2, 3], [2, 4]]
45 y = MultiLabelBinarizer().fit_transform(y)
46 print(classif.fit(X, y).predict(X))
```

kemudian coba jalankan, lihat hasilnya

```
In [20]: runfile('C:/Users/Alit/Desktop/KB3A/src/1174006/chapter1/coba4.py', wdir='C:/Users/Alit/Desktop/KB3A/src/1174006/chapter1')
float32
float64
[0, 0, 0]
['setosa', 'setosa', 'setosa']
[0 0 0 0]
[0 0 0 0]
[0 0 1 1 2]
[[1 0 0]
 [1 0 0]
 [0 1 0]
 [0 0 0]
 [0 0 0]]
[[1 0 1 0 0]
 [1 0 1 0 0]
 [1 0 1 1 0]
 [1 0 1 0 0]
 [1 0 1 0 0]]
```

Gambar 1.60 Result Conventions

1.6.3 Penanganan Error

1. Screenshot Error

```
File "D:/Data Alit/Kuliah/SEMESTER VI/AI/src/1174057/chapter1/learning.py", line 5, in
<module>
    clf.fit(digits.data[:-1], digits.target[:-1]) # clf sebagai estimator/parameter , f
i t sebagai metode , digits . data sebagai item , [:1] sebagai syntax pythonnya dan
menampilkan outputannya

NameError: name 'digits' is not defined
```

Gambar 1.61 Error

```
File "D:/Data Alit/Kuliah/SEMESTER VI/AI/src/1174057/chapter1/learning.py", line 3
    digits = datasets . load digits () #meload datasets digits dan ditampung di variable
digits
^

SyntaxError: invalid syntax
```

Gambar 1.62 Error

2. Tuliskan kode dan jenis error

is not defined, xception yang terjadi saat syntax melakukan eksekusi terhadap local name atau global name yang tidak terdefinisi.

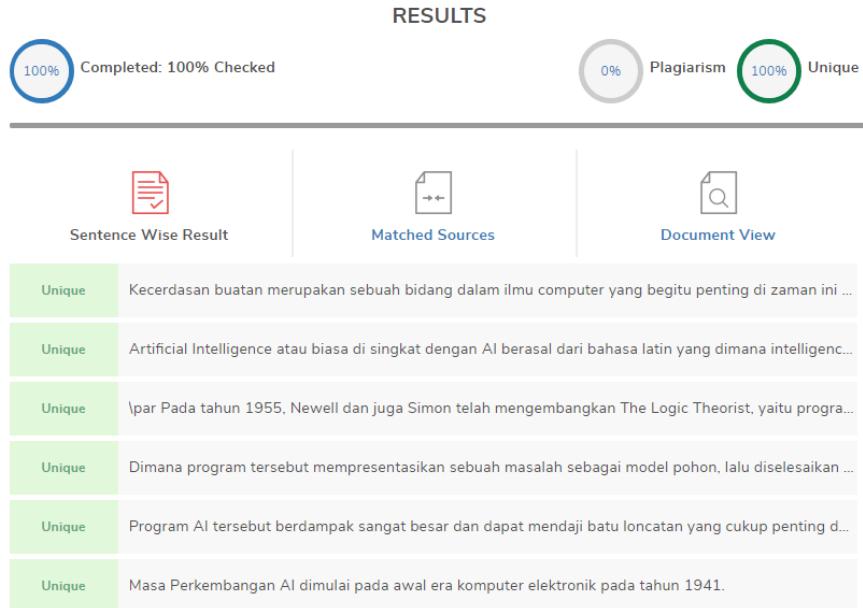
invalid syntax

3. Solusi penanganan error

Solusinya adalah memastikan variabel atau function yang dipanggil ada atau tidak salah ketik.

Periksa kembali syntax yang dibuat, bisa saja ada kesalahan dalam spasi.

1.6.4 Bukti Tidak Plagiat



Gambar 1.63 Plagiarisme

1.7 1174039 - Liyana Majdah Rahma

1.7.1 Teori

1.7.1.1 Definisi Kecerdasan Buatan Artificial Intelligence adalah kecerdasan yang ditambahkan pada suatu sistem yang dapat diatur dalam konteks secara ilmiah. Menurut Michael Haelein menyatakan bahwa AI merupakan “sistem yang mempunyai kemampuan untuk menguraikan, belajar agar dapat dgunakan untuk mencapai tujuan dengan melalui adaptasi yang fleksibel”. Kecerdasan buatan juga dapat dibuat dan dimasukkan ke dalam mesin agar dapat melakukan pekerjaan seperti yang dapat dilakukan manusia dengan cepat dan tepat.

1.7.1.2 Sejarah Kecerdasan Buatan Sejarah Kecerdasan buatan pertama kali terjadi pada zaman kuno, terdapat mitos ataupun cerita dan desas-desus tentang sebuah makhluk yang mempunyai kecerdasan serta kesadaran yang

diberikan oleh pengrajin. Benih - benih nya mulai ditanam oleh para filsuf klasik yang mencari cara untuk menggambarkan proses berfikir manusia sebagai manipulasi simbol secara mekanis yang memuncak pada penemuan komputer digital di tahun 1940-an, yaitu sebuah mesin yang didasarkan penalaran matematika. Istilah kecerdasan buatan sendiri baru muncul pada tahun 1956, dan teori -teori nya sudah muncul sejak tahun 1941. Dan Pada tahun 1973, sebagai tanggapan atas kritik dari James Lighthill dan tekanan terus-menerus dari kongres, AS dan Pemerintah Inggris menghentikan pendanaan penelitian yang tidak diarahkan pada kecerdasan buatan, dan tahun-tahun sulit berikutnya akan dikenal sebagai musim dingin AI. Tujuh tahun kemudian, sebuah inisiatif visioner oleh Pemerintah Jepang mengilhami pemerintah dan industri untuk menyediakan miliaran dolar dalam AI, tetapi pada akhir 80-an investor menjadi kecewa dengan kurangnya daya komputer (perangkat keras) yang dibutuhkan dan menarik lebih banyak dana.

1.7.1.3 Perkembangan Kecerdasan Buatan

1. Pada tahun 1958, McCarthy di MIT AI Lab Memo orang pertama yang mendefinisikan bahasa pemrograman pada tingkat tinggi yaitu LISP, yang sekarang mendominasi pembuatan program-program kecerdasan buatan.
2. Kemudian Pada tahun 1959, Nathaniel Rochester dari IBM serta mahasiswa-mahasiswanya mengeluarkan program kecerdasan buatan yaitu Geometry Theorem Prover. selain itu juga Program ini dapat mengeluarkan suatu teorema menggunakan aksioma-aksioma yang ada.
3. Sistem Berbasis Pengetahuan pada tahun 1969-1979
4. Kecerdasan Buatan menjadi sebuah industri pada tahun 1980-1988. Kembaliya Jaringan Syaraf tiruan pada tahun 1986-sekarang.

1.7.2 Instalasi

1.7.2.1 Instalasi Library Scikit dari Anaconda

1. Download aplikasi Anaconda terlebih dahulu.
2. Install aplikasi Anaconda yang sudah di download tadi.
3. Simpan aplikasi sesuai folder yang kita pilih lalu next.
4. Centang Keduanya lalu tekan tombol install.
5. Setelah itu tunggu sampai proses instalasi selesai lalu jika sudah tekan tombol finish.
6. Lalu buka command prompt anda dan tuliskan perintah berikut ini untuk mengecek apakah aplikasinya sudah terinstall.

7. Kemudian ketikkan perintah pip install -U scikit-learn seperti gambar berikut.



Gambar 1.64 Instalasi

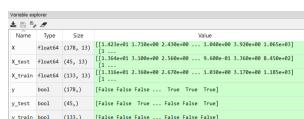
- Pilih salah satu example dari website tersebut lalu jalankan

```

1 import matplotlib.pyplot as plt
2 from sklearn.svm import SVC
3 from sklearn.ensemble import RandomForestClassifier
4 from sklearn.metrics import plot_roc_curve
5 from sklearn.datasets import load_wine
6 from sklearn.model_selection import train_test_split
7
8 X, y = load_wine(return_X_y=True)
9 y = y == 2
10
11 X_train, X_test, y_train, y_test = train_test_split(X, y,
12         random_state=42)
13 svc = SVC(random_state=42)
14 svc.fit(X_train, y_train)

```

- buka variable explolernya



Gambar 1.65 Variable Exploler

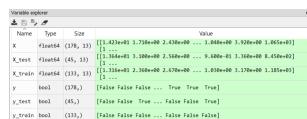
- salah satu example dari website tersebut lalu jalankan

```

1 import matplotlib.pyplot as plt
2 from sklearn.svm import SVC
3 from sklearn.ensemble import RandomForestClassifier
4 from sklearn.metrics import plot_roc_curve
5 from sklearn.datasets import load_wine
6 from sklearn.model_selection import train_test_split
7
8 X, y = load_wine(return_X_y=True)
9 y = y == 2
10
11 X_train, X_test, y_train, y_test = train_test_split(X, y,
12         random_state=42)
13 svc = SVC(random_state=42)
14 svc.fit(X_train, y_train)

```

buka variable explolernya



Gambar 1.66 Variable Exploler

8fcee3a95c7a660681b429f909ba80ac1a4fa418

1.7.2.2 Mencoba loading an example dataset

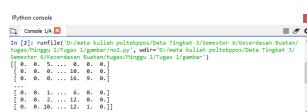
1. mengambil data iris dan digit dari dataset

```

1 iris = datasets.load_iris() # Dapat membuat sebuah variable
    iris yang mempunyai isi yaitu dataset iris
2 digits = datasets.load_digits() # Digunakan untuk membuat
    sebuah variable digits yang mempunyai isi yaitu dataset
    digits

```

2. Menampilkan data digits



```
In [17]: digits.images[0]
Out[17]:
array([[ 0.,  0.,  5., 13.,  9.,  1.,  0.,  0.],
       [ 0.,  0., 13., 15., 10., 15.,  5.,  0.],
       [ 0.,  3., 15.,  2.,  0., 11.,  8.,  0.],
       [ 0.,  4., 12.,  0.,  0.,  8.,  8.,  0.],
       [ 0.,  5.,  8.,  0.,  0.,  9.,  8.,  0.],
       [ 0.,  4., 11.,  0.,  1., 12.,  7.,  0.],
       [ 0.,  2., 14.,  5., 10., 12.,  0.,  0.],
       [ 0.,  0.,  6., 13., 10.,  0.,  0.,  0.]])
```

Gambar 1.69 Data 2D

1.7.2.3 Mencoba Learning and Predicting

```
1 from sklearn.linear_model import LogisticRegression # digunakan
   untuk mengimport linear_model dari library sklearn
2 from sklearn.datasets import make_blobs #digunakan untuk
   mengimport library datasets dari sklearn
3
4 X, y = make_blobs(n_samples=100, centers=2, n_features=2,
   random_state=1) # dapat untuk generate dataset dengan
   klasifikasi 2D
5 model = LogisticRegression() # dapat menggunakan metode loginstic
   regression
6 model.fit(X, y)
7
8 Xnew, _ = make_blobs(n_samples=3, centers=2, n_features=2,
   random_state=1) # dapat menentukan 1 buah contoh baru dimana
   jawabannya tidak dapat diketahui
9
10 ynew = model.predict_proba(Xnew) # membuat sebuah prediksi dan
   memasukkan nya kedalam variable ynew
11 for i in range(len(Xnew)):
12     print("X=%s, Predicted=%s" % (Xnew[i], ynew[i])) #menampilkan
   hasil prediksi
```

1.7.2.4 Mencoba Model Persistence

```
1 from sklearn import svm #dapat mengimport svm dari library
   sklearn
2 from sklearn import datasets #digunakan untuk mengimport datasets
   dari library sklearn
3 clf = svm.SVC() # digunakan dengan menggunakan method SVC
4 iris = datasets.load_iris() # digunakan dengan menggunakan
   dataset iris
5 X, y = iris.data, iris.target #memasukkan x sebagai iris data ,
   dan y sebagai iris target
6 clf.fit(X, y) #laalu menggunakan metod fit .
```

1.7.2.5 Mencoba Conventions

```
1 import numpy as npy #mengimport numpy sebagai npy
2 from sklearn import random_projection #mengimport
   random_projection dari library sklearn
```

```
3
4 rng = np.random.RandomState(0) #Menggunakan fungsi random dari
5     numpy
6 X = rng.rand(10, 2000) #membuat range random diantara 10 sampai
7     2000
8 X = np.array(X, dtype='float32') #yang dijadikan array dengan
9     tipe data float32
10 X.dtype
```

1.7.3 Penanganan Error

```
ValueError: Found array with 0 sample(s) (shape=(0, 2)) while a minimum of 1 is required.
```

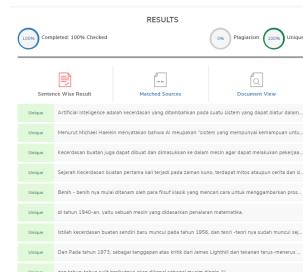
Gambar 1.70 Data 2D

1.7.3.1 Screenshot Error

1.7.3.2 Kode error dan jenis error Jenis errornya adalah value error

1.7.3.3 Solusi Error Solusinya adalah dengan menggantikan nilai nya adalah n_samples nya agar tidak 0

1.7.4 Cek Plagiarism



Gambar 1.71 Cek Plagiarism

BAB 2

CHAPTER 2

2.1 1174042 Faisal Najib Abdullah

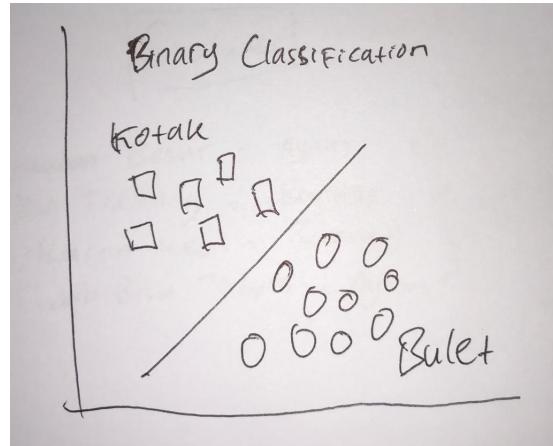
2.1.1 Teori

1. Jelaskan Apa Itu binari classification dilengkapi ilustrasi gambar sendiri.

Binary Classification atau biominal adalah tugas mengklasifikasikan unsur unsur dari himpunan yang diberikan kedalam kedua kelompok berdasarkan aturan klasifikasi yang telah ditetapkan. binari clasification juga dapat diartikan sebagai pembagi yang hanya memberikan dua pilihan contohnya benar dan salah atau klasifikasi tongkat panjang atau pendek. penjelasan lebih singkatnya binari classification merupakan kegiatan mengklasifikasikan yang hanya memberikan dua class.

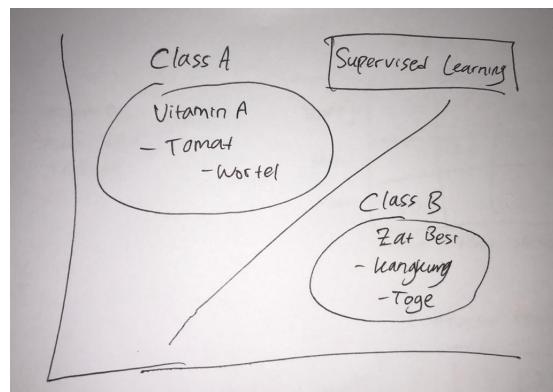
2. Jelaskan Apaitu supervised learning , unsupervised learning dan clustering dengan ilustrasi gambar sendiri.

supervised learning adalah cara untuk mengklasifikasikan suatu objek atau data yang telah di tentukan kelas kelasnya contoh pada sayuran tumbuhan wortel termasuk yang mengandung vitamin A berarti tum-



Gambar 2.1 contoh binari calssification

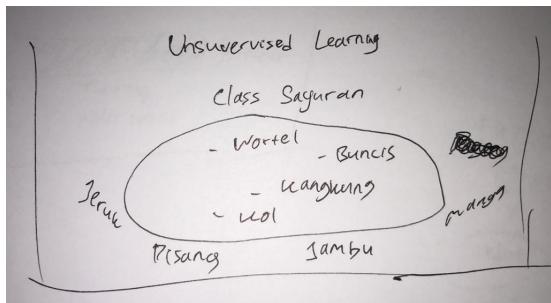
bahan wortel telah di kategorikan kedalam sayuran yang mengandung vitamin A. sedangkan kangkung mengandung zat besi yang berarti tumbuhan kangkung telah di kategorikan kedalam sayuran yang mengandung zat besi untuk lebih jelasnya dapat dilihat pada gambar.



Gambar 2.2 contoh supervised learning

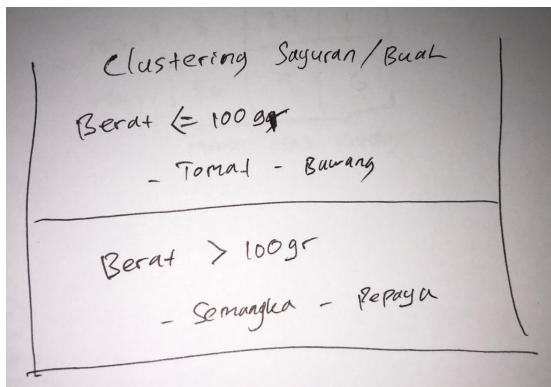
unsupervised learning merupakan cara untuk mengklasifikasi tanpa adanya kelas untuk menentukan jenisnya contoh sayuran berarti semua objek yang memiliki ciri ciri sayuran di kategorikan kedalam sayuran untuk lebih jelasnya dapat dilihat pada gambar.

clustering merupakan peroses mengklasifikasikan yang berdasarkan suatu parameter dalam penentuannya contoh pada berat sayuran sayuran A memiliki berat 100 gr dan sayuran B memiliki berat 120 gr yang berarti



Gambar 2.3 contoh unsupervised learning

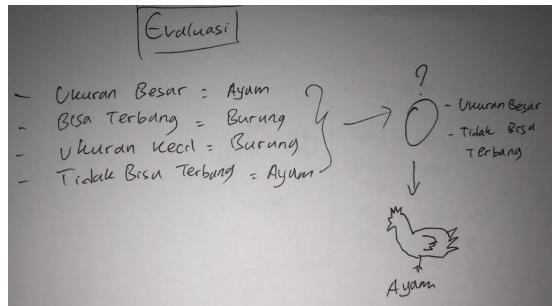
berat sayuran dibagi dua parameter yaitu lebih kecil samadengan 100 gram dan lebih besar dari gram contoh pada gambar.



Gambar 2.4 contoh clusterring

3. Jelaskan apa itu evaluasi dan akurasi dan disertai ilustrasi contoh dengan gambar sendiri.

evaluasi adalah pengumpulan pengumpulan dan pengamatan dari berbagai macam bukti untuk mengukur dampak efektifitas dari suatu objek, program, atau proses berkaitan dengan spesifikasi atau persyaratan yang telah di tetapkan sebelumnya. sedangkan akurasi itu sendiri merupakan bagian dari evaluasi yang merupakan ketepatan data terhadap suatu objek berdasarkan keriteria tertentu. kita dapat mengevaluasi seberapa baik model bekerja dengan mengukur akurasinya. ketepatan akan di definisikan sebagai presentase kasus yang di klasifikasikan dengan benar. hal ini berkaitan dengan confusion matrix pada materi selanjutnya. contoh evaluasi untuk membedakan burung dengan ayam terdapat parameter yaitu ukuran badan dan fungsi sayap pada hewan tersebut. lebih jelasnya pada gambar berikut:



Gambar 2.5 contoh evaluasi dan akurasi

4. Jelaskan bagaimana cara membuat Confusion Matrix, Buat confusion matrix sendiri.

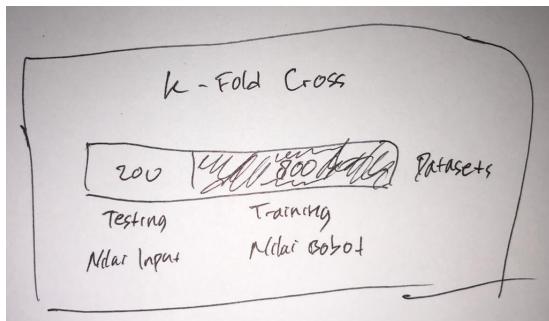
Dalam pembuatan confusion matrix tentukan parameter atau objek yang akan di evaluasi contoh bunga melati , bunga mawar, dan bunga kenangan buat tabel dengan baris dan kolom berjumlah tiga kemudian tentukan nilai miring pada setiap kolom tersebut disini saya memberi nilai 30 dengan ketentuan setiap baris harus berisi nilai 30 nilai tersebut jika terbagi ke kolom lain maka jumlahnya harus bernilai 30 jika tidak berarti data tersebut tidak akurat. untuk lebih jelanya dapat dilihat pada gambar berikut :

| Confusion Matrix | | |
|------------------|---------|---------|
| Kembang | Melati | Mawar |
| 30 | 30 | 30 |
| Melati | Kembang | Mawar |
| Mawar | Melati | Kembang |

| CONFUSION MATRIX | | |
|------------------|--------|---------|
| Kembang | Melati | Mawar |
| 5 | 2 | 23 |
| 5 | 24 | 1 |
| 20 | 4 | 6 |
| Mawar | Melati | Kembang |

Gambar 2.6 contoh Confusion Matrix

5. Jelaskan bagaimana K-fold cross validation bekerja dengan gambar ilustrasi contoh buatan sendiri. K-fold Cross Validation merupakan cara untuk melatih suatu mesin dimana di dalamnya terdapat data set yang dibagi menjadi dua yaitu untuk data testing dan data training contoh 1000 data merupakan data set dan 200 data digunakan untuk data testing kemudian 800 datanya digunakan untuk data training dimana data training tersebut digunakan untuk menentukan nilai bobot yang dimasukan kedalam rumus regresi linier. sedangkan nilai testing akan dijadikan nilai inputan untuk rumus regresi linier. contohnya dapat dilihat pada gambar berikut :



Gambar 2.7 contoh K-fold cross validation

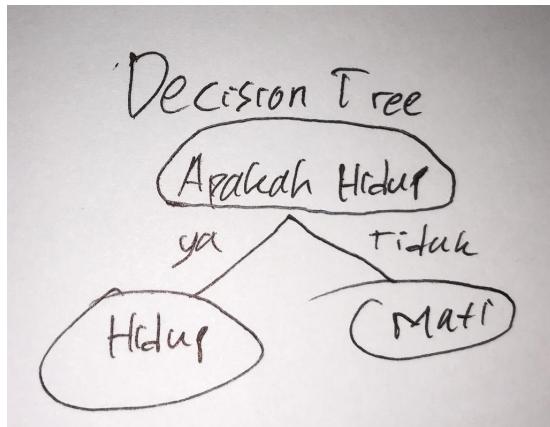
6. Jelaskan Apa itu decision tree dengan gambar ilustrasi contoh buatan sendiri.

Decision tree (pohon keputusan) merupakan implementasi dari binari clasification dimana pada pohon keputusan akan terdapat root atau akar dan cabang cabangnya yang nilainya seperti if contoh pada root berisi nilai jenis kelamin, apakah perempuan pada cabang satu bernilai iya dan pada cabang dua bernilai tidak jika nilainya iya berarti jenis kelamminya perempuan dan jika tidak maka bernilai laki-laki. agar lebih jelas dapat dilihat pada gambar decision tree berikut:

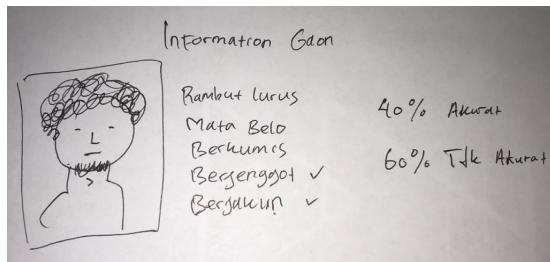
7. jelaskan apa itu information gain dan entropi dengan gambar ilustrasi buatan sendiri.

informasian gain merupakan informasi atau keriteria dalam pembagian sebuah objek contoh information gain pada laki-laki yaitu berrambut lurus, mata belo, berkumis, berjenggot, dan memiliki jakun. untuk lebih jelasnya dapat dilihat pada gambar berikut :

sedangkan entropi merupakan ukuran keacakan dari informasi semakin tinggi entropi maka semakin sulit dalam menentukan keputusan contoh dalam menentukan jenis kelamin semakin detail informasi maka akan semakin susah dalam menentukan keputusan.



Gambar 2.8 contoh decision tree



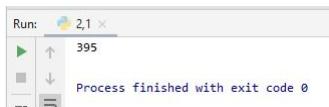
Gambar 2.9 contoh information gain

2.1.2 Sikic-Learn

1. pada surcode pertama yang dapat dilihat pada gambar pada baris pertama di tuliskan yang berarti mengimport library padas. selanjutnya pada baris ke dua codingan tersebut berisi pada code tersebut terdapat variabel muaraenim yang berisi inisialisasi padas dengan aksi untuk membaca vfile csv yang terdapat pada direktori pada komputer kemudian terdapat sep samadengan titik koma yang berarti pemisah field di dalam vile tersebut merupakan titik koma. kemudian pada bagian akhir terdapat code len (nama variabel) yang berarti akan menghotung jumlah baris pada file tersebut.

```

1 # load dataset (student mat pakenya)
2 import pandas as pd
3 muaraenim = pd.read_csv('D:/NAJIB/SEMESTER_6_NAJIB/AI/
Chapter2/dataset/student-mat.csv', sep=';')
4 print(len(muaraenim))
  
```



Gambar 2.10 hasil

2. Pada code selanjutnya akan ditambahkan fungsi untuk lulus atau gagal yang dibuat berupa kolom kolom yang di dalamnya terdapat nilai 0 dan 1 dimana 0 berarti gagal dan 1 berarti lulus. kemudian variabel pada codingan sebelumnya masih digunakan. dimana variabel muaraenim digunakan karena berisi file csv kemudian dilakukan ekseskuasi dengan parameter G1, G2, dan G3 dengan fungsi lambda yang berarti if bersarang atau if didalam if yang berarti nilai yang di eksekusi akan dilempar ke dalam setiap paramater sasuai dengan kriteria dan axis=1 yaitu nilai tersebut akan digunakan dari tiap baris data. sedangkan pada codingan selanjutnya variabel muaraenim di berikan aksi drop yaitu penurunan nilai pada bagian baris data. dan pada code muaraenim.head() yaitu untuk mengeksekusi codingan sebelumnya.

```

1 # generate binary label (pass/fail) based on G1+G2+G3
2 # (test grades, each 0–20 pts); threshold for passing is sum
   >=30
3 muaraenim[ 'pass' ] = muaraenim . apply (lambda row: 1 if (row[ 'G1'
   '] +row[ 'G2' ] +row[ 'G3' ]) 
4 >= 35 else 0, axis=1)
5 muaraenim = muaraenim . drop ([ 'G1' , 'G2' , 'G3' ], axis=1)
6 print (muaraenim . head ())

```

| | 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]

Process finished with exit code 0

Gambar 2.11 hasil

3. pada kodingan selanjutnya diguanakan untuk menambahkan nilai numerik berupa 0 dan 1 yang dirubah dari nilai tidak dan iya hal ini merupakan fungsi dari codingan get_dummies pada baris pertama yang nilainya diambil dari variabel muaraenim yang telah di deklarasikan tadi banyaknya data numerik itu sendiri tergantung pada field dari kolom yang di cattumkan pada codingan dengan catatan field tersebut harus ada dalam data csv yang di import oleh codingan pertama tadi maka hasilnya akan merubah data dalam field tersebut menjadi 0 dan 1.

```

1 # use one-hot encoding on categorical columns
2 muaraenim = pd.get_dummies(muaraenim, columns=['sex', 'school'
    , 'address', 'famsize', 'Pstatus', 'Mjob', 'Fjob', ,
    reason', 'guardian', 'schoolsups', 'famsup', 'paid', ,
    activities', 'nursery', 'higher', 'internet', 'romantic'])
3 muaraenim.head()

```

| | 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 59 columns]

Process finished with exit code 0

Gambar 2.12 hasil

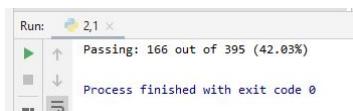
4. selanjutnya pada codingan selanjutnya yaitu menentukan data training dan data testing dari dataset dimana variabel muaraenim yang berisi data csv tadi dibuat perbandingan 500 untuk data training dan sisanya yaitu 149 digunakan untuk data testing hal ini di lakukan pada baris ke 1 sampai ke 3 pada gambar kemudian nilai tersebut di turunkan brdasarkan baris pada data set hal itu dapat dilihat pada baris ke 4 sampai ke 9 pada gambar kemudian selanjutnya mengimport library numpy yang berguna untuk oprasi vektor dan matrix karna data diatas berupa data matrix maka library ini di gunakan.

```

1 # shuffle rows
2 muaraenim = muaraenim.sample(frac=1)
3 # split training and testing data
4 muaraenim_train = muaraenim[:500]
5 muaraenim_test = muaraenim[500:]
6 muaraenim_train_att = muaraenim_train.drop(['pass'], axis=1)
7 muaraenim_train_pass = muaraenim_train['pass']
8 muaraenim_test_att = muaraenim_test.drop(['pass'], axis=1)
9 muaraenim_test_pass = muaraenim_test['pass']
10 muaraenim_att = muaraenim.drop(['pass'], axis=1)
11 muaraenim_pass = muaraenim['pass']
12 # number of passing students in whole dataset:
13 import numpy as np
14 print("Passing: %d out of %d (%.2f%%)" % (np.sum(
        muaraenim_pass), len(muaraenim_pass), 100*float(np.sum(
            muaraenim_pass)) / len(muaraenim_pass)))

```

5. selanjutnya yaitu membuat pohon keputusan pada baris pertama yairu memasukan librari tree kemudian pada baris kedua dibuat variabel palembang dengan nilai DecisionTreeClasifier yang merupakan paket atau fungsi dari scikit-learn yang merupakan class yang mampu melakukan multi

**Gambar 2.13** hasil

class. sedangkan max_depth=5 merupakan untuk penyesuaian data terhadap pohon keputusan itu sendiri.

```
1 # fit a decision tree
2 from sklearn import tree
3 palembang = tree.DecisionTreeClassifier(criterion="entropy",
   max_depth=5)
4 palembang = palembang.fit(muaraeni,_train_att,
   muaraeni_train_pass)
```

**Gambar 2.14** hasil

- selanjutnya yaitu pembuatan gambar dari pohon keputusan yang tadi dibuat pada codingan sebelumnya pada baris pertama codingan ya itu mengimport library graphviz kemudian pada baris ke dua yaitu pemberian nilai pada variabel baru dot data nilainya diambil dari pembuatan pohon keputusan tadi kemudian ditentukannya nilai benar dan salah dari codingan tersebut setelah itu dibuatlah sebuah variabel untuk menampung hasil eksekusi tersebut kemudian variabel tersebut di running.

```
1 # visualize tree
2 import graphviz
3 dot_data = tree.export_graphviz(palembang, out_file=None,
   label="all", impurity=False, proportion=True,
   feature_names=list(
      muaraenim_train_att), class_names=["fail", "pass"],
   filled=True, rounded=True)
5 graph = graphviz.Source(dot_data)
7 graph
```

- selanjutnya pembuatan method untuk menyimpan data pohon dengan menarik data langsung dari pohon keputusan.

```
1 # save tree
2 tree.export_graphviz(palembang, out_file="student-performance
   .dot", label="all", impurity=False, proportion=True,
   feature_names=list(muaraenim_train_att),
   class_names=["fail", "pass"], filled=True, rounded=True)
3
```

```

1  graph TD
2  node [shape=box, style="filled, rounded", color="black", fontname=helvetica];
3  edge [fontname=helvetica];
4  0 [label="failures <= 0.5\nsamples = 100.0%\nvalue = [0.58, 0.42]\nclass = fail", fillcolor="#f8dcc9"];
5  1 [label="schoolsуп_no <= 0.5\nsamples = 79.0%\nvalue = [0.519, 0.481]\nclass = fail", fillcolor="#fdf6f0"];
6  0 -> 1 [label=distance=2.5, labelangle=45, headlabel="True"];
7  2 [label="Mjob_services <= 0.5\nsamples = 10.1%\nvalue = [0.825, 0.175]\nclass = fail", fillcolor="#eb9c63"];
8  1 -> 2 ;
9  3 [label="reason_reputation <= 0.5\nsamples = 6.6%\nvalue = [0.923, 0.077]\nclass = fail", fillcolor="#e78c49"];
10 2 -> 3 ;
11  4 [label="samples = 4.3%\nvalue = [1.0, 0.0]\nclass = fail", fillcolor="#e58139"];
12  3 -> 4 ;
13  5 [label="paid_no <= 0.5\nsamples = 2.3%\nvalue = [0.778, 0.222]\nclass = fail", fillcolor="#eca572"];
14  3 -> 5 ;
15  6 [label="samples = 1.0%\nvalue = [0.5, 0.5]\nclass = fail", fillcolor="#ffffff"];
16  5 -> 6 ;
17  7 [label="samples = 1.3%\nvalue = [1.0, 0.0]\nclass = fail", fillcolor="#e58139"];
18  5 -> 7 ;
19  8 [label="health <= 2.5\nsamples = 3.5%\nvalue = [0.643, 0.357]\nclass = fail", fillcolor="#f3c7a7"];
20  2 -> 8 ;

```

Gambar 2.15 hasil

```

1  graph TD
2  node [shape=box, style="filled, rounded", color="black", fontname=helvetica];
3  edge [fontname=helvetica];
4  0 [label="failures <= 0.5\nsamples = 100.0%\nvalue = [0.58, 0.42]\nclass = fail", fillcolor="#f8dcc9"];
5  1 [label="schoolsуп_no <= 0.5\nsamples = 79.0%\nvalue = [0.519, 0.481]\nclass = fail", fillcolor="#fdf6f0"];
6  0 -> 1 [label=distance=2.5, labelangle=45, headlabel="True"];
7  2 [label="Mjob_services <= 0.5\nsamples = 10.1%\nvalue = [0.825, 0.175]\nclass = fail", fillcolor="#eb9c63"];
8  1 -> 2 ;
9  3 [label="reason_reputation <= 0.5\nsamples = 6.6%\nvalue = [0.923, 0.077]\nclass = fail", fillcolor="#e78c49"];
10 2 -> 3 ;
11  4 [label="samples = 4.3%\nvalue = [1.0, 0.0]\nclass = fail", fillcolor="#e58139"];
12  3 -> 4 ;
13  5 [label="paid_no <= 0.5\nsamples = 2.3%\nvalue = [0.778, 0.222]\nclass = fail", fillcolor="#eca572"];
14  3 -> 5 ;
15  6 [label="samples = 1.0%\nvalue = [0.5, 0.5]\nclass = fail", fillcolor="#ffffff"];
16  5 -> 6 ;
17  7 [label="samples = 1.3%\nvalue = [1.0, 0.0]\nclass = fail", fillcolor="#e58139"];
18  5 -> 7 ;
19  8 [label="health <= 2.5\nsamples = 3.5%\nvalue = [0.643, 0.357]\nclass = fail", fillcolor="#f3c7a7"];
20  2 -> 8 ;

```

Gambar 2.16 hasil

8. selanjutnya pada codingan berikut yaitu digunakan untuk mencetak nilai score rata-rata dari ketepatan data yang telah di olah.

```
1 palembang.score(muaraenim_test_att, muaraenim_test_pass)
```

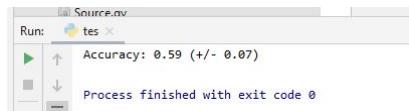
9. selanjutnya yaitu digunakan untuk memeriksa akurasi dari ketepatan hasil pengolahan data tersebut maka akan didapat nilai rata-rata 60 persen dari hasil pengolahan data tersebut, pada codingan tersebut pada baris ke satu melakukan import library dari sklern kemudian pada baris selanjutnya mengisi nilai skor dengan nilai pada variabel palembang setelah hal tersebut dilakukan kemudian data tersebut di eksekusi.

```
1 from sklearn.model_selection import cross_val_score
```

```

2 scores = cross_val_score(palembang, muaraenim_att,
   muaraenim_pass, cv=5)
3 # show average score and +/- two standard deviations away
4 #(covering 95% of scores)
5 print("Accuracy: %0.2f (+/- %0.2f)" % (scores.mean(), scores.
   std() * 2))

```



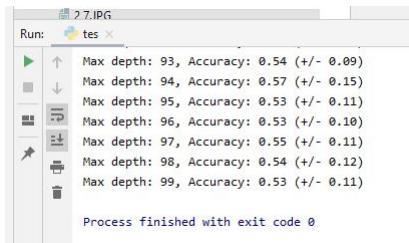
Gambar 2.17 hasil

10. membuat rank akurasi dari 1 sampai 100 untuk melihat akurasi data apakah datatersebut terdapat di rata rata 60 persen atau tidak dengan cara membuat lagi variabel baru dengan nilai tree diadalamnya jadi hampir mirip seperti membuat pohon keputusan namun ini langsung dalam bentuk rata rata akurasi yanlebih spesifik.

```

1 for max_depth in range(1, 100):
2     palembang = tree.DecisionTreeClassifier(criterion="entropy",
   max_depth=max_depth)
3     scores = cross_val_score(palembang, muaraenim_att,
   muaraenim_pass, cv=5)
4     print("Max depth: %d, Accuracy: %0.2f (+/- %0.2f)" % (
   max_depth, scores.mean(), scores.std() * 2))

```



Gambar 2.18 hasil

11. untuk selanjutnya yaitu menentukan nilai untuk grafik hampirsama dengan nilai akurasi tadi pertama tentukan rank atau batasan nilai itu disini batasannya di mulai dari 1 sampai 20 dengan menggunakan nilai tree tadi maka hasilnya dapat ditentuka kemudian buat variabel i untuk penomoran tiap record yang keluar atau record hadil dari eksekusi tree tersebut.

```

1 depth_acc = np.empty((19,3), float)
2 i = 0
3 for max_depth in range(1, 20):
4     palembang = tree.DecisionTreeClassifier(criterion="entropy",
   max_depth=max_depth)

```

```

5     scores = cross_val_score(palembang , muaraenim_att ,
6     muaraenim_pass , cv=5)
7     depth_acc[i , 0] = max_depth
8     depth_acc[i , 1] = scores.mean()
9     depth_acc[i , 2] = scores.std() * 2
10    i += 1
11 print(depth_acc)

```

Gambar 2.19 hasil

12. terakhir yaitu pembuatan grafik untuk pembuatan grafik diambil data dari codingan sebelumnya yang 20 record tadi dengan cara mengimport libray matplotlib.pyplot yang di inisialisasi menjadi bakwankemudian inisialisasi tersebut di eksekusi.

```

1 import matplotlib.pyplot as plt
2 fig , ax = plt.subplots()
3 ax.errorbar(depth_acc[:,0] , depth_acc[:,1] , yerr=depth_acc
4 [:,2])
4 plt.show()

```

2.1.3 Penanganan Error

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

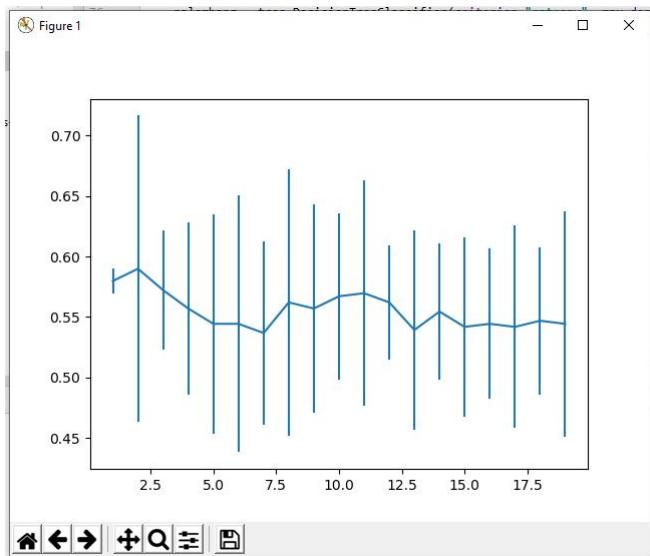
```

import graphviz
dot_data = tree.export_graphviz(palembang, out_file=None, label="all"
                                feature_names=list(muaraenim_train_a
                                filled=True, rounded=True)
graph = graphviz.Source(dot_data)
graph

```

pada codingan tersebut error karena graphiznya belu di istall

3. Solusi pemecahan masalah
buka CMD komputer anda run as administrator koemudian masukan perintah conda install graphviz.



Gambar 2.20 hasil

```
Run: tes
Traceback (most recent call last):
  File "D:/NAJIB/SEMESTER_6 NAJIB/AI/Chapter2/tes.py", line 2, in <module>
    import graphviz
ModuleNotFoundError: No module named 'graphviz'

Process finished with exit code 1
```

Gambar 2.21 Error

2.1.4 Plagiat

2.2 1174035 Luthfi Muhammad Nabil

2.2.1 Teori

1. Jelaskan Apa Itu Binary Classification dilengkapi ilustrasi gambar sendiri.

Binary Classification adalah sebuah metode untuk menklasifikasikan elemen yang dibentuk seperti grup untuk dibagi menjadi 2 grup. Dari 2 grup tersebut diprediksi setiap anggota pada grup yang mana sesuai dengan yang diatur pada aturan klasifikasi. Data atau konteks yang didapat membutuhkan keputusan dari item tersebut memiliki properti kualitatif, karakteristik yang spesifik, atau tipikal klasifikasi biner.

RESULTS



Completed: 100% Checked



Plagiarism



Unique



Sentence Wise Result

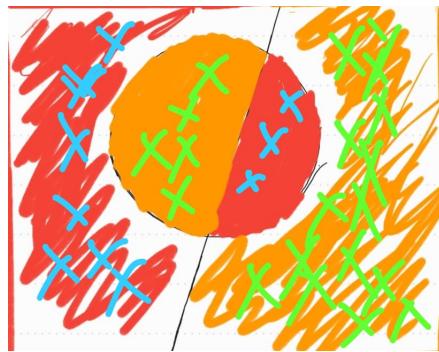


Matched Sources



Document View

| | |
|--------|---|
| Unique | \subitem Definisi kecerdasan buatan adalah suatu pengetahuan yang dapat membuat komputer untuk ... |
| Unique | Contohnya yaitu melakukan analisa penalaran untuk mengambil suatu kesimpulan atau penerjemahan ... |
| Unique | \subitem Sejarah dan perkembangan kecerdasan buatan terjadi pada musim panas tahun 1956 tercata... |
| Unique | Seminar pada waktu itu dihadiri oleh sejumlah pakar komputer dan membahas potensi komputer dala... |
| Unique | Akan tetapi perkembangan yang sering terjadi semenjak diciptakannya LISP, yaitu bahasa kecerdasan ... |

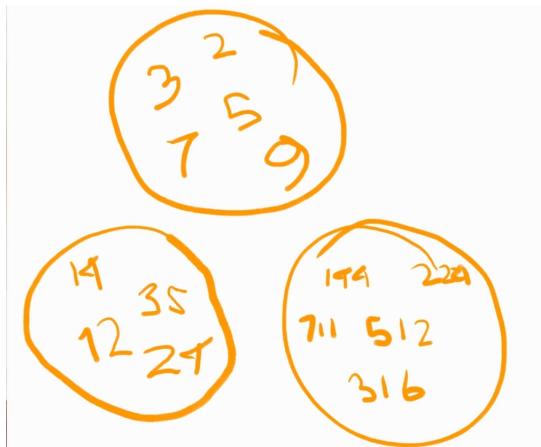
Gambar 2.22 Error**Gambar 2.23** contoh binary classification

2. Jelaskan Apa itu supervised learning , unsupervised learning dan clustering dengan ilustrasi gambar sendiri.

Supervised learning adalah kondisi yang menggunakan variabel input dan output untuk dapat dilakukan pemetaan input output yang sudah didapat. Disebut Supervised Learning karena proses dari pembelajaran algoritma dari pembelajaran yang disumberkan dengan dataset dapat dipikirkan seperti seorang guru yang mengawasi proses pembelajaran. Proses pembelajaran dari algoritma akan berhenti saat algoritma sudah mendapatkan level dari performansi yang dapat diterima.

Unsupervised learning adalah kondisi dimana kamu hanya memiliki input data tanpa memiliki variabel output yang sesuai. Tujuan dari unsupervised learning adalah untuk memodelkan distribusi pada data untuk mengetahui lebih lanjut mengenai data. Disebut unsupervised learning karena pada metode ini, tidak ada jawaban yang tepat dan tidak ada pengarah. Sehingga algoritma akan ditinggalkan sesuai rancangan demi menemukan dan dapat mengolah data yang menarik pada saat yang akan datang.

Clustering adalah sebuah metode untuk membedakan data - data menjadi kumpulan dari group yang isinya merupakan data yang serupa setiap grupnya. Basisnya dapat berupa kesamaan atau perbedaan dari setiap grup tersebut.



Gambar 2.24 contoh clustering

3. Jelaskan apa itu evaluasi dan akurasi dan disertai ilustrasi contoh dengan gambar sendiri.

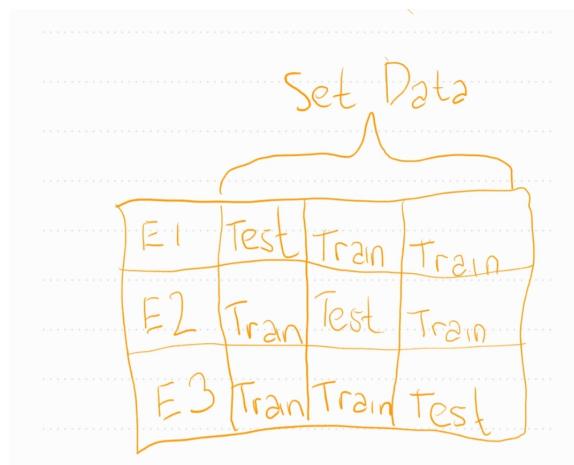
Evaluasi adalah kegiatan yang dilakukan untuk menentukan sebuah nilai yang dapat diambil dari suatu hal. Beberapa contoh evaluasi diantaranya menilai sebuah barang, bahaya dari penyakit, dan lain sebagainya dengan parameter yang digunakan yaitu mengetahui faktor - faktor yang menyebabkan atau akibat yang akan terjadi jika dibiarkan.

4. Jelaskan bagaimana cara membuat Confusion Matrix, Buat confusion matrix sendiri.

Confusion Matrix merupakan metode untuk menghitung akurasi pada data mining atau Sistem Pendukung Keputusan. Untuk menggunakan Confusion Matrix, ada 4 istilah sebagai hasil proses dari klasifikasi. Di antaranya adalah :

- True Positive : Data positif yang terdeteksi memiliki hasil benar
- False Positive : Data Positif yang terdeteksi memiliki hasil salah
- True Negative : Data negatif yang terdeteksi memiliki hasil benar
- False Negative : Data negatif yang terdeteksi memiliki hasil salah

5. Jelaskan bagaimana K-fold cross validation bekerja dengan gambar ilustrasi contoh buatan sendiri. k-Fold Cross-Validation merupakan prosedur untuk mengambil sampel ulang yang digunakan untuk mengevaluasi sebuah "machine learning" model pada sampel data yang terbatas. Procedure yang ada memiliki satu parameter yang dipanggil k yang mengacu pada jumlah grup yang memberikan data sampel untuk dipisahkan. K-fold Cross-validation akan melakukan hal berikut :
- (a) Mengacak dataset secara random
 - (b) Memisahkan dataset menjadi k group
 - (c) Untuk setiap grup, akan disesuaikan dan dievaluasi
 - (d) Merekaphasil dari evaluasi dan penyesuaian menggunakan sampel dari model evaluasi



Gambar 2.25 contoh K-fold cross validation

6. Jelaskan Apa itu decision tree disertakan gambar ilustrasi contoh buatan sendiri.

Decision Tree merupakan sebuah struktur yang menentukan keputusan dan setiap konsekuensinya. Hasil dari setiap struktur biasanya menggunakan jawaban (True dan False) atau cabang lain yang akan menjadi pohon selanjutnya. Setiap keputusan diantaranya akan membandingkan

kondisi yang diberikan kepada struktur untuk dibandingkan kondisi apa saja yang sudah didapat pada sistem tersebut.

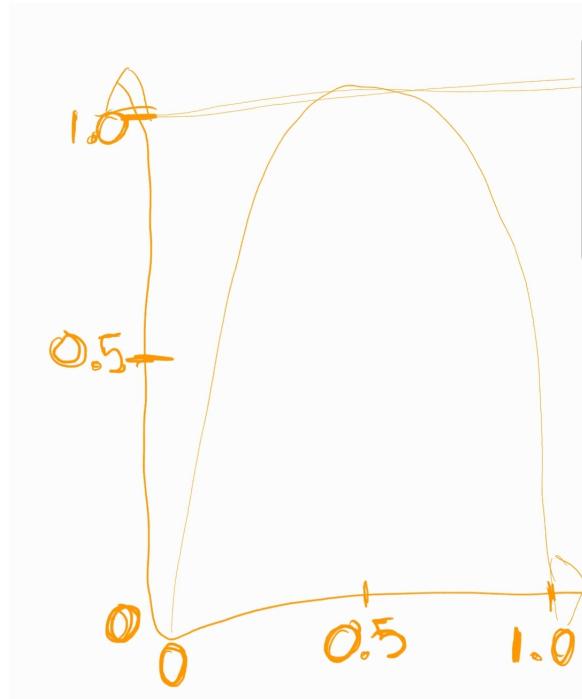
7. jelaskan apa itu information gain dan entropi dengan gambar ilustrasi buatan sendiri.

Information Gain merupakan total data yang didapat dari data - data acak yang data tersebut akan digunakan untuk analisis data lainnya. Information Gain ini digunakan pada decision tree sebagai label setiap aksi - aksi yang perlu dinilai validasinya.

| | | Coding | |
|---------|-------------|--------|-------|
| | | Ya | Tidak |
| Outlook | Fokus | 3 | 2 |
| | Normal | 4 | 0 |
| | Tidak Fokus | 2 | 3 |

Gambar 2.26 contoh information gain

Entropi merupakan pengukuran sebuah data dan validnya data tersebut untuk dapat digunakan sebagai informasi yang akan dimasukkan ke Information Gain. Entropi menilai sebuah obyek berdasarkan kebutuhan di dunia nyata dan pengaruh pada sistem yang akan digunakan



Gambar 2.27 contoh information gain

2.2.2 Scikit-Learn

1.

```
1 import pandas as pd #mengimpor library pandas
2 cakue = pd.read_csv('student-mat.csv', sep=';') #data csv
    dibaca lalu dipisahkan dengan titik koma';'
3 len(cakue) #menghitung total nilai(panjang kumpulan nilai/
    array) yang terpisahkan dari csv tersebut
```

Gambar 2.28 Hasil Percobaan 1

2.

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

```

2 cakue = cakue.drop(['G1', 'G2', 'G3'], axis=1)
3 cakue.head()#mengambil baris pertama dari cakue

```

| In [8]: | cakue[cols] + cakue.apply(lambda row: 1 if (row['G1'] > row['G2']) & (row['G2'] > row['G3']) else 0, axis=1) |
|---------|--|
| Out[8]: | school sex age address family Debt male results absences pass |
| 0 | G1 GP 15 U 0 0 1 1 1 0 0 |
| 1 | G2 GP 15 U 0 0 1 1 1 1 0 |
| 2 | G3 GP 15 U 0 0 1 1 1 2 0 |
| 3 | G1 GP 15 U 0 0 1 1 1 1 0 |
| 4 | G2 GP 15 U 0 0 1 1 1 2 0 |

[5 rows x 31 columns]

Gambar 2.29 Hasil Percobaan 2

3.

```

1 cakue = pd.get_dummies(cakue, columns=['sex', 'school', 'address', 'famsize', 'Pstatus', 'Mjob', 'Fjob', 'reason', 'guardian', 'schoolsup', 'famsup', 'paid', 'activities', 'nursery', 'higher', 'internet', 'romantic'])
2 cakue.head()#mengambil baris pertama dari cakue

```

| In [9]: | cakue[cols] + cakue.apply(lambda row: 1 if (row['G1'] > row['G2']) & (row['G2'] > row['G3']) else 0, axis=1) |
|---------|--|
| Out[9]: | school sex age address familyDebt male results absences pass |
| 0 | GP 15 U 0 0 1 1 1 0 0 |
| 1 | GP 15 U 0 0 1 1 1 1 0 |
| 2 | GP 15 U 0 0 1 1 1 2 0 |
| 3 | GP 15 U 0 0 1 1 1 1 0 |
| 4 | GP 15 U 0 0 1 1 1 2 0 |

[5 rows x 31 columns]

Gambar 2.30 Hasil Percobaan 3

4.

```

1 # shuffle rows
2 cakue = cakue.sample(frac=1) #mengenerate sampel acak dari baris atau kolom dari cakue
3 # split training and testing data
4 cakue_train = cakue[:500]#Mengambil data array dengan batas index 500
5 cakue_test = cakue[500:] #Mengambil data array dengan awal index 500
6 cakue_train_att = cakue_train.drop(['pass'], axis=1) #
7 cakue_train_pass = cakue_train['pass'] #Mengambil data cakue_train dengan index object 'pass' dan dimasukkan ke variable cakue_train_pass
8 cakue_test_att = cakue_test.drop(['pass'], axis=1)
9 cakue_test_pass = cakue_test['pass'] #Mengambil data cakue_train dengan index object 'pass' dan dimasukkan ke variable cakue_test_pass
10 cakue_att = cakue.drop(['pass'], axis=1)
11 cakue_pass = cakue['pass']#Mengambil data cakue_train dengan index object 'pass' dan dimasukkan ke variable cakue_pass
12 # number of passing students in whole dataset:
13 import numpy as np #mengimport library numpy
14 print("Passing: %d out of %d (%.2f%%)" % (np.sum(cakue_pass), len(cakue_pass), 100*float(np.sum(cakue_pass)) / len(cakue_pass))) #Menampilkan hasil integer dan float untuk melihat passing datanya

```

```
In [20]: from sklearn import tree #Mengimport class tree dari library sklearn
In [21]: kwetiau = tree.DecisionTreeClassifier(criterion="entropy",
                                             max_depth=5) #Memanggil fungsi untuk melakukan prediksi dengan aturan keputusan yang simpel
In [22]: kwetiau = kwetiau.fit(cakue_train_att, cakue_train_pass) # Untuk melakukan pengiriman data training set ke method fit
```

Gambar 2.31 Hasil Percobaan 4

5.

```
1 from sklearn import tree #Mengimport class tree dari library sklearn
2 kwetiau = tree.DecisionTreeClassifier(criterion="entropy",
3                                     max_depth=5) #Memanggil fungsi untuk melakukan prediksi dengan aturan keputusan yang simpel
4 kwetiau = kwetiau.fit(cakue_train_att, cakue_train_pass) # Untuk melakukan pengiriman data training set ke method fit
```

```
In [23]: import graphviz #mengimport class graphviz
In [24]: dot_data = tree.export_graphviz(kwetiau, out_file=None, label="all", impurity=False, proportion=True, feature_names=list(cakue_train_att), class_names=["fail", "pass"], filled=True, rounded=True) #Untuk mengekspor data ke format graphviz
In [25]: graph = graphviz.Source(dot_data) #Untuk mengambil sumber data berformat graphviz dan dimasukkan ke variable graph
In [26]: graph #Menampilkan isi variable graph untuk di spyder
```

Gambar 2.32 Hasil Percobaan 5

6.

```
1 import graphviz #mengimport class graphviz
2 dot_data = tree.export_graphviz(kwetiau, out_file=None, label="all", impurity=False, proportion=True, feature_names=list(cakue_train_att), class_names=["fail", "pass"], filled=True, rounded=True) #Untuk mengekspor data ke format graphviz
3 graph = graphviz.Source(dot_data) #Untuk mengambil sumber data berformat graphviz dan dimasukkan ke variable graph
4 graph #Menampilkan isi variable graph untuk di spyder
```

```
In [27]: import graphviz #mengimport class graphviz
In [28]: dot_data = tree.export_graphviz(kwetiau, out_file=None, label="all", impurity=False, proportion=True, feature_names=list(cakue_train_att), class_names=["fail", "pass"], filled=True, rounded=True) #Untuk mengekspor data ke format graphviz
In [29]: graph = graphviz.Source(dot_data) #Untuk mengambil sumber data berformat graphviz dan dimasukkan ke variable graph
In [30]: graph #Menampilkan isi variable graph untuk di spyder
In [31]: tree.export_graphviz(kwetiau, out_file="student-performance.dot", label="all", impurity=False, proportion=True, feature_names=list(cakue_train_att), class_names=["fail", "pass"], filled=True, rounded=True) #Untuk mengekspor data ke format graphviz
In [32]: !dot -Tpdf student-performance.dot > student-performance.pdf
In [33]: graphviz.Source('student-performance.pdf')
```

Gambar 2.33 Hasil Percobaan 6

7.

```
1 tree.export_graphviz(kwetiau, out_file="student-performance.dot", label="all", impurity=False, proportion=True, feature_names=list(cakue_train_att), class_names=["fail", "pass"], filled=True, rounded=True) #Untuk mengekspor data ke format graphviz
```

Gambar 2.34 Hasil Percobaan 7

8.

```
1 kwetiau.score(cakue_test_att, cakue_test_pass) #Untuk
   melakukan penilaian sesuai dengan drop – drop pada
   variable cakue_test_att dan cakue_test_pass
```

```
In [12]: kwetiau.score(cakue_test_att, cakue_test_pass)
Out[12]: 0.67
In [13]: #Untuk mengevaluasi nilai menggunakan metode cross-validation
In [14]: scores = cross_val_score(kwetiau, cakue_att, cakue_pass, cv=5)
In [15]: print("Accuracy: %0.2f (+/- %0.2f)" % (scores.mean(), scores.std() * 2))
Out[15]: Accuracy: 0.67 (+/- 0.21)
```

Gambar 2.35 Hasil Percobaan 8

9.

```
1 from sklearn.model_selection import cross_val_score # Mengambil class cross_val_score dari library sklearn.model_selection
2 scores = cross_val_score(kwetiau, cakue_att, cakue_pass, cv=5) #Untuk mengevaluasi nilai menggunakan metode cross-validation
3 # show average score and +/- two standard deviations away
4 #(covering 95% of scores)
5 print("Accuracy: %0.2f (+/- %0.2f)" % (scores.mean(), scores.std() * 2)) #Menampilkan score
```

```
In [15]: print("Accuracy: %0.2f (+/- %0.2f)" % (scores.mean(), scores.std() * 2))
Out[15]: Accuracy: 0.67 (+/- 0.21)
```

Gambar 2.36 Hasil Percobaan 9

10.

```
1 for max_depth in range(1, 20): #Looping dengan for berdasarkan nilai dari 1 sampai 20 dengan variable index yaitu max_depth
2     kwetiau = tree.DecisionTreeClassifier(criterion="entropy", max_depth=max_depth) #Memanggil fungsi untuk melakukan prediksi dengan aturan keputusan yang simpel
3     scores = cross_val_score(kwetiau, cakue_att, cakue_pass, cv=5) #Untuk mengevaluasi nilai menggunakan metode cross-validation
4     print("Max depth: %d, Accuracy: %0.2f (+/- %0.2f)" % (max_depth, scores.mean(), scores.std() * 2)) #Menampilkan score
```



Gambar 2.37 Hasil Percobaan 10

11.

```

1 depth_acc = np.empty((19,3), float) #Mengembalikan array
    dengan format bentuk dan tipe
2 i = 0#Inisiasi variable
3 for max_depth in range(1, 20): #Looping dengan for
    berdasarkan nilai dari 1 sampai 20 dengan variable index
    yaitu max_depth
4     kwetiau = tree.DecisionTreeClassifier(criterion="entropy",
        , max_depth=max_depth) #Memanggil fungsi untuk melakukan
        prediksi dengan aturan keputusan yang simpel
5     scores = cross_val_score(kwetiau, cakue_att, cakue_pass,
        cv=5) #Untuk mengevaluasi nilai menggunakan metode cross-
        validation
6     depth_acc[i,0] = max_depth
7     depth_acc[i,1] = scores.mean()
8     depth_acc[i,2] = scores.std() * 2
9     i += 1
10    depth_acc

```



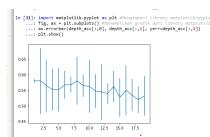
Gambar 2.38 Hasil Percobaan 11

12.

```

1 import matplotlib.pyplot as plt #Mengimport library
    matplotlib>pyplot
2 fig, ax = plt.subplots() #Menampilkan grafik dari library
    matplotlib
3 ax.errorbar(depth_acc[:,0], depth_acc[:,1], yerr=depth_acc
    [:,2])
4 plt.show()

```



Gambar 2.39 Hasil Percobaan 12

2.2.3 Skrinsut Error

Error yang didapat yaitu mengalami error tidak ada library graphviz yang terdeteksi



Gambar 2.40 Error

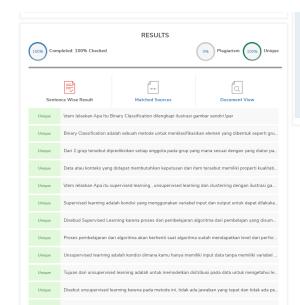
2.2.4 Penanganan Error

Solusinya adalah dengan menginstall library graphviz yang ada



Gambar 2.41 Error

2.2.5 Plagiarisme



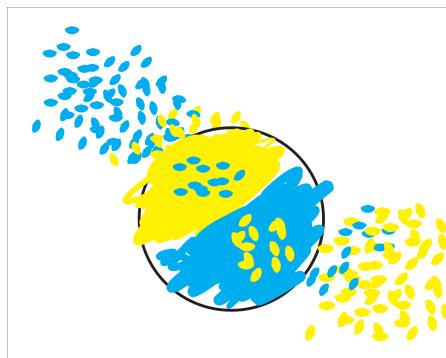
Gambar 2.42 Hasil Pengecekan Plagiarisme

2.3 1174057 - Alit Fajar Kurniawan

2.3.1 Teori

2.3.1.1 Jelaskan apa itu binary classification dilengkapi ilustrasi gambar sendiri

Binary classification merupakan jenis masalah pembelajaran mesin yang paling sederhana. mengklarifikasi elemen-elemen dari himpunan yang diberikan kedalam dua kelompok berdasarkan aturan klarifikasi. Contoh yang paling sederhana dalam binary classification yaitu mendeteksi dan mendiagnosa. Berikut contoh gambar Binary Classification, gambar 2.1



Gambar 2.43 Klasifikasi Binari

2.3.1.2 Jelaskan apa itu supervised learning dan unsupervised learning dan clustering dengan ilustrasi gambar sendiri

Supervised Learning merupakan paradigma belajar yang berkaitan dengan studi tentang bagaimana komputer dan sistem alami seperti manusia belajar [?]. Tujuannya untuk menyimpulkan pemetaan fungsional berdasarkan serangkaian pelatihan atau mengelompokkan suatu data ke data yang sudah ada. Berikut contoh gambar Supervised Learning

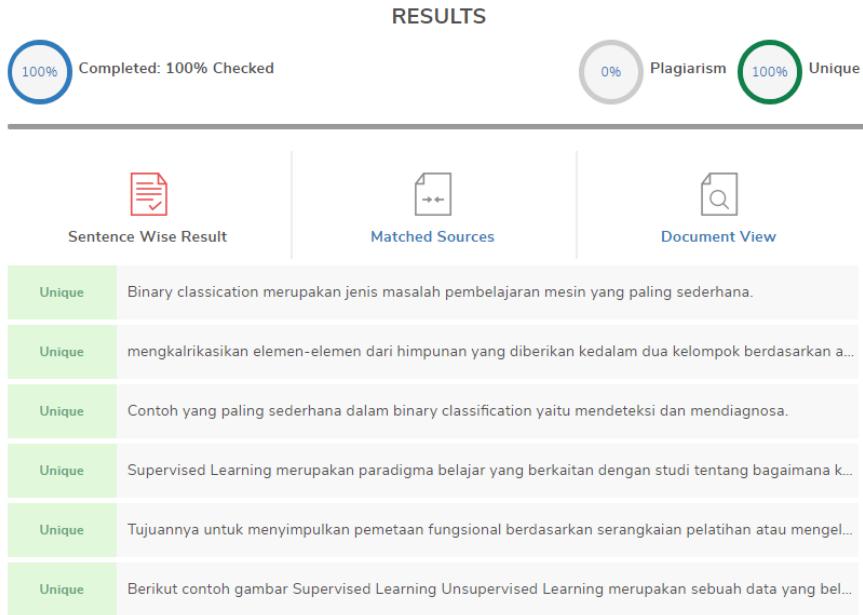
Unsupervised Learning merupakan sebuah data yang belum ditentukan variabelnya jadi hanya berupa data saja. Dalam sebuah kasus Unsupervised Learning adalah aggap saja anda belum pernah membeli buku sama sekali dan pada suatu hari anda telah membeli buku dengan sangat banyak dalam kategori yang berbeda.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

Clustering merupakan metode pengelompokan data, clustering juga suatu proses partisi satu set objek data ke dalam himpunan bagian yang disebut dengan cluster. Objek yang pada cluster memiliki kesamaan secara karakteristik antara satu sama lainnya dan berbeda dengan cluster yang lain.

2.3.2 Praktek

2.3.3 Penanganan Error

2.3.4 Bukti Tidak Plagiat



Gambar 2.44 Plagiarisme

2.4 1174050 Dika Sukma Pradana

2.4.1 Teori

1. Jelaskan Apa Itu binari calssification dilengkapi ilustrasi gambar sendiri.

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



Gambar 2.45 contoh binari calssification

2. Jelaskan Apaitu supervised learning , unsupervised learning dan clustering dengan ilustrasi gambar sendiri.

Supervised learning adalah sebuah metode pendekatan yang mana sudah terdapat data yang dilatih, dan terdapat variabel yang ditargetkan atau yang menjadi tujuan sehingga tujuan dari pendekatan ini adalah mengelompokan suatu data ke data yang sudah ada. contoh pada nasi termasuk yang mengandung karbohidrat berarti nasi telah di kategorikan kedalam karbohidrat. sedangkan ayam mengandung protein yang berarti ayam telah di kategorikan yang mengandung protein untuk lebih jelasnya dapat dilihat pada gambar.



Gambar 2.46 contoh supervised learning

Sedangkan unsupervised learning tidak memiliki data latih, sehingga dari data yang ada, kita mengelompokan data tersebut menjadi dua bagian atau bahkan tiga bagian dan seterusnya. contoh ayam di kategorikan kedalam karbohidrat untuk lebih jelasnya dapat dilihat pada gambar.



Gambar 2.47 contoh unsupervised learning

clustering merupakan proses mengklasifikasikan yang berdasarkan suatu parameter dalam penentuannya contoh pada berat protein memiliki berat 1000 gr dan karbohidrat memiliki berat 1200 gr yang berarti berat dibagi dua parameter yaitu lebih kecil samadengan 1000 gram dan lebih besar dari 1000gram contoh pada gambar.



Gambar 2.48 contoh clusterring

3. Jelaskan apa itu evaluasi dan akurasi dan disertai ilustrasi contoh dengan gambar sendiri.

evaluasi adalah pengumpulan pengumpulan dan pengamatan dari berbagai macam bukti untuk mengukur dampak efektifitas dari suatu objek, program, atau proses berkaitan dengan spesifikasi atau persyaratan yang telah di tetapkan sebelumnya. sedangkan akurasi itu sendiri merupakan bagian dari evaluasi yang merupakan ketepatan data terhadap suatu objek berdasarkan keriteria tertentu. kita dapat mengevaluasi seberapa baik model bekerja dengan mengukur akurasiinya. ketepatan akan definisikan sebagai persentase kasus yang diklasifikasikan dengan benar. hal ini berkaitan dengan confusion matrix pada materi selanjutnya. contoh evaluasi untuk membedakan angsa dengan entok terdapat parameter yaitu panjang leher dan fungsi sifat dari hewan tersebut. lebih jelasnya pada gambar berikut:



Gambar 2.49 contoh evaluasi dan akurasi

4. Jelaskan bagaimana cara membuat Confusion Matrix, Buat confusion matrix sendiri.

Dalam pembuatan confusion matrix tentukan parameter atau objek yang akan di evaluasi contoh angsa, entok, bebek buat tabel dengan baris dan kolom berjumlah tiga kemudian tentukan nilai miring pada setiap kolom tersebut disini saya memberi nilai 15 dengan ketentuan setiap baris harus berisi nilai 15 nilai tersebut jika terbagi ke kolom lain maka jumlahnya harus bernilai 15 jika tidak berarti data tersebut tidak akurat. untuk lebih jelasnya dapat dilihat pada gambar berikut :

| | | Predicted | |
|--------|-------|-----------|-------|
| | | Angsa | Entok |
| Actual | Angsa | 15 | 5 |
| | Entok | 2 | 15 |

Gambar 2.50 contoh Confusion Matrix

5. Jelaskan bagaimana K-fold cross validation bekerja dengan gambar ilustrasi contoh buatan sendiri. K-fold Cross Validation merupakan cara untuk melatih suatu mesin dimana dalamnya terdapat data set yang dibagi menjadi dua yaitu untuk data testing dan data training contoh 1000 data merupakan data set dan 300 data digunakan untuk data testing kemudian 700 datanya digunakan untuk data training dimana data training tersebut digunakan untuk menentukan nilai bobot yang dimasukkan kedalam rumus regresi linier. sedangkan nilai testing akan dijadikan nili

inputan untuk rumus regresi linier. contohnya dapat dilihat pada gambar berikut :



Gambar 2.51 contoh K-fold cross validation

6. Jelaskan Apa itu decision tree dengan gambar ilustrasi contoh buatan sendiri.

Decision tree merupakan implementasi dari binari clasification dimana pada pohon keputusan akan terdapat root atau akar dan cabang cabangnya yang nilainya seperti if contoh pada root berisi nilai warna mawar, apakah merah pada cabang satu bernilai iya dan pada cabang dua bernilai tidak jika nilainya iya berarti mawar dan jika tidak maka bukan mawar. agar lebih jelas dapat dilihat pada gambar decision tree berikut:



Gambar 2.52 contoh decision tree

7. jelaskan apa itu information gain dan entropi dengan gambar ilustrasi buatan sendiri.

informasi gain merupakan informasi atau kriteria dalam pembagian sebuah objek contoh information gain pada llama yaitu leher panjang, ekor panjang, berkaki 4, buas, karnivora. untuk lebih jelasnya dapat dilihat pada gambar berikut :



Gambar 2.53 contoh information gain

sedangkan entropi merupakan ukuran keacakan dari informasi semakin tinggi entropi maka semakin sulit dalam menentukan keputusan contoh dalam menentukan satu gen semakin detail informasi maka akan semakin susah dalam menentukan keputusan.

2.4.2 Sikic-Learn

1. pada surcode pertama yang dapat dilihat pada gambar pada baris pertama di tuliskan import pandas as baso yang berarti mengimport library pandas. selanjutnya pada baris ke dua codingan tersebut berisi code tersebut terdapat variabel pecel yang berisi inisialisasi pandas dengan aksi untuk membaca vfile csv yang terdapat pada direktori pada komputer kemudian terdapat sep samadengan titik koma yang berarti pemisah field di dalam file tersebut merupakan titik koma. kemudian pada bagian akhir terdapat code len (nama variabel) yang berarti akan menghitung jumlah baris pada file tersebut. untuk hasilnya dapat dilihat pada gambar dibawah.

```

1 # -*- coding: utf-8 -*-
2 """
3 Created on Thu Mar  5 21:32:13 2020
4
5 @author: User
6 """
7
8 # load dataset (student mat pakenya)
9 import pandas as pd
10 pecel = pd.read_csv('D:\\SEMESTER 6\\Python-Artificial-
    Intelligence-Projects-for-Beginners-master\\Chapter01\\
        dataset\\student-mat.csv', sep=';')
11 print(len(pecel))

```

In [1]: runfile('D:/SEMESTER 6/wert/untitled0.py', wdir='D:/SEMESTER 6/wert')
395

Gambar 2.54 hasil

2. Pada code selanjutnya akan ditambahkan fungsi untuk lulus atau gagal yang dibuat berupa kolom kolom yang di dalamnya terdapat nilai 0 dan 1 dimana 0 berarti gagal dan 1 berarti lulus. kemudian variabel pada codingan sebelumnya masih digunakan. dimana variabel pecel digunakan karena berisi nilai file csv kemudian dilakukan ekseskuasi dengan parameter G1, G2, dan G3 dengan fungsi lambda yang berarti if bersarang atau if didalam if yang berarti nilai yang di eksekusi akan dilempar ke dalam setiap paramater sasuai dengan kriteria dan axis=1 yaitu nilai tersebut akan digunakan dari tiap baris data. sedangkan pada codingan variabel pecel di berikan aksi drop yaitu penurunan nilai pada bagian baris data. dan pada code pecel.head yaitu untuk mengeksekusi codingan sebelumnya untuk lebih jelasnya dapat dilihat pada gambar dan hasilnya seperti pada gambar berikut :

```

1 # -*- coding: utf-8 -*-
2 """
3 Created on Thu Mar  5 21:36:23 2020
4

```

```

5 @author: User
6 """
7
8 # generate binary label (pass/fail) based on G1+G2+G3
9 # (test grades, each 0-20 pts); threshold for passing is sum
10    >=30
11 pecel[ 'pass' ] = pecelapply(lambda row: 1 if (row[ 'G1' ]+row[ 'G2' ]+row[ 'G3' ]) 
12 >= 35 else 0, axis=1)
13 pecel = pecel.drop([ 'G1' , 'G2' , 'G3' ], axis=1)
14 print(pecel.head())

```

```

In [4]: runfile('D:/SEMESTER 6/wert/2.py', wdir='D:/SEMESTER 6/wert')
   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]

```

Gambar 2.55 hasil

3. pada kodingan selanjutnya diguanakan untuk menambahkan nilai numerik berupa 0 dan 1 yang dirubah dari nilai tidak dan iya hal ini merupakan fungsi dari codingan get_dummies pada baris pertama pada gambar yang nilainya diambil dari variabel pecel yang telah di dekralasikan tadi banyaknya data numerik itu sendiri tergantung pada field dari kolom yang di camtumkan pada codingan dengan catatan field tersebut harus ada dalam data csv yang di import oleh codingan pertama tadi maka hasilnya akan merubah data dalam field tersebut menjadi 0 dan 1 untuk lebih jelasnya dapat di lihat pada gambar berikut;

```

1 # -*- coding: utf-8 -*-
2 """
3 Created on Thu Mar  5 21:41:22 2020
4
5 @author: User
6 """
7
8 # use one-hot encoding on categorical columns
9 pecel = pd.get_dummies(pecel, columns=[ 'sex' , 'school' ,
10                           'address' , 'famsize' , 'Pstatus' , 'Mjob' , 'Fjob' , 'reason' ,
11                           'guardian' , 'schoolsup' , 'famsup' , 'paid' , 'activities' ,
12                           'nursery' , 'higher' , 'internet' , 'romantic' ])
13 pecel.head()

```

```

In [6]: runfile('D:/SEMESTER 6/wert/4.py', wdir='D:/SEMESTER 6/wert')
Passing: 166 out of 395 (42.03%)

```

Gambar 2.56 hasil

4. selanjutnya pada codingan selanjutnya yaitu menentukan data training dan data testing dari dataset dimana variabel pecel yang berisi data csv

tadi dibuat perbandingan 500 untuk data training dan sisanya yaitu 149 digunakan untuk data testing hal ini di lakukan pada baris ke 1 sampai ke 3 pada gambar kemudian nilai tersebut di turunkan brdasarkan baris pada data set hal itu dapat dilihat pada baris ke 4 sampai ke 9 pada gambar kemudian selanjutnya mengimport library numpy yang berguna untuk oprasi vektor dan matrix karna data diatas berupa data matrix maka library ini di gunakan. untuk hasilnya dapat dilihat pada gambar.

```

1 # -*- coding: utf-8 -*-
2 """
3 Created on Thu Mar  5 21:42:26 2020
4
5 @author: User
6 """
7
8 # shuffle rows
9 pecel = pecel.sample(frac=1)
10 # split training and testing data
11 pecel_train = pecel[:500]
12 pecel_test = pecel[500:]
13 pecel_train_att = pecel_train.drop(['pass'], axis=1)
14 pecel_train_pass = pecel_train['pass']
15 pecel_test_att = pecel_test.drop(['pass'], axis=1)
16 pecel_test_pass = pecel_test['pass']
17 pecel_att = pecel.drop(['pass'], axis=1)
18 pecel_pass = pecel['pass']
19 # number of passing students in whole dataset:
20 import numpy as np
21 print("Passing: %d out of %d (%.2f%%)" % (np.sum(pecel_pass),
   len(pecel_pass), 100*float(np.sum(pecel_pass)) / len(
   pecel_pass)))

```

| | 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 |

Gambar 2.57 hasil

- selanjutnya yaitu membuat pohon keputusan dapat lebih jelasnya dapat di lihat pada gambar. pada baris pertama yairu memasukan librari tree kemudian pada baris kedua dibuat variabel lontong dengan nilai DecisionTreeClasifier yang merupakan paket atau fungsi dari scikit-learn yang merupakan class yang mampu melakukan multi class. sedangkan max_depth=5 merupakan untuk penyesuaian data terhadap pohon keputusan itu sndiri. untuk hasilnya dapat dilihat pada gambar

```

1 # -*- coding: utf-8 -*-
2 """
3 Created on Thu Mar  5 21:56:41 2020
4

```

```

5 @author: User
6 """
7
8 # fit a decision tree
9 from sklearn import tree
10 lontong = tree.DecisionTreeClassifier(criterion="entropy",
11                                     max_depth=5)
12 lontong = lontong.fit(pecel,_train_att, pecel_train_pass)

```

In [18]: runfile('D:/SEMESTER 6/wert/4.py', wdir='D:/SEMESTER 6/wert')
Passing: 166 out of 395 (42.03%)

Gambar 2.58 hasil

6. selanjutnya yaitu pembuatan gambar dari pohon keputusan yang tadi di buta pada codingan sebelumnya pada baris pertama codingan ya itu mengimport library graphviz kemudian pada baris ke dua yaitu pem-berian nilai pada variabel baru dot data nilainya diambil dari pembuatan puhon keputusan tadi kemudian di tentukannya nilai benar dan salah dari codingan tersebut setelah itu dibuatlah sebuah variabel untuk menam-pung hasil eksekusi tersebut kemudian variabel tersebut di running untuk lebih jelasnya dapat di lihat pada gambar.kemudian hasilnya dapat dili-hat pada gambar.

```

1 # -*- coding: utf-8 -*-
2 """
3 Created on Thu Mar  5 22:00:57 2020
4
5 @author: User
6 """
7
8 # visualize tree
9 import graphviz
10 dot_data = tree.export_graphviz(lontong, out_file=None, label
11                                 ="all", impurity=False, proportion=True,
12                                 feature_names=list(
13                                   pecel_train_att), class_names=["fail", "pass"],
14                                 filled=True, rounded=True)
15 graph = graphviz.Source(dot_data)
16 graph

```

7. selanjutnya pembuatasn method untuk menyimpan data pohon dengan menarik data langsung dari pohon keputusan di buat tadi untuk code lebih jelasnya dapat dilihat pada gambar. kemudian intuk hasilnya dapat dilihat pada gambar berikut.

```

1 # -*- coding: utf-8 -*-
2 """
3 Created on Thu Mar  5 22:03:32 2020
4
5 @author: User
6 """

```

```

graph TD
    node [shape=box, style="filled, rounded", color="black", fontname=helvetica] ;
    edge [fontname=helvetica] ;
    0 [label="failures <= 0.5\nsamples = 100.0%\nvalue = [0.58, 0.42]\nclass = fail", fillcolor="#f8dcc9"] ;
    1 [label="schoolsup_no <= 0.5\nsamples = 79.0%\nvalue = [0.519, 0.481]\nclass = fail", fillcolor="#fdfef0"] ;
    0 --> 1 [label=distance:2.5, labelangle:45, headLabel="true"] ;
    2 [label="Mjob_services <= 0.5\nsamples = 10.1%\nvalue = [0.825, 0.175]\nclass = fail", fillcolor="#eb9c63"] ;
    1 --> 2 ;
    3 [label="Fedu <= 2.5\nsamples = 6.6%\nvalue = [0.923, 0.077]\nclass = fail", fillcolor="#e78c49"] ;
    2 --> 3 ;
    4 [label="famsize_LE3 <= 0.5\nsamples = 2.3%\nvalue = [0.778, 0.222]\nclass = fail", fillcolor="#eca572"] ;
    3 --> 4 ;
    5 [label="samples = 1.0%\nvalue = [0.5, 0.5]\nclass = fail", fillcolor="#ffffff"] ;
    4 --> 5 ;
    6 [label="samples = 1.3%\nvalue = [1.0, 0.0]\nclass = fail", fillcolor="#e58139"] ;
    4 --> 6 ;
    7 [label="samples = 4.3%\nvalue = [1.0, 0.0]\nclass = fail", fillcolor="#e58139"] ;
    3 --> 7 ;
    8 [label="Walc <= 3.5\nsamples = 3.5%\nvalue = [0.643, 0.357]\nclass = fail", fillcolor="#f3c7a7"] ;
    2 --> 8 ;
    9 [label="famsup_no <= 0.5\nsamples = 2.8%\nvalue = [0.545, 0.455]\nclass = fail", fillcolor="#fbeade"] ;
    8 --> 9 ;
    10 [label="samples = 2.3%\nvalue = [0.667, 0.333]\nclass = fail", fillcolor="#f2c09c"] ;
    9 --> 10 ;
    11 [label="samples = 0.5%\nvalue = [0.0, 1.0]\nclass = pass", fillcolor="#399de5"] ;
    9 --> 11 ;
    12 [label="samples = 0.8%\nvalue = [1.0, 0.0]\nclass = fail", fillcolor="#e58139"] ;
    8 --> 12 ;
    13 [label="famrel <= 1.5\nsamples = 68.9%\nvalue = [0.474, 0.526]\nclass = pass", fillcolor="#ecf5fc"] ;
    1 --> 13 ;
    14 [label="samples = 1.5%\nvalue = [0.0, 1.0]\nclass = pass", fillcolor="#399de5"] ;
    13 --> 14 ;
    15 [label="Mjob_services <= 0.5\nsamples = 67.3%\nvalue = [0.485, 0.515]\nclass = pass", fillcolor="#f3f9fd"] ;
    13 --> 15 ;
    16 [label="Mjob_health <= 0.5\nsamples = 52.7%\nvalue = [0.529, 0.471]\nclass = fail", fillcolor="#fcf1e9"] ;
    15 --> 16 ;
    17 [label="samples = 45.6%\nvalue = [0.567, 0.433]\nclass = fail", fillcolor="#f9e1d0"] ;
    16 --> 17 ;
    18 [label="samples = 7.1%\nvalue = [0.286, 0.714]\nclass = pass", fillcolor="#88c4e5"] ;
    16 --> 18 ;
    19 [label="goout <= 1.5\nsamples = 14.7%\nvalue = [0.328, 0.672]\nclass = pass", fillcolor="#99cdf2"] ;
    15 --> 19 ;
    20 [label="samples = 1.5%\nvalue = [0.833, 0.167]\nclass = fail", fillcolor="#ea9a61"] ;

```

Gambar 2.59 hasil

```

7
8 # save tree
9 tree.export_graphviz(lontong, out_file="student-performance.
    dot", label="all", impurity=False, proportion=True,
10                         feature_names=list(pecel_train_att),
    class_names=["fail", "pass"], filled=True, rounded=True)

```

8. selanjutnya pada codingan berikut yaitu digunakan untuk mencetak nilai score rata-rata dari ketepatan data yang telah diolah tadi lebih jelsnya dapat dilihat pada gambar kemudian untuk hasilnya dapat dilihat pada gambar tersebut:

```

1 # -*- coding: utf-8 -*-
2 """
3 Created on Thu Mar  5 22:03:35 2020
4
5 @author: User
6 """
7
8 lontong.score(pecel_test_att, pecel_test_pass)

```

9. selanjutnya yaitu digunakan untuk memeriksa akurasi dari ketepatan hasil pengolahan data tersebut maka akan didapat nilai rata-rata 60

```

graph TD
    node [shape=box, style="filled, rounded", color="black", fontname=helvetica];
    edge [fontname=helvetica];
    0 [label="failures <= 0.5\nsamples = 100.0%\nvalue = [0.58, 0.42]\nclass = fail", fillcolor="#f8dc99"];
    1 [label="schoolsup_no <= 0.5\nsamples = 79.0%\nvalue = [0.519, 0.481]\nclass = fail", fillcolor="#fdfe00"];
    0 --> 1 [label=distance:2.5, labelangle:45, headLabel="true"];
    2 [label="Mjob_services <= 0.5\nsamples = 10.1%\nvalue = [0.825, 0.175]\nclass = fail", fillcolor="#eb9c63"];
    1 --> 2 ;
    3 [label="Fedu <= 2.5\nsamples = 6.6%\nvalue = [0.923, 0.077]\nclass = fail", fillcolor="#e78c49"];
    2 --> 3 ;
    4 [label="famsize_LE3 <= 0.5\nsamples = 2.3%\nvalue = [0.778, 0.222]\nclass = fail", fillcolor="#eca572"];
    3 --> 4 ;
    5 [label="samples = 1.0%\nvalue = [0.5, 0.5]\nclass = fail", fillcolor="#ffffff"];
    4 --> 5 ;
    6 [label="samples = 1.3%\nvalue = [1.0, 0.0]\nclass = fail", fillcolor="#e58139"];
    4 --> 6 ;
    7 [label="samples = 4.3%\nvalue = [1.0, 0.0]\nclass = fail", fillcolor="#e58139"];
    3 --> 7 ;
    1 [label="Walc <= 3.5\nsamples = 3.5%\nvalue = [0.643, 0.357]\nclass = fail", fillcolor="#f3c7a7"];
    2 --> 8 ;
    9 [label="famsup_no <= 0.5\nsamples = 2.8%\nvalue = [0.545, 0.455]\nclass = fail", fillcolor="#fbeade"];
    8 --> 9 ;
    10 [label="samples = 2.3%\nvalue = [0.667, 0.333]\nclass = fail", fillcolor="#f2c09c"];
    9 --> 10 ;
    11 [label="samples = 0.5%\nvalue = [0.0, 1.0]\nclass = pass", fillcolor="#399de5"];
    9 --> 11 ;
    12 [label="samples = 0.8%\nvalue = [1.0, 0.0]\nclass = fail", fillcolor="#e58139"];
    8 --> 12 ;
    13 [label="famrel <= 1.5\nsamples = 68.9%\nvalue = [0.474, 0.526]\nclass = pass", fillcolor="#ecf5fc"];
    1 --> 13 ;
    14 [label="samples = 1.5%\nvalue = [0.0, 1.0]\nclass = pass", fillcolor="#399de5"];
    13 --> 14 ;
    15 [label="Mjob_services <= 0.5\nsamples = 67.3%\nvalue = [0.485, 0.515]\nclass = pass", fillcolor="#f3f9fd"];
    13 --> 15 ;
    16 [label="Mjob_health <= 0.5\nsamples = 52.7%\nvalue = [0.529, 0.471]\nclass = fail", fillcolor="#fcf1e9"];
    15 --> 16 ;
    17 [label="samples = 45.6%\nvalue = [0.567, 0.433]\nclass = fail", fillcolor="#f9e1d0"];
    16 --> 17 ;
    18 [label="samples = 7.1%\nvalue = [0.286, 0.714]\nclass = pass", fillcolor="#88c4ef"];
    16 --> 18 ;
    19 [label="goout <= 1.5\nsamples = 14.7%\nvalue = [0.328, 0.672]\nclass = pass", fillcolor="#99cdf2"];
    15 --> 19 ;
    20 [label="samples = 1.5%\nvalue = [0.833, 0.167]\nclass = fail", fillcolor="#ea9a61"];

```

Gambar 2.60 hasil

persen dari hasil pengolahan data tersebut untuk lebih jelasnya dapat di lihat pada gambar pada codingan tersebut pada baris ke satu melakukan import library dari sklern kemudian pada baris selanjutnya mengisi nilai skor dengan nilai pada variabel lontong setelah hal tersebut dilakukan kemudian data tersebut di eksekusi. berikut merupakan hasil dari code tersebut dapat dilihat pada gambar.

```

1 # -*- coding: utf-8 -*-
2 """
3 Created on Thu Mar  5 22:03:36 2020
4
5 @author: User
6 """
7
8 from sklearn.model_selection import cross_val_score
9 scores = cross_val_score(lontong, pecel_att, pecel_pass, cv
10                         =5)
11 # show average score and +/- two standard deviations away
12 #(covering 95% of scores)
13 print("Accuracy: %0.2f (+/- %0.2f)" % (scores.mean(), scores.
14                                         std() * 2))

```

```
In [61]: runfile('D:/SEMESTER 6/wert/9.py', wdir='D:/SEMESTER 6/wert')
Accuracy: 0.59 (+/- 0.05)
```

Gambar 2.61 hasil

10. membuat rank akurasi dari 1 sampai 20 untuk melihat akurasi data apakah datatersebut terdapat di rata rata 60 persen atau tidak dengan cara membuat lagi variabel baru dengan nilai tree diadalamnya jadi hampir mirip seperti membuat pohon keputusan namun ini langsung dalam bentuk rata rata akurasi yanlebih spesifik untuk lebih jelasnya dapat dilihat pada gambar codingan berikut. dan untuk hasilnya dapat dilihat pada gambar berikut.

```
1 # -*- coding: utf-8 -*-
2 """
3 Created on Thu Mar  5 22:03:38 2020
4
5 @author: User
6 """
7
8 for max_depth in range(1, 100):
9     lontong = tree.DecisionTreeClassifier(criterion="entropy",
10                                         , max_depth=max_depth)
11     scores = cross_val_score(lontong, pecel_att, pecel_pass,
12                               cv=5)
13     print("Max depth: %d, Accuracy: %0.2f (+/- %0.2f)" % (
14         max_depth, scores.mean(), scores.std() * 2))
```

```
Max depth: 1, Accuracy: 0.58 (+/- 0.01)
Max depth: 2, Accuracy: 0.58 (+/- 0.06)
Max depth: 3, Accuracy: 0.56 (+/- 0.06)
Max depth: 4, Accuracy: 0.56 (+/- 0.08)
Max depth: 5, Accuracy: 0.58 (+/- 0.05)
Max depth: 6, Accuracy: 0.62 (+/- 0.05)
Max depth: 7, Accuracy: 0.58 (+/- 0.05)
Max depth: 8, Accuracy: 0.60 (+/- 0.07)
Max depth: 9, Accuracy: 0.58 (+/- 0.10)
Max depth: 10, Accuracy: 0.57 (+/- 0.04)
Max depth: 11, Accuracy: 0.58 (+/- 0.06)
Max depth: 12, Accuracy: 0.57 (+/- 0.08)
Max depth: 13, Accuracy: 0.57 (+/- 0.10)
Max depth: 14, Accuracy: 0.55 (+/- 0.07)
Max depth: 15, Accuracy: 0.58 (+/- 0.08)
Max depth: 16, Accuracy: 0.58 (+/- 0.05)
Max depth: 17, Accuracy: 0.57 (+/- 0.09)
Max depth: 18, Accuracy: 0.57 (+/- 0.11)
Max depth: 19, Accuracy: 0.57 (+/- 0.09)
Max depth: 20, Accuracy: 0.57 (+/- 0.07)
Max depth: 21, Accuracy: 0.57 (+/- 0.06)
Max depth: 22, Accuracy: 0.58 (+/- 0.07)
Max depth: 23, Accuracy: 0.58 (+/- 0.10)
```

Gambar 2.62 hasil

11. untuk selanjutnya yaitu menentukan nilai untuk grafik hampirsama dengan nilai akurasi tadi pertama tentukan rank atau batasan nilai itu disini batasannya di mulai dari 1 sampai 20 dengan menggunakan nilai tree tadi maka hasilnya dapat ditentuka kemudian buat variabel i untuk penomoran tiap record yang keluar atau record hadil dari eksekusi tree tersebut. un-

tuk lebih jelasnya dapat dilihat pada gambar dan untuk hasilnya dapat dilihat pada gambar.

```

1 # -*- coding: utf-8 -*-
2 """
3 Created on Thu Mar  5 22:07:50 2020
4
5 @author: User
6 """
7
8 depth_acc = np.empty((19,3), float)
9 i = 0
10 for max_depth in range(1, 20):
11     lontong = tree.DecisionTreeClassifier(criterion="entropy",
12                                            max_depth=max_depth)
13     scores = cross_val_score(lontong, pecel_att, pecel_pass,
14                               cv=5)
15     depth_acc[i,0] = max_depth
16     depth_acc[i,1] = scores.mean()
17     depth_acc[i,2] = scores.std() * 2
18     i += 1
19
20 print(depth_acc)

```

```

[[1.0000000e+00 5.79746835e-01 1.01265823e-02]
 [2.0000000e+00 5.69620253e-01 2.77327877e-02]
 [3.0000000e+00 5.82278481e-01 7.33740595e-02]
 [4.0000000e+00 5.77215190e-01 3.03797468e-02]
 [5.0000000e+00 5.84810127e-01 9.39100607e-02]
 [6.0000000e+00 5.56962025e-01 1.01265823e-01]
 [7.0000000e+00 5.92405063e-01 6.28337906e-02]
 [8.0000000e+00 6.0000000e-01 6.71721477e-02]
 [9.0000000e+00 5.79746835e-01 6.86818266e-02]
 [1.0000000e+01 6.10126582e-01 7.74534610e-02]
 [1.1000000e+01 6.07594937e-01 5.77304013e-02]
 [1.2000000e+01 5.94936709e-01 9.47255035e-02]
 [1.3000000e+01 5.97468354e-01 5.16356406e-02]
 [1.4000000e+01 5.82278481e-01 5.77304013e-02]
 [1.5000000e+01 5.77215190e-01 6.11799796e-02]
 [1.6000000e+01 5.82278481e-01 7.67886121e-02]
 [1.7000000e+01 5.82278481e-01 7.51007442e-02]
 [1.8000000e+01 6.02531646e-01 5.68353021e-02]
 [1.9000000e+01 5.92405063e-01 7.05234849e-02]]

```

Gambar 2.63 hasil

12. terakhir yaitu pembuatan grafik untuk pembuatan grafik diambil data dari codingan sebelumnya yang 20 record tadi dengan cara mengimport libray matplotlib.pyplot yang di inisialisasi menjadi bakwankemudian inisialisasi tersebut di eksekusi. untuk lebih jelasnya codingannya seperti gambar dan untuk hasilnya seperti gambar berikut.

```

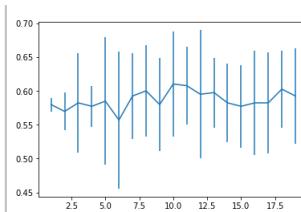
1 # -*- coding: utf-8 -*-
2 """
3 Created on Thu Mar  5 22:07:50 2020
4
5 @author: User
6 """
7
8 import matplotlib.pyplot as plt
9 fig, ax = plt.subplots()

```

```

10 ax.errorbar(depth_acc[:,0], depth_acc[:,1], yerr=depth_acc
11 [:,2])
plt.show()

```



Gambar 2.64 hasil

2.4.3 Penanganan Error

1. skrinsut error

```

In [2]: runfile('D:/SEMESTER 6/wert/2.py', wdir='D:/SEMESTER 6/wert')
Traceback (most recent call last):

  File "<ipython-input-2-4b3c06f7fd19>", line 1, in <module>
    runfile('D:/SEMESTER 6/wert/2.py', wdir='D:/SEMESTER 6/wert')

  File "C:\Users\User\Anaconda3\lib\site-packages\spyder_kernels\customize
\spydercustomize.py", line 827, in runfile
    execfile(filename, namespace)

  File "C:\Users\User\Anaconda3\lib\site-packages\spyder_kernels\customize
\spydercustomize.py", line 110, in execfile
    exec(compile(f.read(), filename, 'exec'), namespace)

  File "D:/SEMESTER 6/wert/2.py", line 10, in <module>
    pecel['pass'] = pecelapply(lambda row: 1 if (row['F1']+row['F2']+row['F3'])

NameError: name 'pecelapply' is not defined

```

Gambar 2.65 Error

2. kode error dan jenis errornya .

```

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

```

NameError: name pecelapply is not defined

3. Solusi pemecahan masalah

pada codingan tersebut error karena pecelapply tergabung, seharusnya dipisahkan oleh titik



Date: 2020-03-05

PLAGIARISM SCAN REPORT



Exclude Url : None

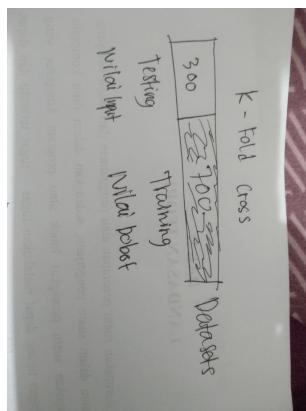
Gambar 2.66 Error

2.4.4 Plagiat

2.5 1174039 Liyana Majdah Rahma

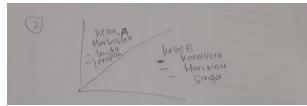
2.5.1 Teori

1. Jelaskan Apa Itu binari classification dilengkapi ilustrasi gambar sendiri.
Klasifikasi biner merupakan tugas yang digunakan untuk mengklasifikasi suatu elemen-elemen dari himpunan tertentu yang terdiri dari dua kelompok yang ditentukan berdasarkan klasifikasi.

**Gambar 2.67** contoh K-fold cross validation

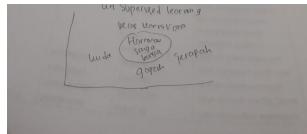
2. Jelaskan Apaitu supervised learning , unsupervised learning dan clustering dengan ilustrasi gambar sendiri.

supervised learning merupakan tipe learning dimana terdapat sebuah metode pendekatan yang mempunyai variable input dan output, serta terdapat variable yang ditargetkan dari pendekatan ini adalah mengelompokan suatu data ke data yang sudah ada sebelumnya. Dimana terdapat kelas A itu dikategorikan sebagai hewan herbivora seperti kuda, dan jerapah. Sedangkan kelas B dikategorikan sebagai hewan karnivora seperti harimau dan singa. Dilihat pada gambar.



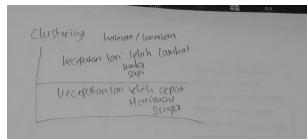
Gambar 2.68 contoh supervised learning

unsupervised merupakan tipe learning yang tidak memiliki data latih sehingga data yang sudah ada, kita kelompokkan data tersebut menjadi dua bagian ataupun menjadi tiga bagian. Untuk lebih jelasnya dapat dilihat pada gambar.



Gambar 2.69 contoh unsupervised learning

clustering merupakan proses yang mengklasifikasi berdasarkan suatu parameter dalam pententuannya. Untuk lebih jelasnya dilihat pada gambar berikut



Gambar 2.70 contoh clustering

3. Jelaskan apa itu evaluasi dan akurasi dan disertai ilustrasi contoh dengan gambar sendiri.

evaluasi merupakan suatu kegiatan yang dilakukan dari pengamatan berbagai macam bukti untuk mengukur dampak efektifitas dari objek atau proses yang berkaitan dengan spesifikasi yang telah ditentukan. sedangkan akurasi merupakan bagian dari evaluasi yang merupakan ketepatan

data terhadap suatu objek yang memiliki kriteria. Dapat dilihat pada gambar berikut

| Evaluasi dan Akurasi | |
|----------------------|--------------------|
| Ojeng | Kemiripan 2 klas |
| Burung | Kemiripan 2 burung |
| Elang | Kemiripan 2 elang |
| Elang | Kemiripan 2 singa |
| Ojeng | Maximal 2 singa |
| Ojeng | Maximal 2 burung |
| Ojeng | Maximal 2 elang |

Gambar 2.71 contoh evaluasi dan akurasi

4. Jelaskan bagaimana cara membuat Confusion Matrix, Buat confusion matrix sendiri.

Pada confusion matrix menentukan parameter yang akan di evaluasi contohnya elang,burung merpati,burung kakatua dengan tabel baris dan kolom berjumlah tida kemudian ditentukan dengan nilai miring pada setiap kolom saya beri nilai 12 dengan ketentuan setiap baris harus bernilai 12 jika kolom lain harus jumlah 15 jika tidak berarti data tidak akurat. Dapat dilihat pada gambar berikut

| Confusion Matrix | | | |
|------------------|--------|-------|---------|
| Ojeng | Burung | Elang | Merpati |
| Ojeng | 12 | 12 | 12 |
| Burung | 6 | 3 | 3 |
| Elang | 5 | 6 | 1 |
| Merpati | 3 | 7 | 2 |
| Confusion Matrix | | | |

Gambar 2.72 contoh Confusion Matrix

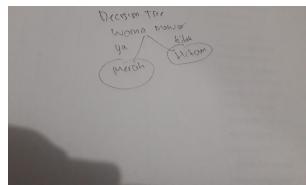
5. Jelaskan bagaimana K-fold cross validation bekerja dengan gambar ilustrasi contoh buatan sendiri. K-fold Cross Validation merupakan cara untuk melatih suatu mesin dimana di dalamnya terdapat data set yang dibagi menjadi dua yaitu untuk data testing dan data training contoh 1000 data merupakan data set dan 300 data digunakan untuk data testing kemudian 700 datanya. Sedangkan nilai testing akan dijadikan nilai inputan untuk rumus regresi linier. contohnya dapat dilihat pada gambar berikut :

| K - Fold Cross | |
|------------------------|-------------------------|
| 1000 | 600 |
| Testing Nilai Input | Training Nilai Label |
| | Bila Saja |

Gambar 2.73 contoh K-fold cross validation

6. Jelaskan Apa itu decision tree dengan gambar ilustrasi contoh buatan sendiri.

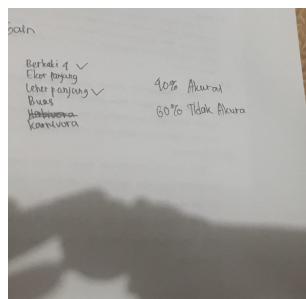
Decision tree merupakan implementasi dari binari clasification dimana pada pohon keputusan akan terdapat root atau akar dan cabang cabangnya yang nilainya seperti if contoh jika pada root berisi nilai warna mawar, apakah merah pada cabang satu bernilai iya dan pada cabang dua bernilai tidak jika nilainya iya berarti mawar dan jika tidak maka bukan mawar. agar lebih jelas dapat dilihat pada gambar decision tree berikut:



Gambar 2.74 contoh decision tree

7. jelaskan apa itu information gain dan entropi dengan gambar ilustrasi buatan sendiri.

information gain merupakan kriteria yang terdapat dalam pembagian sebuah objek seperti gigi tajam, berkaki 4, pemakan daging termasuk karnivora. untuk lebih jelasnya dapat dilihat pada gambar berikut :



Gambar 2.75 contoh information gain

sedangkan entropi merupakan ukuran keacakan dari informasi semakin tinggi entropi maka semakin sulit dalam menentukan keputusan contoh dalam menentukan satu gen semakin detail informasi maka akan semakin susah dalam menentukan keputusan.

2.5.2 Sikic-Learn

1. pada surcode pertama yang dapat dilihat pada gambar pada baris pertama di tuliskan import pandas as baso yang berarti mengimport library pandas. selanjutnya pada baris ke dua codingan tersebut berisi code tersebut terdapat variabel pecel yang berisi inisialisasi pandas dengan aksi untuk membaca vfile csv yang terdapat pada direktori pada komputer kemudian terdapat sep samadengan titik koma yang berarti pemisah field di dalam file tersebut merupakan titik koma. kemudian pada bagian akhir terdapat code len (nama variabel) yang berarti akan menghitung jumlah baris pada file tersebut. untuk hasilnya dapat dilihat pada gambar dibawah.

```

1 # -*- coding: utf-8 -*-
2 """
3 Created on Sat Mar  7 22:56:28 2020
4
5 @author: Liyana
6 """
7
8 # load dataset (student mat pakenya)
9 import pandas as pd
10 jatibarang = pd.read_csv('D:\mata kuliah poltekppos\Python-
    Artificial-Intelligence-Projects-for-Beginners-master\
    Chapter01\dataset\student-mat.csv', sep=';')
11 print(len(jatibarang))

```



Gambar 2.76 hasil

2. Pada code selanjutnya akan ditambahkan fungsi untuk lulus atau gagal yang dibuat berupa kolom kolom yang di dalamnya terdapat nilai 0 dan 1 dimana 0 berarti gagal dan 1 berarti lulus. kemudian variabel pada codingan sebelumnya masih digunakan. dimana variabel pecel digunakan karena berisi nilai file csv kemudian dilakukan ekseskuji dengan parameter G1, G2, dan G3 dengan fungsi lambda yang berarti if bersarang atau if didalam if yang berarti nilai yang di eksekusi akan dilempar ke dalam setiap paramater sesuai dengan kriteria dan axis=1 yaitu nilai tersebut akan digunakan dari tiap baris data. sedangkan pada codingan variabel pecel di berikan aksi drop yaitu penurunan nilai pada bagian baris data. dan pada code pecel.head yaitu untuk mengeksekusi codingan sebelumnya untuk lebih jelasnya dapat dilihat pada gambar dan hasilnya seperti pada gambar berikut :

```

1 # -*- coding: utf-8 -*-
2 """
3 Created on Sat Mar  7 22:41:20 2020
4
5 @author: Liyana

```

```

6 """
7
8 # generate binary label (pass/fail) based on G1+G2+G3
9 # (test grades, each 0–20 pts); threshold for passing is sum
10 >=30
11 jatibarang[ 'pass' ] = jatibarang .apply(lambda row: 1 if (row[ 'G1' ]+row[ 'G2' ]+row[ 'G3' ]) 
12 >= 35 else 0, axis=1)
13 jatibarang = jatibarang .drop([ 'G1' , 'G2' , 'G3' ], axis=1)
14 print(jatibarang .head())

```

| SCHOOL | sex | age | address | famsize | Pstatus | Mjob | Walc | health | absences | pass |
|--------|-----|-----|---------|---------|---------|------|------|--------|----------|------|
| G1 | F | 17 | U | GT3 ... | L | T | 3 | 4 | 0 | |
| G2 | F | 15 | U | LE3 ... | X | T | 5 | 10 | 0 | |
| G3 | F | 13 | U | LE3 ... | X | T | 5 | 7 | 1 | |
| G4 | F | 16 | U | GT3 ... | X | T | 5 | 6 | 0 | |

Gambar 2.77 hasil

3. pada kodingan selanjutnya diguanakan untuk menambahkan nilai numerik berupa 0 dan 1 yang dirubah dari nilai tidak dan iya hal ini merupakan fungsi dari codingan get_dummies pada baris pertama pada gambar yang nilainya diambil dari variabel pecel yang telah di dekralasikan tadi banyaknya data numerik itu sendiri tergantung pada field dari kolom yang di camtumkan pada codingan dengan catatan field tersebut harus ada dalam data csv yang di import oleh codingan pertama tadi maka hasilnya akam merubah data dalam field tersebut menjadi 0 dan 1 untuk lebih jelasnya dapat di lihat pada gambar berikut;

```

1 # -*- coding: utf-8 -*-
2 """
3 Created on Sat Mar  7 22:54:20 2020
4
5 @author: Liyana
6 """
7
8 # use one-hot encoding on categorical columns
9 jatibarang = pd.get_dummies(jatibarang , columns=[ 'sex' ,
10 'school' , 'address' , 'famsize' , 'Pstatus' , 'Mjob' , 'Fjob' ,
11 'reason' , 'guardian' , 'schoolsups' , 'famsup' , 'paid' , 'activities' ,
12 'nursery' , 'higher' , 'internet' , 'romantic' ])
13 jatibarang .head()

```

```
[15 rows x 31 columns]
Passing: 166 out of 395 (42.03%)
Traceback (most recent call last):
```

Gambar 2.78 hasil

4. selanjutnya pada codingan selanjutnya yaitu menentukan data training dan data testing dari dataset dimana variabel pecel yang berisi data csv tadi dibuat perbandingan 500 untuk data training dan sisanya yaitu 149 digunakan untuk data testing hal ini di lakukan pada baris ke 1 sampai ke 3 pada gambar kemudian nilai tersebut di turunkan brdasarkan baris

pada data set hal itu dapat dilihat pada baris ke 4 sampai ke 9 pada gambar kemudian selanjutnya mengimport library numpy yang berguna untuk oprasi vektor dan matrix karna data diatas berupa data matrix maka library ini di gunakan. untuk hasilnya dapat dilihat pada gambar.

```

1 # -*- coding: utf-8 -*-
2 """
3 Created on Sat Mar  7 22:57:59 2020
4
5 @author: Liyana
6 """
7
8 # shuffle rows
9 jatibarang = jatibarang.sample(frac=1)
10 # split training and testing data
11 jatibarang_train = jatibarang[:500]
12 jatibarang_test = jatibarang[500:]
13 jatibarang_train.att = jatibarang_train.drop(['pass'], axis=1)
14 jatibarang_train.pass = jatibarang_train['pass']
15 jatibarang_test.att = jatibarang_test.drop(['pass'], axis=1)
16 jatibarang_test.pass = jatibarang_test['pass']
17 jatibarang.att = jatibarang.drop(['pass'], axis=1)
18 jatibarang.pass = jatibarang['pass']
19 # number of passing students in whole dataset:
20 import numpy as np
21 print("Passing: %d out of %d (%.2f%%)" % (np.sum(
    jatibarang.pass), len(jatibarang.pass), 100*float(np.sum(
        jatibarang.pass)) / len(jatibarang.pass)))

```

| | school | sex | age | address | family size | ... | Dalc | Walc | health | absences | pass |
|---|--------|-----|-----|---------|-------------|-----|------|------|--------|----------|------|
| 0 | GP | F | 18 | U | G1 | ... | 1 | 1 | 3 | 6 | 0 |
| 1 | GP | F | 19 | U | G1 | ... | 1 | 1 | 3 | 6 | 0 |
| 2 | GP | F | 15 | U | L1 | ... | 2 | 1 | 3 | 10 | 0 |
| 3 | GP | F | 15 | U | G1 | ... | 1 | 1 | 5 | 2 | 1 |
| 4 | GP | F | 16 | U | G1 | ... | 1 | 2 | 5 | 4 | 0 |

Gambar 2.79 hasil

- selanjutnya yaitu membuat pohon keputusan dapat lebih jelasnya dapat di lihat pada gambar. pada baris pertama yairu memasukan librari tree kemudian pada baris kedua dibuat variabel lontong dengan nilai DecisionTreeClasifier yang merupakan paket atau fungsi dari scikit-learn yang merupakan class yang mampu melakukan multi class. sedangkan max_depth=5 merupakan untuk penyesuaian data terhadap pohon keputusan itu sndiri. untuk hasilnya dapat dilihat pada gambar

```

1 # -*- coding: utf-8 -*-
2 """
3 Created on Sat Mar  7 23:01:08 2020
4
5 @author: Liyana
6 """
7
8 # fit a decision tree
9 from sklearn import tree

```

```
10 lobener = tree.DecisionTreeClassifier(criterion="entropy",
   max_depth=5)
11 lobener = lobener.fit(jatibarang_train_att,
   jatibarang_train_pass)
```

[5 rows x 31 columns]
Passing: 166 out of 395 (42.03%)

Gambar 2.80 hasil

6. selanjutnya yaitu pembuatan gambar dari pohon keputusan yang tadi di buta pada codingan sebelumnya pada baris pertama codingan ya itu mengimport library graphviz kemudian pada baris ke dua yaitu pem-berian nilai pada variabel baru dot data nilainya diambil dari pembuatan puhon keputusan tadi kemudian di tentukannya nilai benar dan salah dari codingan tersebut setelah itu dibuatlah sebuah variabel untuk menam-pung hasil eksekusi tersebut kemudian variabel tersebut di running untuk lebih jelasnya dapat di lihat pada gambar.kemudian hasilnya dapat dili-hat pada gambar.

```
1 # -*- coding: utf-8 -*-
2 """
3 Created on Sat Mar  7 23:05:14 2020
4
5 @author: Liyana
6 """
7
8 # visualize tree
9 import graphviz
10 dot_data = tree.export_graphviz(lobener, out_file=None, label=
11     ="all", impurity=False, proportion=True,
12                                     feature_names=list(
13                                         jatibarang_train_att), class_names=["fail", "pass"],
14                                         filled=True, rounded=True)
15 graph = graphviz.Source(dot_data)
16 graph
```

Gambar 2.81 hasil

7. selanjutnya pembuatan method untuk menyimpan data pohon dengan menarik data langsung dari pohon keputusan di buat tadi untuk code lebih jelasnya dapat dilihat pada gambar. kemudian intuk hasilnya dapat dilihat pada gambar berikut.

```

1 # -*- coding: utf-8 -*-
2 """
3 Created on Sat Mar  7 23:08:17 2020
4
5 @author: Liyana
6 """
7
8 tree.export_graphviz(lobener, out_file="student-performance.
9         dot", label="all", impurity=False, proportion=True,
10                     feature_names=list(jatibarang_train_att)
11                     , class_names=["fail", "pass"], filled=True, rounded=True)
```

```

graph TD
    root[jatibarang] --> 1{Jika <math>x_1 \leq 0.48</math>}
    1 --> 2{Jika <math>x_2 \leq 0.38</math>}
    2 --> 3{Jika <math>x_3 \leq 0.38</math>}
    3 --> 4{Jika <math>x_4 \leq 0.48</math>}
    4 --> 5{Jika <math>x_5 \leq 0.5</math>}
    5 --> 6{Jika <math>x_6 \leq 0.48</math>}
    6 --> 7{Jika <math>x_7 \leq 0.48</math>}
    7 --> 8{Jika <math>x_8 \leq 0.48</math>}
    8 --> 9{Jika <math>x_9 \leq 0.48</math>}
    9 --> 10{Jika <math>x_{10} \leq 0.48</math>}
    10 --> 11{Jika <math>x_{11} \leq 0.48</math>}
    11 --> 12{Jika <math>x_{12} \leq 0.48</math>}
    12 --> 13{Jika <math>x_{13} \leq 0.48</math>}
    13 --> 14{Jika <math>x_{14} \leq 0.48</math>}
    14 --> 15{Jika <math>x_{15} \leq 0.48</math>}
    15 --> 16{Jika <math>x_{16} \leq 0.48</math>}
    16 --> 17{Jika <math>x_{17} \leq 0.48</math>}
    17 --> 18{Jika <math>x_{18} \leq 0.48</math>}
    18 --> 19{Jika <math>x_{19} \leq 0.48</math>}
    19 --> 20{Jika <math>x_{20} \leq 0.48</math>}
    20 --> 21{Jika <math>x_{21} \leq 0.48</math>}
    21 --> 22{Jika <math>x_{22} \leq 0.48</math>}
    22 --> 23{Jika <math>x_{23} \leq 0.48</math>}
    23 --> 24{Jika <math>x_{24} \leq 0.48</math>}
    24 --> 25{Jika <math>x_{25} \leq 0.48</math>}
    25 --> 26{Jika <math>x_{26} \leq 0.48</math>}
    26 --> 27{Jika <math>x_{27} \leq 0.48</math>}
    27 --> 28{Jika <math>x_{28} \leq 0.48</math>}
    28 --> 29{Jika <math>x_{29} \leq 0.48</math>}
    29 --> 30{Jika <math>x_{30} \leq 0.48</math>}
    30 --> 31{Jika <math>x_{31} \leq 0.48</math>}
    31 --> 32{Jika <math>x_{32} \leq 0.48</math>}
    32 --> 33{Jika <math>x_{33} \leq 0.48</math>}
    33 --> 34{Jika <math>x_{34} \leq 0.48</math>}
    34 --> 35{Jika <math>x_{35} \leq 0.48</math>}
    35 --> 36{Jika <math>x_{36} \leq 0.48</math>}
    36 --> 37{Jika <math>x_{37} \leq 0.48</math>}
    37 --> 38{Jika <math>x_{38} \leq 0.48</math>}
    38 --> 39{Jika <math>x_{39} \leq 0.48</math>}
    39 --> 40{Jika <math>x_{40} \leq 0.48</math>}
    40 --> 41{Jika <math>x_{41} \leq 0.48</math>}
    41 --> 42{Jika <math>x_{42} \leq 0.48</math>}
    42 --> 43{Jika <math>x_{43} \leq 0.48</math>}
    43 --> 44{Jika <math>x_{44} \leq 0.48</math>}
    44 --> 45{Jika <math>x_{45} \leq 0.48</math>}
    45 --> 46{Jika <math>x_{46} \leq 0.48</math>}
    46 --> 47{Jika <math>x_{47} \leq 0.48</math>}
    47 --> 48{Jika <math>x_{48} \leq 0.48</math>}
    48 --> 49{Jika <math>x_{49} \leq 0.48</math>}
    49 --> 50{Jika <math>x_{50} \leq 0.48</math>}
    50 --> 51{Jika <math>x_{51} \leq 0.48</math>}
    51 --> 52{Jika <math>x_{52} \leq 0.48</math>}
    52 --> 53{Jika <math>x_{53} \leq 0.48</math>}
    53 --> 54{Jika <math>x_{54} \leq 0.48</math>}
    54 --> 55{Jika <math>x_{55} \leq 0.48</math>}
    55 --> 56{Jika <math>x_{56} \leq 0.48</math>}
    56 --> 57{Jika <math>x_{57} \leq 0.48</math>}
    57 --> 58{Jika <math>x_{58} \leq 0.48</math>}
    58 --> 59{Jika <math>x_{59} \leq 0.48</math>}
    59 --> 60{Jika <math>x_{60} \leq 0.48</math>}
    60 --> 61{Jika <math>x_{61} \leq 0.48</math>}
    61 --> 62{Jika <math>x_{62} \leq 0.48</math>}
    62 --> 63{Jika <math>x_{63} \leq 0.48</math>}
    63 --> 64{Jika <math>x_{64} \leq 0.48</math>}
    64 --> 65{Jika <math>x_{65} \leq 0.48</math>}
    65 --> 66{Jika <math>x_{66} \leq 0.48</math>}
    66 --> 67{Jika <math>x_{67} \leq 0.48</math>}
    67 --> 68{Jika <math>x_{68} \leq 0.48</math>}
    68 --> 69{Jika <math>x_{69} \leq 0.48</math>}
    69 --> 70{Jika <math>x_{70} \leq 0.48</math>}
    70 --> 71{Jika <math>x_{71} \leq 0.48</math>}
    71 --> 72{Jika <math>x_{72} \leq 0.48</math>}
    72 --> 73{Jika <math>x_{73} \leq 0.48</math>}
    73 --> 74{Jika <math>x_{74} \leq 0.48</math>}
    74 --> 75{Jika <math>x_{75} \leq 0.48</math>}
    75 --> 76{Jika <math>x_{76} \leq 0.48</math>}
    76 --> 77{Jika <math>x_{77} \leq 0.48</math>}
    77 --> 78{Jika <math>x_{78} \leq 0.48</math>}
    78 --> 79{Jika <math>x_{79} \leq 0.48</math>}
    79 --> 80{Jika <math>x_{80} \leq 0.48</math>}
    80 --> 81{Jika <math>x_{81} \leq 0.48</math>}
    81 --> 82{Jika <math>x_{82} \leq 0.48</math>}
    82 --> 83{Jika <math>x_{83} \leq 0.48</math>}
    83 --> 84{Jika <math>x_{84} \leq 0.48</math>}
    84 --> 85{Jika <math>x_{85} \leq 0.48</math>}
    85 --> 86{Jika <math>x_{86} \leq 0.48</math>}
    86 --> 87{Jika <math>x_{87} \leq 0.48</math>}
    87 --> 88{Jika <math>x_{88} \leq 0.48</math>}
    88 --> 89{Jika <math>x_{89} \leq 0.48</math>}
    89 --> 90{Jika <math>x_{90} \leq 0.48</math>}
    90 --> 91{Jika <math>x_{91} \leq 0.48</math>}
    91 --> 92{Jika <math>x_{92} \leq 0.48</math>}
    92 --> 93{Jika <math>x_{93} \leq 0.48</math>}
    93 --> 94{Jika <math>x_{94} \leq 0.48</math>}
    94 --> 95{Jika <math>x_{95} \leq 0.48</math>}
    95 --> 96{Jika <math>x_{96} \leq 0.48</math>}
    96 --> 97{Jika <math>x_{97} \leq 0.48</math>}
    97 --> 98{Jika <math>x_{98} \leq 0.48</math>}
    98 --> 99{Jika <math>x_{99} \leq 0.48</math>}
    99 --> 100{Jika <math>x_{100} \leq 0.48</math>}
    100 --> 101{Jika <math>x_{101} \leq 0.48</math>}
```

Gambar 2.82 hasil

8. selanjutnya pada codingan berikut yaitu digunakan untuk mencetak nilai score rata-rata dari ketepatan data yang telah diolah tadi lebih jelasnya dapat dilihat pada gambar kemudian untuk hasilnya dapat dilihat pada gambar tersebut:

```

1 # -*- coding: utf-8 -*-
2 """
3 Created on Sat Mar  7 23:09:11 2020
4
5 @author: Liyana
6 """
7
8 lobener.score(jatibarang_test_att , jatibarang_test_pass)
```

9. selanjutnya yaitu digunakan untuk memeriksa akurasi dari ketepatan hasil pengolahan data tersebut maka akan didapat nilai rata-rata 60 persen dari hasil pengolahan data tersebut untuk lebih jelasnya dapat dilihat pada gambar pada codingan tersebut pada baris ke satu melakukan import library dari sklern kemudian pada baris selanjutnya mengisi nilai skor dengan nilai pada variabel lontong setelah hal tersebut dilakukan kemudian data tersebut di eksekusi. berikut merupakan hasil dari code tersebut dapat dilihat pada gambar.

```

1 # -*- coding: utf-8 -*-
2 """
3 Created on Sat Mar  7 23:09:58 2020
4
5 @author: Liyana
6 """
7
8 from sklearn.model_selection import cross_val_score
9 scores = cross_val_score(lobener, jatibarang_att,
10                         jatibarang_pass, cv=5)
11 # show average score and +/- two standard deviations away
12 #(covering 95% of scores)
13 print("Accuracy: %0.2f (+/- %0.2f)" % (scores.mean(), scores.
14                                         std() * 2))

```

Accuracy: 0.59 (+/- 0.05)

Gambar 2.83 hasil

10. membuat rank akurasi dari 1 sampai 20 untuk melihat akurasi data apakah datatersebut terdapat di rata rata 60 persen atau tidak dengan cara membuat lagi variabel baru dengan nilai tree diadalamnya jadi ham-pir mirip seperti membuat pohon keputusan namun ini langsung dalam bentuk rata rata akurasi yanlebih spesifik untuk lebih jelasnya dpt dil-ihat pada gambar codingan berikut. dan untuk hasilnya dapat dilihat pada gambar berikut.

```

1 # -*- coding: utf-8 -*-
2 """
3 Created on Sat Mar  7 23:10:49 2020
4
5 @author: Liyana
6 """
7
8 for max_depth in range(1, 100):
9     lobener= tree.DecisionTreeClassifier(criterion="entropy",
10                                         max_depth=max_depth)
11     scores = cross_val_score(lobener, jatibarang_att,
12                             jatibarang_pass, cv=5)
13     print("Max depth: %d, Accuracy: %0.2f (+/- %0.2f)" % (
14         max_depth, scores.mean(), scores.std() * 2))

```

```

Max depth: 1, Accuracy: 0.58 (+/- 0.01)
Max depth: 2, Accuracy: 0.58 (+/- 0.06)
Max depth: 3, Accuracy: 0.56 (+/- 0.06)
Max depth: 4, Accuracy: 0.56 (+/- 0.08)
Max depth: 5, Accuracy: 0.58 (+/- 0.05)
Max depth: 6, Accuracy: 0.58 (+/- 0.05)
Max depth: 7, Accuracy: 0.56 (+/- 0.05)
Max depth: 8, Accuracy: 0.56 (+/- 0.07)
Max depth: 9, Accuracy: 0.58 (+/- 0.18)
Max depth: 10, Accuracy: 0.57 (+/- 0.04)
Max depth: 11, Accuracy: 0.58 (+/- 0.06)
Max depth: 12, Accuracy: 0.57 (+/- 0.08)
Max depth: 13, Accuracy: 0.57 (+/- 0.10)
Max depth: 14, Accuracy: 0.55 (+/- 0.07)
Max depth: 15, Accuracy: 0.58 (+/- 0.08)
Max depth: 16, Accuracy: 0.56 (+/- 0.05)
Max depth: 17, Accuracy: 0.57 (+/- 0.09)
Max depth: 18, Accuracy: 0.57 (+/- 0.11)
Max depth: 19, Accuracy: 0.57 (+/- 0.09)
Max depth: 20, Accuracy: 0.57 (+/- 0.07)
Max depth: 21, Accuracy: 0.57 (+/- 0.06)
Max depth: 22, Accuracy: 0.58 (+/- 0.07)
Max depth: 23, Accuracy: 0.58 (+/- 0.10)

```

Gambar 2.84 hasil

11. untuk selanjutnya yaitu menentukan nilai untuk grafik hampir sama dengan nilai akurasi tadi pertama tentukan rank atau batasan nilai itu disini batasannya di mulai dari 1 sampai 20 dengan menggunakan nilai tree tadi maka hasilnya dapat ditentuka kemudian buat variabel i untuk penomoran tiap record yang keluar atau recod hadil dari eksekusi tree tersebut. untuk lebih jelasnya dapat dilihat pada gambar dan untuk hasilnya dapat dilihat pada gambar.

```

1 # -*- coding: utf-8 -*-
2 """
3 Created on Sat Mar  7 23:11:44 2020
4
5 @author: Liyana
6 """
7
8 depth_acc = np.empty((19,3), float)
9 i = 0
10 for max_depth in range(1, 20):
11     lobener = tree.DecisionTreeClassifier(criterion="entropy",
12                                           max_depth=max_depth)
13     scores = cross_val_score(lobener, jatibarang_att,
14                             jatibarang_pass, cv=5)
15     depth_acc[i,0] = max_depth
16     depth_acc[i,1] = scores.mean()
17     depth_acc[i,2] = scores.std() * 2
18     i += 1
19
20 print(depth_acc)

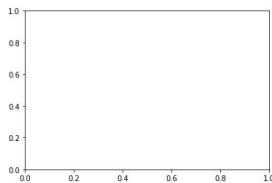
```

```
[[1.0000000e+00 5.79746835e-01 1.01265823e-02]
[2.0000000e+00 5.69620253e-01 2.77327877e-02]
[3.0000000e+00 5.82278481e-01 7.33740595e-02]
[4.0000000e+00 5.77215190e-01 3.03797468e-02]
[5.0000000e+00 5.84810127e-01 9.39106607e-02]
[6.0000000e+00 5.56962025e-01 1.01265823e-01]
[7.0000000e+00 5.92405063e-01 6.28337966e-02]
[8.0000000e+00 6.00000000e-01 6.71721477e-02]
[9.0000000e+00 5.79746835e-01 6.86618226e-02]
[1.0000000e+01 6.10126582e-01 7.74534610e-02]
[1.1000000e+01 6.07536337e-01 5.77304013e-02]
[1.2000000e+01 5.94938709e-01 9.40255035e-02]
[1.3000000e+01 5.82278481e-01 5.16254013e-02]
[1.4000000e+01 5.62278481e-01 5.77304013e-02]
[1.5000000e+01 5.77215190e-01 6.11790796e-02]
[1.6000000e+01 5.82278481e-01 7.67886121e-02]
[1.7000000e+01 5.82278481e-01 7.51007442e-02]
[1.8000000e+01 6.02531546e-01 5.68353021e-02]
[1.9000000e+01 5.92405063e-01 7.05234849e-02]]
```

Gambar 2.85 hasil

12. terakhir yaitu pembuatan grafik untuk pembuatan grafik diambil data dari codingan sebelumnya yang 20 record tadi dengan cara mengimport libray matplotlib.pyplot yang di inisialisasi menjadi bakwankemudian inisialisasi tersebut di eksekusi. untuk lebih jelasnya codingannya seperti gambar dan untuk hasilnya seperti gambar berikut.

```
1 # -*- coding: utf-8 -*-
2 """
3 Created on Sat Mar  7 23:12:39 2020
4
5 @author: Liyana
6 """
7
8 import matplotlib.pyplot as plt
9 fig , ax = plt.subplots()
10 ax.errorbar(depth_acc[:,0] , depth_acc[:,1] , yerr=depth_acc
11 [:,2])
12 plt.show()
```

**Gambar 2.86** hasil

2.5.3 Penanganan Error

- skrinsut error

```
Pada: C:\Users\laptop\PycharmProjects\untitled\src\main\java\com\example\untitled\Main.java", line 16, in method
    fig, ax = plt.subplots()
    ^
    No such variable 'plt' available.
  File "C:\Users\laptop\PycharmProjects\untitled\src\main\java\com\example\untitled\Main.java", line 16, in method
    fig, ax = plt.subplots()
    ^
    NameError: name 'plt' is not defined
```

Gambar 2.87 Error

- kode error dan jenis errornya .

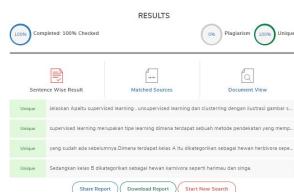
```
jatibarang['pass'] = jatibarangapply(lambda row: 1 if (row['G1']+row['G2']+row['G3'])>= 35 else 0, axis=1)
jatibarang = jatibarang.drop(['G1', 'G2', 'G3'], axis=1)
print(jatibarang.head())
```

NameError: name jatibarangapply is not defined

- Solusi pemecahan masalah

pada codingan tersebut error karena pecelapply tergabung, seharusnya dipisahkan oleh titik

2.5.4 Plagiat



Gambar 2.88 plagiat

2.6 1174043 Irvan Rizkiansyah

2.6.1 Teori

- Jelaskan Apa Itu binari classification dilengkapi ilustrasi gambar sendiri.
- Binary Classification merupakan kegiatan yang berguna untuk mengklasifikasikan elemen-elemen dari sebuah himpunan tertentu ke dalam dua kelompok berbeda (memprediksi kelompok mana yang masing-masing dimiliki) berdasarkan dari aturan klasifikasi.



Gambar 2.89 contoh Binary Classification

- Jelaskan Apaitu supervised learning , unsupervised learning dan clustering dengan ilustrasi gambar sendiri.
- Supervised learning adalah sebuah metode pendekatan yang mana sudah terdapat data yang dilatih, dan terdapat variabel yang ditargetkan

atau yang menjadi tujuan sehingga tujuan dari pendekatan ini adalah mengelompokan suatu data ke data yang sudah ada. contoh pada jeruk termasuk yang mengandung vitamin c maka jeruk telah di kategorikan kedalam vitamin c. sedangkan salmon mengandung vitamin d yang berarti salmon telah di kategorikan yang mengandung vitamin d untuk lebih jelasnya dapat dilihat pada gambar.



Gambar 2.90 contoh supervised learning

Sedangkan unsupervised learning tidak memiliki data latih, sehingga dari data yang ada, kita mengelompokan data tersebut menjadi dua bagian atau bahkan tiga bagian dan seterusnya. contoh Salmon di kategorikan ke dalam vitamin d untuk lebih jelasnya dapat dilihat pada gambar.



Gambar 2.91 contoh unsupervised learning

clustering merupakan peroses mengklasifikasikan yang berdasarkan suatu parameter dalam penentuannya contoh pada berat vitamin c memiliki berat 500 gr dan vitamin d memiliki berat 1000 gr yang berarti berat dibagi dua parameter yaitu lebih kecil samadengan 500 gram dan lebih besar dari 500 gram contoh pada gambar.



Gambar 2.92 contoh clustering

3. Jelaskan apa itu evaluasi dan akurasi dan disertai ilustrasi contoh dengan gambar sendiri.

evaluasi adalah pengumpulan pengumpulan dan pengamatan dari berbagai macam bukti untuk mengukur dampak efektifitas dari suatu objek, program, atau proses berkaitan dengan spesifikasi atau persyaratan yang telah di tetapkan sebelumnya. sedangkan akurasi itu sendiri merupakan bagian dari evaluasi yang merupakan ketepatan data terhadap suatu objek berdasarkan keriteria tertentu. kita dapat mengevaluasi seberapa baik model bekerja dengan mengukur akurasiinya. ketepatan akan di

definisikan sebagai persentase kasus yang diklasifikasikan dengan benar. hal ini berkaitan dengan confusion matrix pada materi selanjutnya. lebih jelasnya pada gambar berikut:



Gambar 2.93 contoh Evaluasi

- Jelaskan bagaimana cara membuat Confusion Matrix, Buat confusion matrix sendiri.

Dalam pembuatan confusion matrix tentukan parameter atau objek yang akan di evaluasi contoh udang, salmon, tuna buat tabel dengan baris dan kolom berjumlah tiga kemudian tentukan nilai miring pada setiap kolom tersebut disini saya memberi nilai 20 dengan ketentuan setiap baris harus berisi nilai 20 nilai tersebut jika terbagi ke kolom lain maka jumlahnya harus bernilai 20 jika tidak berarti data tersebut tidak akurat. untuk lebih jelanya dapat dilihat pada gambar berikut :

| Confusion Matrix | | | | |
|------------------|-------|--------|------|----|
| Actual | Udang | Salmon | Tuna | |
| Prediction | Udang | 120 | 20 | 10 |
| Udang | 120 | 20 | 10 | |
| Salmon | 20 | 10 | 10 | |
| Tuna | 10 | 10 | 10 | |
| Total | 150 | 30 | 30 | |

Gambar 2.94 contoh Confusion Matrix

- Jelaskan bagaimana K-fold cross validation bekerja dengan gambar ilustrasi contoh buatan sendiri.

K-fold Cross Validation merupakan cara untuk melatih suatu mesin dimana di dalamnya terdapat data set yang dibagi menjadi dua yaitu untuk data testing dan data training contoh 1000 data merupakan data set dan 400 data digunakan untuk data testing kemudian 600 datanya digunakan untuk data training dimana data training tersebut digunakan untuk menentukan nilai bobot yang dimasukan kedalam rumus regresi linier. sedangkan nilai testing akan dijadikan nilai inputan untuk rumus regresi linier. contohnya dapat dilihat pada gambar berikut :



Gambar 2.95 contoh K-fold cross validation

- Jelaskan Apa itu decision tree dengan gambar ilustrasi contoh buatan sendiri.

Decision tree merupakan implementasi dari binari clasification dimana pada pohon keputusan akan terdapat root atau akar dan cabang cabangnya yang nilainya seperti if contoh pada root berisi nilai hidup di air, apakah ikan pada cabang satu bernilai iya dan pada cabang dua bernilai tidak jika nilainya iya berarti hidup di air dan jika tidak maka bukan hidup diair.



Gambar 2.96 contoh decision tree

7. jelaskan apa itu information gain dan entropi dengan gambar ilustrasi buatan sendiri.

informasian gain merupakan informasi atau keriteria dalam pembagian sebuah objek contoh information gain pada ikan yaitu hidup di air, berkoloni, berinsang. untuk lebih jelasnya dapat dilihat pada gambar berikut :



Gambar 2.97 contoh information gain

2.6.2 Scikit-Learn

- 1.

```

1 # load dataset (student mat pakenya)
2 import pandas as pd
3 jeruk = pd.read_csv('D:\student-mat.csv', sep=';')
4 print(len(jeruk))

```

```

In [19]: import pandas as pd
...: jeruk = pd.read_csv('D:\student-mat.csv', sep=';')
...: print(len(jeruk))
395

```

Gambar 2.98 Hasil Percobaan 1

- 2.

```

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

```

```
In [20]: jeruk['pass'] = jeruk.apply(lambda row: 1 if (row['G1']>row['G2'])&(row['G1']>row['G3']) else 0, axis=1)
....;
print(jeruk.head())
.....;
print(jeruk.head())
.....;
# Mengonversi address,famsiz... data ke health absences pass
0 GP F 18 0 GT3 ... 1 1 3 6 0
1 GP F 19 0 GT3 ... 1 1 3 6 0
2 GP F 15 0 LT4 ... 2 3 3 10 0
3 GP F 15 0 GT3 ... 1 1 5 10 0
4 GP F 16 0 GT3 ... 1 2 5 2 0
[5 rows x 31 columns]
```

Gambar 2.99 Hasil Percobaan 2

3.

```
1 jeruk = pd.get_dummies(jeruk, columns=['sex', 'school', '',
   'address', 'famsize', 'Pstatus', 'Mjob', 'Fjob', 'reason', '',
   'guardian', 'schoolsup', 'famsup', 'paid', 'activities', '',
   'nursery', 'higher', 'internet', 'romantic']) # Mengonversi kategori variabel menjadi variabel indikator
2 jeruk.head()#mengambil baris pertama dari cakue
```

```
In [21]: jeruk = pd.get_dummies(jeruk, columns=['sex', 'school', 'address', 'famsize',
   'Pstatus', 'Mjob', 'Fjob', 'reason', 'parent1', 'fathers', 'mothers', 'paid',
   'activities', 'nursery', 'higher', 'internet', 'romantic']) # Mengonversi kategori
.....;
jeruk.head()#mengambil baris pertama dari cakue
Out[21]:
   age Medu Fedu ... internet yes romantic_no romantic_yes
0 15 3 3 1 1 1 1 1 0
1 17 3 3 1 1 1 1 1 0
2 15 3 3 1 1 1 1 1 0
3 15 4 2 1 1 1 1 0 1
4 16 3 3 1 1 1 0 1 0
[5 rows x 57 columns]
```

Gambar 2.100 Hasil Percobaan 3

4.

```
1 jeruk = jeruk.sample(frac=1)
2 # split training and testing data
3 jeruk_train = jeruk[:500]
4 jeruk_test = jeruk[500:]
5 jeruk_train.att = jeruk_train.drop(['pass'], axis=1)
6 jeruk_train.pass = jeruk_train['pass']
7 jeruk_test.att = jeruk_test.drop(['pass'], axis=1)
8 jeruk_test.pass = jeruk_test['pass']
9 jeruk.att = jeruk.drop(['pass'], axis=1)
10 jeruk_pass = jeruk['pass']
11 # number of passing students in whole dataset:
12 import numpy as np
13 print("Passing: %d out of %d (%.2f%%)" % (np.sum(jeruk_pass),
   len(jeruk_pass), 100*float(np.sum(jeruk_pass)) / len(
   jeruk_pass)))
```

```
In [22]: jeruk = jeruk.sample(frac=1)
....;
jeruk_train = jeruk[:500]
....;
jeruk_train.att = jeruk_train.drop(['pass'], axis=1)
....;
jeruk_train.pass = jeruk_train['pass']
....;
jeruk_test.att = jeruk_test.drop(['pass'], axis=1)
....;
jeruk_test.pass = jeruk_test['pass']
....;
jeruk.att = jeruk.drop(['pass'], axis=1)
....;
jeruk_pass = jeruk['pass']
....;
# number of passing students in whole dataset:
....;
print("Passing: %d out of %d (%.2f%%)" % (np.sum(jeruk_pass),
   len(jeruk_pass), 100*float(np.sum(jeruk_pass)) / len(
   jeruk_pass)))
Passing: 160 out of 395 (42.05%)
```

Gambar 2.101 Hasil Percobaan 4

5.

```

1 from sklearn import tree
2 anggur = tree.DecisionTreeClassifier(criterion="entropy",
3                                     max_depth=5)
4 anggur = anggur.fit(jeruk_train_att, jeruk_train_pass)
```

```
[In 28]: from sklearn import tree
...: anggur = tree.DecisionTreeClassifier(criterion="entropy", max_depth=5)
...: anggur = anggur.fit(jeruk_train_att, jeruk_train_pass)
```

Gambar 2.102 Hasil Percobaan 5

6.

```

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

7.

```

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

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

Gambar 2.103 Hasil Percobaan 7

8.

```
1 anggur.score(jeruk_test_att, jeruk_test_pass)
```

9.

```

1 from sklearn.model_selection import cross_val_score
2 scores = cross_val_score(anggur, jeruk_att, jeruk_pass, cv
3                         =5)
4 # show average score and +/- two standard deviations away
4 #(covering 95% of scores)
5 print("Accuracy: %0.2f (+/- %0.2f)" % (scores.mean(), scores.
6                                         std() * 2))
```

```
In [52]: from sklearn.model_selection import cross_val_score
... scores = cross_val_score(anggur, jeruk_att, jeruk_pass, cv=5)
... print("Mean score: %0.2f (+/- %0.2f)" % (scores.mean(), scores.std() * 2))
... print("Accuracy: %0.2f (+/- %0.2f)" % (scores.mean(), scores.std() * 2))

Accuracy: 0.58 (+/- 0.07)
```

Gambar 2.104 Hasil Percobaan 9

10.

```
1 for max_depth in range(1, 100):
2     anggur = tree.DecisionTreeClassifier(criterion="entropy",
3                                         max_depth=max_depth)
4     scores = cross_val_score(anggur, jeruk_att, jeruk_pass,
5                               cv=5)
5     print("Max depth: %d, Accuracy: %0.2f (+/- %0.2f)" % (
#%%%
```

```
In [53]: for max_depth in range(1, 100):
...     anggur = tree.DecisionTreeClassifier(criterion="entropy", max_depth=max_depth)
...     scores = cross_val_score(anggur, jeruk_att, jeruk_pass, cv=5)
...     print("Max depth: %d, Accuracy: %0.2f (+/- %0.2f)" % (max_depth, scores.mean(),
...     scores.std()))
... Max depth: 1, Accuracy: 0.58 (+/- 0.01)
... Max depth: 2, Accuracy: 0.58 (+/- 0.09)
... Max depth: 3, Accuracy: 0.58 (+/- 0.07)
... Max depth: 4, Accuracy: 0.57 (+/- 0.04)
... Max depth: 5, Accuracy: 0.58 (+/- 0.03)
... Max depth: 6, Accuracy: 0.58 (+/- 0.03)
... Max depth: 7, Accuracy: 0.58 (+/- 0.03)
... Max depth: 8, Accuracy: 0.58 (+/- 0.02)
... Max depth: 9, Accuracy: 0.58 (+/- 0.02)
... Max depth: 10, Accuracy: 0.58 (+/- 0.02)
... Max depth: 11, Accuracy: 0.58 (+/- 0.02)
... Max depth: 12, Accuracy: 0.60 (+/- 0.03)
... Max depth: 13, Accuracy: 0.60 (+/- 0.03)
... Max depth: 14, Accuracy: 0.61 (+/- 0.03)
... Max depth: 15, Accuracy: 0.61 (+/- 0.03)
... Max depth: 16, Accuracy: 0.60 (+/- 0.03)
... Max depth: 17, Accuracy: 0.60 (+/- 0.03)
... Max depth: 18, Accuracy: 0.60 (+/- 0.03)
... Max depth: 19, Accuracy: 0.60 (+/- 0.03)
... Max depth: 20, Accuracy: 0.60 (+/- 0.03)
... Max depth: 21, Accuracy: 0.60 (+/- 0.03)
... Max depth: 22, Accuracy: 0.60 (+/- 0.03)
... Max depth: 23, Accuracy: 0.60 (+/- 0.03)
... Max depth: 24, Accuracy: 0.62 (+/- 0.03)
... Max depth: 25, Accuracy: 0.62 (+/- 0.03)
... Max depth: 26, Accuracy: 0.61 (+/- 0.03)
... Max depth: 27, Accuracy: 0.61 (+/- 0.03)
... Max depth: 28, Accuracy: 0.61 (+/- 0.03)
```

Gambar 2.105 Hasil Percobaan 10

11.

```
1 depth_acc = np.empty((19,3), float)
2 i = 0
3 for max_depth in range(1, 20):
4     anggur = tree.DecisionTreeClassifier(criterion="entropy",
5                                         max_depth=max_depth)
6     scores = cross_val_score(anggur, jeruk_att, jeruk_pass,
7                               cv=5)
8     depth_acc[i,0] = max_depth
9     depth_acc[i,1] = scores.mean()
10    depth_acc[i,2] = scores.std() * 2
11    i += 1
12
13 print(depth_acc)
```

Gambar 2.106 Hasil Percobaan 11

12.

```
1 import matplotlib.pyplot as plt
2 fig, ax = plt.subplots()
3 ax.errorbar(depth_acc[:,0], depth_acc[:,1], yerr=depth_acc
4 [:,2])
5 plt.show()
```

```
In [55]: import matplotlib.pyplot as plt
fig, ax = plt.subplots()
ax.errorbar(depth_acc[:,0], depth_acc[:,1], yerr=depth_acc[:,2])
plt.show()
```

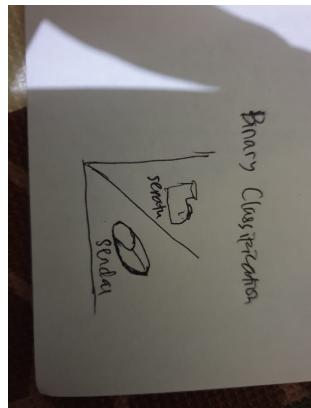
Gambar 2.107 Hasil Percobaan 12

2.7 1174040 - Hagan Rowlenstino A. S.

2.7.1 Teori

1. Jelaskan Apa Itu binari classification dilengkapi ilustrasi gambar sendiri.

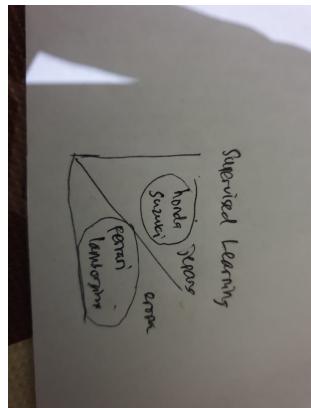
Binary Classification adalah sebuah aksi dimana dilakukannya klasifikasi dari sebuah kumpulan objek tertentu ke dalam dua buah kelompok yang berbeda berdasarkan beberapa fitur atau sifat-sifat.



Gambar 2.108 Binary Classification

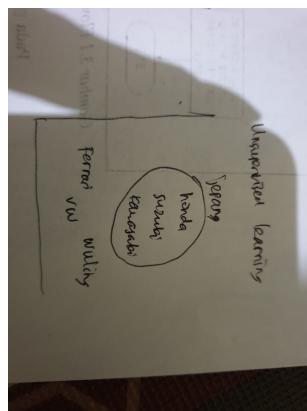
2. Jelaskan Apaitu supervised learning , unsupervised learning dan clustering dengan ilustrasi gambar sendiri.

Supervised learning adalah sebuah cara untuk mengklasifikasikan kumpulan objek yang fitur ataupun sifat-sifatnya dari kelas nya sudah di tentukan sebelumnya. contoh pada merk mobil honda yang merupakan buatan jepang dan ferarri merupakan buatan eropa



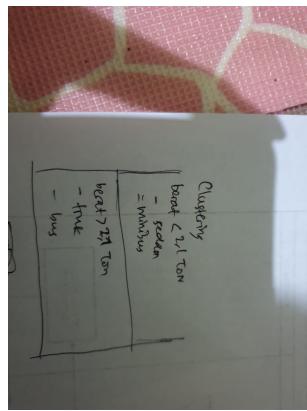
Gambar 2.109 Supervised

Unsupervised learning merupakan teknik pengklasifikasian tanpa perlu di supervised sebelumnya, dimana model tersebut yang akan bekerja sendiri untuk menemukan infomasi yang terkait. seperti pada contoh di gambar.



Gambar 2.110 Unsupervised

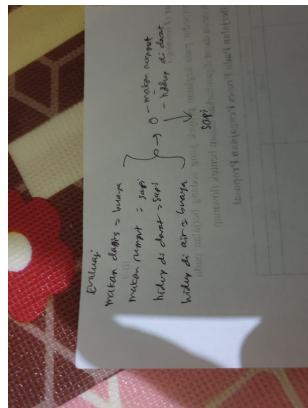
clustering merupakan metode pengelompokan data dengan membagi ke dalam beberapa kelompok. disini saya memberi contoh dengan kriteria kendaraan yang lebih dari 2,1 ton dan kurang dari 2,1 ton.



Gambar 2.111 clustering

3. Jelaskan apa itu evaluasi dan akurasi dan disertai ilustrasi contoh dengan gambar sendiri.

Evaluasi adalah proses yang sistematis yang ditujukan untuk menentukan ataupun membuat keputusan dengan memeriksa pernyataan-pernyataan yang telah ada sebelumnya. Akurasi adalah tingkat ketepatan dari sebuah data yang telah dihasilkan dari evaluasi yang telah dilakukan sebelumnya.



Gambar 2.112 Evaluasi

- Jelaskan bagaimana cara membuat Confusion Matrix, Buat confusion matrix sendiri.

Untuk membuat confusion matrix kita perlu menentukan objek yang akan dievaluasi terlebih dahulu, setelah itu tentukan nilai miring pada setiap kolom objek tersebut lalu setiap baris dan jika lainnya di bagi baris dan kolom nya, harus bernilai sesuai nilai yang telah di tetapkan sebelumnya.

| | | benar | salah | Total |
|-------|-------|-------|-------|-------|
| benar | benar | 15 | 5 | 20 |
| | salah | 10 | 13 | 23 |
| | | benar | salah | Total |
| | | 15 | 10 | 25 |
| | | 5 | 13 | 18 |
| | | 20 | 23 | 43 |

Gambar 2.113 Confussion Matrix

- Jelaskan bagaimana K-fold cross validation bekerja dengan gambar ilustrasi contoh buatan sendiri.

Melatih mesin dengan membagi data set yang akan di olah menjadi dua buah yaitu data testing dan training. sebagai contoh ada 500 data pada dataset, maka 200 sebagai data testing dan 300 akan menjadi data training.

**Gambar 2.114** K-Fold

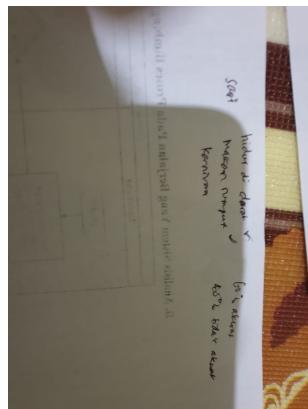
6. Jelaskan Apa itu decision tree dengan gambar ilustrasi contoh buatan sendiri.

Desicion tree adalah sebuah model prediksi yang menggunakan struktur yang menyerupai pohon ataupun sebuah hirarki

**Gambar 2.115** Decision Tree

7. jelaskan apa itu information gain dan entropi dengan gambar ilustrasi buatan sendiri.

information Gain mengukur berapa banyak informasi yang sebuah fitur berikan kepada kita tentang kelas tersebut. sedangkan entropi menentukan bagaimana decision tree memilah data nya.



Gambar 2.116 Information Gain

2.7.2 scikit-learn

1. baris pertama yaitu mengimport library pandas. pada baris kedua ada fungsi read_csv untuk membaca file csv serta datanya dipisah dengan tanda titik koma lalu dimasukkan kedalam variable bernama pisang. fungsi pada baris terakhir yaitu untuk mengetahui panjang data / banyak data yang ada

```
1 import pandas as pd
2 pisang = pd.read_csv('student-mat.csv', sep=';')
3 print(len(pisang))
```

```
In [42]: import pandas as pd
....: pisang = pd.read_csv('student-mat.csv', sep=';')
....: print(len(pisang))
395
```

Gambar 2.117 No 1

2. Disini digunakan fungsi untuk menunjukkan lulus atau gagal dimana jika lulus ditandai dengan angka 1 dan gagal dengan angka 0. data yang diperlukan yaitu data dari csv yang sebelumnya karena itu masih digunakan variable pisang. yang selanjutnya akan dieksekusi.

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

| # | school | sex | age | address | famsize | Pstatus | Mjob | Fjob | reason | guardian | schoolsup | famsup | paid | activities | nursery | higher | internet | romantic |
|---|--------|-----|-----|---------|---------|---------|-------|-------|--------|----------|-----------|--------|------|------------|---------|--------|----------|----------|
| 0 | GP | F | 15 | U | GT3 | T | Atkin | Atkin | LE3 | Atkin | N | N | Y | Y | N | Y | Y | |
| 1 | GP | F | 17 | U | GT3 | T | Atkin | Atkin | LE3 | Atkin | N | N | Y | Y | N | Y | Y | |
| 2 | GP | F | 15 | U | LE3 | T | Atkin | Atkin | LE3 | Atkin | N | N | Y | Y | N | Y | Y | |
| 3 | GP | F | 15 | U | GT3 | T | Atkin | Atkin | LE3 | Atkin | N | N | Y | Y | N | Y | Y | |
| 4 | GP | F | 16 | U | GT3 | T | Atkin | Atkin | LE3 | Atkin | N | N | Y | Y | N | Y | Y | |

[5 rows x 31 columns]

Gambar 2.118 No 2

3. Disini yaitu mengkonversi kategori variable menjadi variable indikator

```

1 pisang = pd.get_dummies(pisang, columns=['sex', 'school', 'address', 'famsize', 'Pstatus', 'Mjob', 'Fjob', 'reason', 'guardian', 'schoolsup', 'famsup', 'paid', 'activities', 'nursery', 'higher', 'internet', 'romantic']) # Mongkonversi kategori variabel menjadi variabel indikator
2 pisang.head()

```

| # | age | Medu | Fedu | ... | internet_yes | romantic_no | romantic_yes |
|---|-----|------|------|-----|--------------|-------------|--------------|
| 0 | 18 | 4 | 4 | ... | 0 | 1 | 0 |
| 1 | 17 | 3 | 3 | ... | 1 | 1 | 0 |
| 2 | 15 | 1 | 1 | ... | 1 | 1 | 0 |
| 3 | 15 | 2 | 2 | ... | 1 | 0 | 1 |
| 4 | 16 | 3 | 2 | ... | 0 | 1 | 0 |

[5 rows x 57 columns]

Gambar 2.119 No 3

4. bagian ini berfungsi untuk menentukan data training dan data testing dari dataset csv yang telah dimasukkan di dalam variable pisang tadi. lalu mengimport library numpy untuk operasi vektor serta matrix karena data diatas berupa matrix.

```

1 pisang = pisang.sample(frac=1)
2 # split training and testing data
3 pisang_train = pisang[:500]
4 pisang_test = pisang[500:]
5 pisang_train_att = pisang_train.drop(['pass'], axis=1)
6 pisang_train_pass = pisang_train['pass']
7 pisang_test_att = pisang_test.drop(['pass'], axis=1)
8 pisang_test_pass = pisang_test['pass']
9 pisang_att = pisang.drop(['pass'], axis=1)
10 pisang_pass = pisang['pass']
11 # number of passing students in whole dataset:
12 import numpy as np
13 print("Passing: %d out of %d (%.2f%%)" % (np.sum(pisang_pass),
14 , len(pisang_pass), 100*float(np.sum(pisang_pass)) / len(pisang_pass)))

```

```

In [46]: pisang = pisang.sample(frac=1)
...: # split training and testing data
...: pisang_train = pisang[:500]
...: pisang_test = pisang[500:]
...: pisang_train_att = pisang_train.drop(['pass'], axis=1)
...: pisang_train_pass = pisang_train['pass']
...: pisang_test_att = pisang_test.drop(['pass'], axis=1)
...: pisang_test_pass = pisang_test['pass']
...: pisang_att = pisang.drop(['pass'], axis=1)
...: pisang_pass = pisang['pass']
...: # number of passing students in whole dataset:
...: import numpy as np
...: print("Passing: %d out of %d (%.2f%%)" % (np.sum(pisang_pass),
...: len(pisang_pass), 100*float(np.sum(pisang_pass)) / len(pisang_pass)))
Passing: 166 out of 395 (42.05%)

```

Gambar 2.120 No 4

5. pada baris pertam kita mengimport tree dari library sklearn yang berfungsi untuk membuat desicion tree

```
1 from sklearn import tree
2 apel = tree.DecisionTreeClassifier(criterion="entropy",
3                                   max_depth=5)
4 apel = apel.fit(pisang_train_att, pisang_train_pass)
5 print(apel)
```

Gambar 2.121 No 5

- ## 6. mengimport library graphviz

```
1 import graphviz
2 dot_data = tree.export_graphviz(apel, out_file=None, label="all",
3                                 impurity=False, proportion=True, feature_names=list(pisang_train.att), class_names=["fail", "pass"], filled=True, rounded=True)
4 graph = graphviz.Source(dot_data)
5 %%
```

7. untuk menyimpan data dari tree dan menarik data langsung dari desicion tree

1 #%

```
In [54]: tree.export_graphviz(apel, out_file="student-performance.dot", label="all",
    impurity=False, proportion=True,
    ...
    feature_names=list(pisang_train_att),
    class_names=["fail", "pass"], filled=True, rounded=True)
```

Gambar 2.122 No 7

8. untuk mencari ketepatan data yang telah diolah

9. untuk menghitung akurasi ketepatan data

```
1 # show average score and +/- two standard deviations away
2 #(covering 95% of scores)
3 print("Accuracy: %0.2f (+/- %0.2f)" % (scores.mean(), scores.
4 std() * 2))
5 #%%
```

```
In [54]: from sklearn.model_selection import cross_val_score
...: scores = cross_val_score(apel, pisang_att, pisang_pass, cv=5)
...: print("Meaningful 95% of scores")
...: print((scores.mean() - 30.21) / (-30.21) * (scores.mean(), scores.std() * 2))
Accuracy: 0.58 (+/- 0.06)
```

Gambar 2.123 No 9

10. membuat variable baru dengan nilai tree di dalamnya dengan bentuk akurasi yang spesifik

```
1 scores = cross_val_score(apel, pisang_att, pisang_pass,
2                         cv=5)
3 print("Max depth: %d, Accuracy: %0.2f (+/- %0.2f)" % (
4     max_depth, scores.mean(), scores.std() * 2))
#%%

```

```
Max depth: 88, Accuracy: 0.61 (+/- 0.05)
Max depth: 89, Accuracy: 0.61 (+/- 0.06)
Max depth: 90, Accuracy: 0.60 (+/- 0.07)
Max depth: 91, Accuracy: 0.62 (+/- 0.07)
Max depth: 92, Accuracy: 0.61 (+/- 0.09)
Max depth: 93, Accuracy: 0.61 (+/- 0.05)
Max depth: 94, Accuracy: 0.60 (+/- 0.08)
Max depth: 95, Accuracy: 0.60 (+/- 0.05)
Max depth: 96, Accuracy: 0.61 (+/- 0.08)
Max depth: 97, Accuracy: 0.62 (+/- 0.06)
Max depth: 98, Accuracy: 0.61 (+/- 0.06)
Max depth: 99, Accuracy: 0.61 (+/- 0.06)
```

Gambar 2.124 No 10

11. menentukan rank batasannya yaitu 1 sampai 20 dari nilai tree tadi yang digunakan untuk menentukan nilai untuk grafik

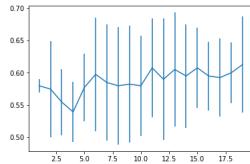
```
1 for max_depth in range(1, 20):
2     apel = tree.DecisionTreeClassifier(criterion="entropy",
3                                         max_depth=max_depth)
4     scores = cross_val_score(apel, pisang_att, pisang_pass,
5                               cv=5)
6     depth_acc[i, 0] = max_depth
7     depth_acc[i, 1] = scores.mean()
8     depth_acc[i, 2] = scores.std() * 2
9     i += 1
10 print(depth_acc)
#%
```

```
[{1.00000000e+00 5.79746835e-01 1.01265823e-02]
[2.00000000e+00 5.74683544e-01 7.44148783e-02]
[3.00000000e+00 5.54430580e-01 5.16356466e-02]
[4.00000000e+00 5.39240506e-01 4.69559304e-02]
[5.00000000e+00 5.24050532e-01 4.22762942e-02]
[6.00000000e+00 5.07468354e-01 3.82814976e-02]
[7.00000000e+00 5.04810127e-01 3.97221747e-02]
[8.00000000e+00 5.07946835e-01 9.11392405e-02]
[9.00000000e+00 5.82278481e-01 9.05749954e-02]
[1.00000000e+01 5.79746835e-01 7.74534610e-02]
[1.10000000e+01 5.80574683e-01 7.64825621e-02]
[1.20000000e+01 5.80574683e-01 7.61825621e-02]
[1.30000000e+01 6.05063201e-01 8.82814976e-02]
[1.40000000e+01 5.04936709e-01 8.00576623e-02]
[1.50000000e+01 6.0750494937e-01 6.20123986e-02]
[1.60000000e+01 5.04936709e-01 5.31042455e-02]
[1.70000000e+01 5.92405636e-01 5.07594937e-02]
[1.80000000e+01 6.00000000e-01 4.69559304e-02]
[1.90000000e+01 6.12658228e-01 7.44148783e-02]]
```

Gambar 2.125 No 11

12. mengimport matplotlib.pyplot yang akan digunakan untuk membuat grafik

```
1 ax.errorbar(depth_acc[:,0], depth_acc[:,1], yerr=depth_acc
              [:,2])
2 plt.show()
3 #%%
```

**Gambar 2.126** No 12

2.7.3 Penanganan Error

1. Screenshot

```
KeyError: "[('sex', 'school', 'address', 'famsize', 'Pstatus', 'Mjob', 'Fjob', 'reason', 'guardian', 'schoolsup', 'famsup', 'paid', 'activities', 'nursery', 'higher', 'internet', 'romantic') not in index"]
```

Gambar 2.127 Screenshot Error

2. Kode dan Jenis Error Jenis error adalah Key Error

```
1 pisang = pd.get_dummies(pisang, columns=['sex', 'school', 'address', 'famsize', 'Pstatus', 'Mjob', 'Fjob', 'reason', 'guardian', 'schoolsup', 'famsup', 'paid', 'activities', 'nursery', 'higher', 'internet', 'romantic']) #
Mongkonversi kategori variabel menjadi variabel indikator
2 kuda.head()
```

3. Solusi Penanganan Error Mengganti kuda dengan pisang

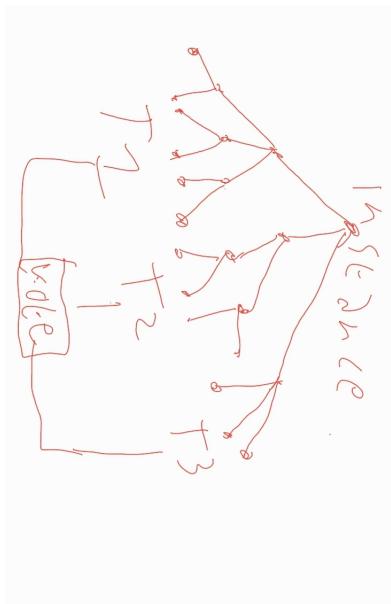
BAB 3

CHAPTER 3

3.0.1 Soal Teori

1. Jelaskan apa itu random forest, sertakan gambar ilustrasi buatan sendiri.

Random Forest atau hutan acak yaitu kumpulan dari pohon keputusan yang difungsikan untuk membaca objek tertentu sesuai dengan yang telah di sepakati untuk di baca pada sistem yang menggunakan pengkondisian seperti Artificial Intelligence. Pohon-pohon keputusan tersebut akan memunculkan hasil yang akan disimpulkan oleh random forest. Pembagian jumlah data yang dimasukan kedalam decision tree pada random forest akan di bagi sama rata sesuai dengan ketentuan tertentu yang disepakati saat sebelum membuat sebuah sistem tersebut.



Gambar 3.1 Contoh Confusion Matrix

2. Jelaskan cara membaca dataset khusus dan artikan makna setiap file dan isi field masing masing file. Yang harus dilakukan untuk membaca dataset adalah mendownload dataset yang sudah disediakan, lalu dibuka menggunakan IDE khusus dari python seperti Spyder untuk mengetahui isi dari dataset yang sudah didownload. Tergantung dari kebutuhan, file dapat berbentuk txt ataupun csv yang sudah sering digunakan karena bentuk datanya seperti tabel dan mudah untuk digunakan. Di dalam file akan mengandung class dari field atau kumpulan data hasil penilaian yang sudah dilakukan. Total datanya sendiri bisa sampai ribuan walaupun hanyalah kategori dan status seperti 0 dan 1.
3. Jelaskan apa itu Cross Validation. Cross Validation merupakan metode untuk mengevaluasi hasil dari sebuah penilaian yang telah digunakan dengan cara membagi dua bagian dari dataset menjadi data training dan data testing. Lalu data akan diolah sehingga muncul tingkat akurasi dari metode yang digunakan contoh pada metode random forest dataset nya dibagi menjadi dua menjadi data training dan data testing kemudian data tersebut diolah oleh mesin untuk melihat tingkat akurasinya maka akan muncul misalkan akurasi kebenaran sebesar 44 % begitu pula dengan menggunakan metode-metode yang lain seperti decision tree dan SVM.
4. Jelaskan apa arti score 44 % pada random forest, 27 % pada decision tree dan 29 % dari SVM. Maksud dari score 44 % tersebut yaitu nilai ketepatan atau kebenaran dari sebuah parameter, tingkat ketepatan

tersebut berpengaruh pada bagaimana mesin tersebut bisa menyatakan jenis burung tersebut dengan akurasi kebenaran 44 %. sedangkan pada metode decision tree yaitu 27 % yang berarti menunjukan bahwa tingkat akurasi ketepatan mesin jika mengerjakan sesuatu atau menyatakan keputusan dengan metode decision tree maka nilai kebenarannya bernilai 27 %. sedangkan dengan menggunakan metode SVM menunjukan hasil 29 % yang berarti nilai ketepatan atau kebenaran dalam memecahkan masalah menggunakan metode SVM ini sebesar 29 % . maka dari itu dapat di simpulkan bahwa dengan menggunakan metode random forest mesin dapat memecahkan masalah lebih akurat dibandingkan dengan menggunakan decision tree dan SVM.

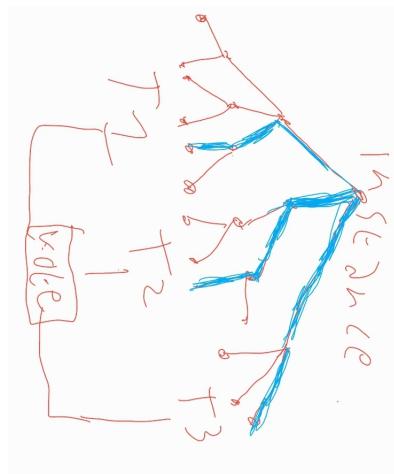
5. Jelaskan bagaimana cara membaca confusion matriks dan contohnya memakai gambar atau ilustrasi sendiri. Cara untuk membaca confusion matrix adalah dengan cara memasukan parameter nilai yang ada pada datasets. Contoh pada dataset terdapat class yang disandingkan dengan nama burung untuk di normalisasi maka akan menunjukan nilai matrix yang mendekati nilai benar dalam bentuk angka misalkan 0,5 0,2 dan seterusnya mendekati nilai satu. di karenakan susahnya membaca nilai angka maka sering di ubah menjadi bentuk grafik. Untuk nilai - nilai yang ada, adalah sebagai berikut :
 - True Positive : Data positif yang terdeteksi memiliki hasil benar
 - False Positive : Data Positif yang terdeteksi memiliki hasil salah
 - True Negative : Data negatif yang terdeteksi memiliki hasil benar
 - False Negative : Data negatif yang terdeteksi memiliki hasil salah

| | | C ₁ | C ₂ |
|--------|----------------|----------------|----------------|
| | | Predict | Predict |
| Actual | C ₁ | TP | FN |
| | C ₂ | FP | TN |

Gambar 3.2 Contoh Confusion Matrix

6. Jelaskan apa itu voting pada random forest disertai dengan ilustrasi gambar sendiri.

Voting merupakan data hasil dari decision tree yang terdapat pada random forest. Dimana hasil data tersebut di gunakan sebagai acuan untuk hasil dari random forest. sebagai contoh misalkan pada satu random forest terdapat enam decision tree untuk menentukan jenis pekerjaan orang, pada decision tree ke satu menyimpulkan bahwa pekerjaanya yaitu dosen , pada decision tree ke dua yaitu dosen kemudian pada decision tiga dosen , pada decision tree ke empat yaitu pekerja kantoran, pada decision tree ke lima yaitu pekerja kantoran dan pada decision tree ke enam yaitu dosen. maka pada random forest dapat menyimpulkan hasilnya yaitu dosen.



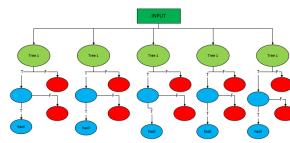
Gambar 3.3 Contoh Random Forest yang sudah Divote

3.1 Faisal Najib Abdullah / 1174042

3.1.1 Teori

1. Jelaskan apa itu random forest, sertakan gambar ilustrasi buatan sendiri.

Random Forest atau hutan acak yaitu kumpulan dari pohon-pohon keputusan yang digunakan untuk membaca objek tertentu yang telah di sepakati untuk di baca dalam AI. pohon-pohon keputusan tersebut akan memunculkan hasil-hasil yang akan disimpulkan oleh random forest. pembagian jumlah data yang dimasukan kedalam decision tree pada random forest akan di bagi sama rata sesuai codingan atau ketentuan tertentu yang di sepakati. misalkan data yang akan digunakan sebanyak 314 jika dalam satu decision tree di putuskan untuk memiliki 50 data maka pada satu random forest akan terdapat enam atau tujuh decision tree.



Gambar 3.4 contoh binari calssification

2. Jelaskan cara membaca dataset khusus dan artikan makna setiap file dan isi field masing masing file. langkah pertama download terlebih dahulu dataset nya kemudian buka menggunakan spyder bawaan anaconda untuk mengetahui isi dari dataset tersebut. biasanya data tersebut berisi

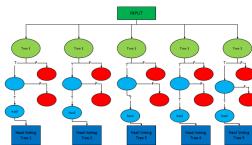
databerekstensi .txt yang di dalamnya terdapat class dari field atau data data yang ada data tersebut. contoh pada data burung ada field index dan angka, index biasanya berisi angka, angka angka tersebut memiliki makna yaitu pengganti nama atau jenis dari burung tersebut sedangkan pada field yang berisi nilai 0 dan 1 berarti menyatakan atau maknanya yaitu memberikan nilai ya dan tidak nilai tersebut di ubah menjadi angka nol dan satu karna data pada field tersebut harus berisi nilai boolean atau pilihan ya dan tidak di karenakan komputer susah membaca nilai dan tidak maka di ubahlah menjadi 0 dan 1 dengan 0 bernilai tidak dan 1 bernilai ya.

3. Jelaskan apa itu Cross Validation. Cross Validation merupakan cara untuk mengevaluasi hasil dari sebuah metode yang telah digunakan dengan cara membagi dua bagian dari dataset menjadi data training dan data testing kemudian data tersebut diolah hingga muncul tingkat akurasi dari metode yang digunakan contoh pada metode random forest dataset nya di bagi menjadi dua menjadi data training dan data testing kemudian data tersebut di olah oleh mesin untuk melihat tingkat akurasinya maka akan muncul misalkan akurasi kebenaran sebesar 44 % begitu pula dengan menggunakan metode-metode yang lain seperti decision tree dan SVM.
4. Jelaskan apa arti score 44 % pada random forest, 27 % pada decision tree dan 29 % dari SVM. maksud dari score 44 % tersebut yaitu nilai ketepatan atau kebenaran atau bisa disebut hasil dari random forest misalkan dengan metode random forest mesin membaca objek burung, mesin tersebut bisa menyatakan jenis burung tersebut dengan akurasi kebenaran 44 %. sedangkan pada metode decision tree yaitu 27 % yang berarti menunjukan bahwa tingkat akurasi ketepatan mesin jika mengerjakan sesuatu atau menyatakan keputusan dengan metode decision tree maka nilai kebenarannya bernilai 27 %. sedangkan dengan menggunakan metode SVM menunjukan hasil 29 % yang berarti nilai ketepatan atau kebenaran dalam memecahkan masalah menggunakan metode SVM ini sebesar 29 % . maka dari itu dapat di simpulkan bahwa dengan menggunakan metode random forest mesin dapat memecahkan masalah lebih akurat dibandingkan dengan menggunakan decision tree dan SVM.
5. Jelaskan bagaimana cara membaca confusion matriks dan contohnya memakai gambar atau ilustrasi sendiri. cara membaca confusio matrix dengan cara memasukan para meter nilai yang ada pada datasets contoh pada dataset terdapat class yang disandingkan dengan nama burung untuk di normalisasi maka akan menunjukan nilai matrix yang mendekati nilai benar dalam bentuk angka misalkan 0,5 0,2 dan seterusnya mendekati nilai satu. di karenakan susahnya membaca nilai angka maka sering di ubah menjadi bentuk grafik.
6. Jelaskan apa itu voting pada random forest disertai dengan ilustrasi gambar sendiri.

| | Bueng 1 | Bueng 2 | Bueng 3 | Bueng 4 | Bueng 5 | Bueng 6 | Bueng 7 | Bueng 8 | Bueng 9 | Bueng 10 | Bueng 11 |
|----------|---------|---------|---------|---------|---------|---------|---------|---------|---------|----------|----------|
| Bueng 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Bueng 2 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Bueng 3 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Bueng 4 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Bueng 5 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| Bueng 6 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 |
| Bueng 7 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 |
| Bueng 8 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 |
| Bueng 9 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 |
| Bueng 10 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 |
| Bueng 11 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |

Gambar 3.5 contoh binari calssification

Voting merupakan data hasil dari decision tree yang terdapat pada random forest. Dimana hasil data tersebut di gunakan sebagai acuan untuk hasil dari random forest. sebagai contoh misalkan pada satu random forest terdapat enam decision tree untuk menentukan jenis pekerjaan orang, pada decision tree ke satu menyimpulkan bahwa pekerjaanya yaitu dosen , pada decision tree ke dua yaitu dosen kemudian pada decision tiga dosen , pada decision tree ke empat yaitu pekerja kantoran, pada decision tree ke lima yaitu pekerja kantoran dan pada decision tree ke enam yaitu dosen. maka pada random forest dapat menyimpulkan hasilnya yaitu dosen.

**Gambar 3.6** contoh binari calssification

3.1.2 Praktikum

1. pandas

pada baris ke satu yaitu perintah mengimport library padas pada python atau anaconda kemudian di inisialisasikan menjadi kue. selanjutnya pada baris ke 3 terdapat nama variabel yaitu nama_kue_tradisional = yang di dalamnya terdapat tiga nama field yakni Name Kue, harga satuan dan terbilang kemudian pada baris ke tujuh terdapat variabel baru bernama Data_kue = kemudian didalamnya mendeskripsikan kue berdasarkan tipe DataFrame yang berisi variabel nama_kue_tradisional selanjutnya data tersebut di cetak pada console dengan perintah (Data_kue).

```

1 import pandas as kue
2 nama_kue = { 'Nama Kue' : [ 'Cuhcur' , 'Putri Noong' , 'Bugis' , 'Papais' , 'Ali-Ali' ] ,
3   'Harga Satuan' : [ 2000 , 5000 , 1500 , 2500 , 1000 ] , 'Terbilang' : [
4     'Dua Ribu Rupiah' , 'Lima Ribu Rupiah' ,
5     'Seribu Lima Ratus Rupiah' , 'Dua Ribu Limaratus Rupiah' , 'Seribu Rupiah' ] }
6 Data_kue = kue.DataFrame(nama_kue)
7 print(Data_kue)
  
```

| | Nama Kue | Harga Satuan | Terbilang |
|---|-------------|--------------|---------------------------|
| 0 | Cucur | 2000 | Dua Ribu Rupiah |
| 1 | Putri Noong | 5000 | Lima Ribu Rupiah |
| 2 | Bugis | 1500 | Seribu Lima Ratus Rupiah |
| 3 | Papais | 2500 | Dua Ribu Limaratus Rupiah |
| 4 | Ali-Ali | 1000 | Seribu Rupiah |

Process finished with exit code 0

Gambar 3.7 hasil

2. numpy

Arti tiap baris codingan pada aplikasi sederhana numpy adalah sebagai berikut : pada baris ke satu yaitu mengimport numpy yang di inisialisasi menjadi np kemudian pada baris ke tiga dibut variabel lala yang berisi numpy bertipekan arange 6 yang berarti berisi nilai array dari 0 sampai 5 kemudian pada baris ke empat di cetak hasilnya dengan memasukan perintah print (lala) selanjutnya yaitu membuat nilai array tiga dimensi pada baris ke tujuh dengan cara membuat variabel botak yang berisi rank nilainya kemudian dimensinya yaitu 4 3 3 kemudian variabel tersebut di print. selanjutnya pada baris ke 12 dibuat variabel nilai_array_1 dengan isian nilai array 1 2 3 4 kemudian pada baris ke 13 di buat variabe nilai_array_2 dengan nilai array 20 30 40 dan 50 selanjutnya pada baris ke 14 dibut nilai variabel Nilai_array_3 dengan rank 4 yang berarti berisi nilai dari 0 sampai 3 setelah itu di buat variabel hasil dimana isinya yaitu penjumlahan nilai_array_1+Nilai_array_3+Nilai_array_3 setelah itu nilai_array_1 , Nilai_array_3, dan Hasil di prin untuk melihat nilai dari array tersebut.

```

1 import numpy as np
2
3 lala = np.arange(6)
4 print(lala)
5
6 #array 3dimensi
7 lalae = np.arange(36).reshape(4,3,3)
8 print(lalae)
9
10 #penjumlahan array
11 nilai_array_1 = np.array([1,2,3,4])
12 nilai_array_2 = np.array([20,30,40,50])
13 nilai_array_3 = np.arange(4)
14 Hasil = nilai_array_1+nilai_array_3=nilai_array_3
15 print(nilai_array_1)
16 print(nilai_array_3)
17 print(Hasil)

```

3. matplotlib

Arti tiap baris aplikasi sederhana matplotlib pada baris ke satu yaitu memasukan library matplotlib.pyplot yang di definisikan menjadi plt ke-

```

Run: 2,2 <
[0 1 2 3 4 5]
[[[ 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]]]
[1 2 3 4]
[0 1 2 3]
[ 1 4 7 10]

Process finished with exit code 0

```

Gambar 3.8 hasil

mudian membuat variabel kelas_ti3 pada baris ke tiga yang berisi label setelah itu di buat variabel jumlah_mhs3 pada baris ke empat yang berisi nilai dari setiap label tersebut. begitu juga pada baris ke emam dan ke tujuh kemudian pada baris ke sembilan matplotlib mendefinisikan gambar dengan ukurannya dan pada baris ke 10 di dekralasikan subplot setelah itu pada baris ke 11 matplotlib mendefinisikan jenis grafik yang digunakan dan dimasukan variabel kelas dan jumlah_mhs. begitujuga oada baris ke 13 14 dan 15 setelah itu di buat title pada baris ke 17 dan matplotlib di show untuk mendapatkan hasil dari grafiknya.

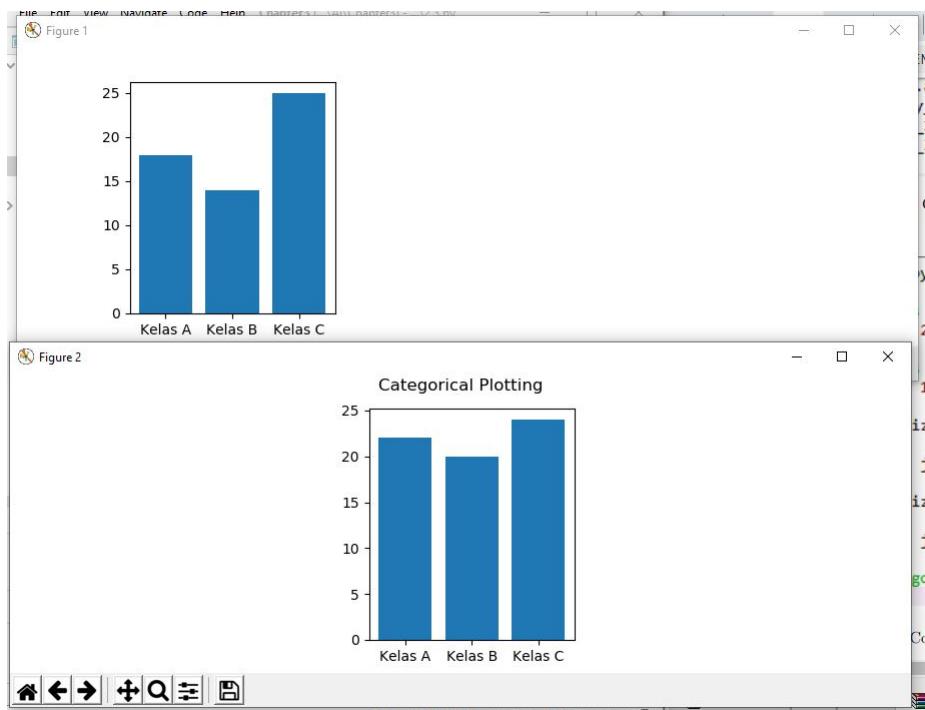
```

1 import matplotlib.pyplot as plt
2
3 kelas_ti3 = ['Kelas A', 'Kelas B', 'Kelas C']
4 jumlah_mhs3 = [18, 14, 25]
5
6 kelas_ti2 = ['Kelas A', 'Kelas B', 'Kelas C']
7 jumlah_mhs2 = [22, 20, 24]
8
9 plt.figure(1, figsize=(9,3))
10 plt.subplot(131)
11 plt.bar(kelas_ti3, jumlah_mhs3)
12 plt.figure(2, figsize=(9,3))
13 plt.subplot(132)
14 plt.bar(kelas_ti2, jumlah_mhs2)
15 plt.suptitle('Categorical Plotting')
16 plt.show()

```

4. Random Forest

Arti tiap baris hasil codingan random forest pada baris pertama random forest di import dari sklearn dengan ketentuan yaitu maksimal isi dari de-



Gambar 3.9 hasil

cision tree berisi 50 data dengan keadaan random dan dengan estimators 100 data ini berada dalam variabel clf kemudian setelah itu variabel clf di running berdasarkan data training dan data label yang telah di definisikan jumlahnya. kemudian variabel clf di running berdasarkan data training paling atas untuk memunculkan hasil data training lima paling atas. setelah itu data tersebut di di running scorenya untuk melihat tingkat akurasi yang dia kerjakan maka tingkat akurasi yang dihasilkan berada pada kisaran 0,437 atau kisaran 43 %.

```

1 from sklearn.ensemble import RandomForestClassifier
2 clf = RandomForestClassifier(max_features=50, random_state=0,
3     n_estimators=100)
4 clf.fit(df_train_att, df_train_label)
5 print(clf.predict(df_train_att.head()))
6 clf.score(df_test_att, df_test_label)

```

5. Confusion Matrix

arti codingan pada hasil tiap codingan confusion matrix pada baris pertama codingan tersebut mendeskripsikan atau mengimport confusion matrix dari sklearn kemudian dibuat variabel pred_labels dengan di isikan clf prdic df_test_att setelah itu membuat variabel cm yang isinya terdapat

```

Terminal: Local + 
>>>
>>> from sklearn.ensemble import RandomForestClassifier
>>> clf = RandomForestClassifier(max_features=50, random_state=0, n_estimators=100)
>>> clf.fit(df_train_att, df_train_label)
RandomForestClassifier(bootstrap=True, ccp_alpha=0.0, class_weight=None,
                      criterion='gini', max_depth=None, max_features=50,
                      max_leaf_nodes=None, max_samples=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)
>>> print(clf.predict(df_train_att.head()))
[[0 0 0 ... 0 0 1]
 [0 0 0 ... 0 0 1]
 [0 0 0 ... 0 0 1]
 [0 0 0 ... 0 1 0]
 [0 0 0 ... 0 0 0]]
>>> clf.score(df_test_att, df_test_label)
0.009503695881731784
>>> []

```

Gambar 3.10 hasil

data yang di buat confusion matrix berdasarkan data test setelah variabel cm akan di running yang mana akan menghasilkan gambar berupa matrix matrix tersebut berisi nilai nilai kebenaran yang mendekati nilai benar atau mutlak nilai benar.

```

1 from sklearn.metrics import confusion_matrix
2 pred_labels = clf.predict(df_test_att)
3 cm = confusion_matrix(df_test_label, pred_labels)
4 cm

```

```

>>> from sklearn.metrics import confusion_matrix
>>> pred_labels = clf.predict(df_test_att)
>>> cm = confusion_matrix(df_test_label, pred_labels)
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
    File "C:\Users\najib\AppData\Local\Programs\Python\Python37\lib\site-packages\sklearn\metrics\_classification.py", line 270, in confusion_matrix
      raise ValueError("%s is not supported" % y_type)
ValueError: multilabel-indicator is not supported
>>> cm
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
NameError: name 'cm' is not defined

```

Gambar 3.11 hasil

6. SVM dan Decision Tree

Arti dari setiap baris hasil codingan decision tree dan SVM pada tree masukan terlebih dahulu library tree setelah itu buat variabel clftree yang berisi decision tree setelah itu masukan nilai data training dan label yang telah di deklarasikan tadi setelah itu running variabel tersebut untuk mendapatkan score 0,266 kisaran 26 sampai 27 % akurasinya kemudian pada svm juga hampir sama masukan terlebih dahulu librarynya

setelah itu buat variabel clfsvm yang berarti berisi nilai data training dan data label dan pendeklarasian svm itu sendiri setelah itu di running untuk mendapatkan nilai akurasinya atau score sebesar 0,283 atau dalam kisaran 23 %.

```

1 from sklearn import tree
2 clftree = tree.DecisionTreeClassifier()
3 clftree.fit(df_train_att, df_train_label)
4 clftree.score(df_test_att, df_test_label)
5
6 from sklearn import svm
7 clfsvm = svm.SVC()
8 clfsvm.fit(df_train_att, df_train_label)
9 clfsvm.score(df_test_att, df_test_label)
```

```

>>> from sklearn import tree
>>> clftrree = tree.DecisionTreeClassifier()
>>> clftrree.fit(df_train_att, df_train_label)
DecisionTreeClassifier(ccp_alpha=0.0, class_weight=None, criterion='gini',
                      max_depth=None, max_features=None, 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, presort='deprecated',
                      random_state=None, splitter='best')
>>> clftrree.score(df_test_att, df_test_label)
0.0008044333214158026
>>> █
```

Gambar 3.12 hasil

7. Cross Validation

arti dari setiap baris hasil cross validation pada gambar?? tersebut diperlihatkan codingan error dikarenakan data training terlalu besar maka untuk mengatasinya dapat dilihat pada sub bab penanganan error

```

1 from sklearn.model_selection import cross_val_score
2 scores = cross_val_score(clf, df_train_att, df_train_label,
                           cv=5)
3 print("Accuracy: %.2f (+/- %.2f)" % (scores.mean(), scores.
                                         std() * 2))
```

```

>>> from sklearn.model_selection import cross_val_score
>>> scores = cross_val_score(clf, df_train_att, df_train_label, cv=5)
>>> print("Accuracy: %.2f (+/- %.2f)" % (scores.mean(), scores.std() * 2))
Accuracy: 0.00 (+/- 0.00)
>>> █
```

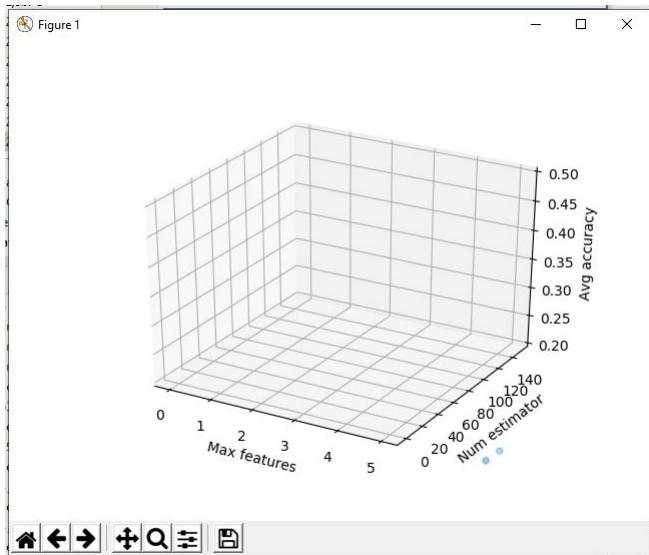
Gambar 3.13 hasil

8. program pengamatan arti dari hasil program pengamatan. perogram pengamatan ini menggunakan library matplotlib supaya hasil dari presentase hasil random forest, svm dan decision tree dapat di bandingkan dengan membuat variabel X Y Z kemudian memberikan label untuk setiap dimensinya untuk lebih jelas dapat dilihat gambar ?? yang menunjukan hasil perbandingan presentase dari tiga metode tersebut.

```

1 import matplotlib.pyplot as plt
2 from mpl_toolkits.mplot3d import Axes3D
3 from matplotlib import cm
4 fig = plt.figure()
5 fig.clf()
6 ax = fig.gca(projection='3d')
7 x = rf_params[:,0]
8 y = rf_params[:,1]
9 z = rf_params[:,2]
10 ax.scatter(x, y, z)
11 ax.set_zlim(0.2, 0.5)
12 ax.set_xlabel('Max features')
13 ax.set_ylabel('Num estimator')
14 ax.set_zlabel('Avg accuracy')
15 plt.show()

```



Gambar 3.14 hasil

3.1.3 Penanganan Error / cokro

Screenshot error

- Untuk gambar screenshot error

Code Errornya

- kode error pada screenshot ke satu yaitu dikarenakan `clfsvm.fit(df_train_att, df_train_label)` dikarenakan data trainingnya terlalu besar sehingga komputernya error.

```
In [48]: scoressvm = cross_val_score(clfsvm, df_train_att, df_train_label, cv=5)
...: print("Accuracy: %.2f (+/- %.2f)" % (scoressvm.mean(), scoressvm.std() * 2))
Traceback (most recent call last):

File "<ipython-input-48-d7a7153ce4e9>", line 1, in <module>
    scoressvm = cross_val_score(clfsvm, df_train_att, df_train_label, cv=5)

File "C:\ProgramData\Anaconda3\lib\site-packages\sklearn\model_selection\_validation.py", line 402, in cross_val_score
    error_score=error_score)

File "C:\ProgramData\Anaconda3\lib\site-packages\sklearn\model_selection\_validation.py", line 240, in cross_validate
    for train, test in cv.split(X, y, groups))

File "C:\ProgramData\Anaconda3\lib\site-packages\sklearn\externals\joblib\parallel.py", line 917, in __call__
    if self.dispatch_one_batch(iterator):

File "C:\ProgramData\Anaconda3\lib\site-packages\sklearn\externals\joblib\parallel.py", line 759, in dispatch_one_batch
    self._dispatch(tasks)

File "C:\ProgramData\Anaconda3\lib\site-packages\sklearn\externals\joblib\parallel.py", line 716, in _dispatch
    job = self._backend.apply_async(batch, callback=cb)

File "C:\ProgramData\Anaconda3\lib\site-packages\sklearn\externals\joblib\_parallel_backends.py", line 182, in apply_async
    result = ImmediateResult(func)

File "C:\ProgramData\Anaconda3\lib\site-packages\sklearn\externals\joblib\_parallel_backends.py", line 549, in __init__
    self.results = batch()

File "C:\ProgramData\Anaconda3\lib\site-packages\sklearn\externals\joblib\parallel.py", line 225, in __call__
    for func, args, kwargs in self.items]

File "C:\ProgramData\Anaconda3\lib\site-packages\sklearn\externals\joblib\parallel.py", line 225, in <listcomp>
```

Gambar 3.15 hasil

- untuk kode error pada screen shoot ke 2 sampai ke 4 dikarenakan pada kode berikut scores = cross_val_score(clf, df_train_att, df_train_label, cv=5) scorestree = cross_val_score(clftree, df_train_att, df_train_label, cv=5) dan scoressvm = cross_val_score(clfsvm, df_train_att, df_train_label, cv=5) hal ini di karenakan data trainingterlalu besar sehingga berdampak pada komputer sehingga library dari python tidak mampu mengolah data dan hasilnya menjadi error.

Solusi Untuk mengatasi Error

- solusinya untuk yang ke satu yaitu dengan cara merestart spyder atau mematikannya kemudian nyalakan kembali setelah itu jalankan code yang error tersebut di CMD cika dalam python CMD jalam maka bisa di running. setelah itu buka kembali spyder dan jalankan codingan dari awal hingga pada bagian SVM tunggu sebenar sampai muncul nilai akurasinya.
- solusi untuk mengatasi error tersebut yaitu dengan cara merubah bobot data pada data training.

```
df = imgatt2.join(imglabels)
df = df.sample(frac=1)
df_att = df.iloc[:, :312]
df_label = df.iloc[:, :312]
df_train_att = df_att[:600]
df_train_label = df_label[:600]
df_test_att = df_att[600:]
df_test_label = df_label[600:]
```

Gambar 3.16 hasil

BAB 4

CHAPTER 4

BAB 5

CHAPTER 5

BAB 6

CHAPTER 6

BAB 7

CHAPTER 7
