

Modul Praktikum

Kecerdasan Buatan



Rolly Maulana Awangga

0410118609

Applied Bachelor of Informatics Engineering

Program Studi D4 Teknik Informatika

Applied Bachelor Program of Informatics Engineering

Politeknik Pos Indonesia

Bandung 2019

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

Acknowledgements

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

Abstract

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

Contents

1 Mengenal Kecerdasan Buatan dan Scikit-Learn	1
1.1 Teori	1
1.2 Instalasi	2
1.3 Penanganan Error	2
1.4 Ahmad Syafrizal Huda/1164062	2
1.4.1 Teori	2
1.4.2 Instalasi	4
1.4.2.1 Instalasi Library Scikit dari Anaconda	4
1.4.2.2 Mencoba Loading an example Dataset	8
1.4.2.3 Learning and Predicting	9
1.4.2.4 Model Persistence	10
1.4.2.5 Conventions	12
1.4.3 Penanganan eror	17
1.4.3.1 ScreenShoot Eror	17
1.4.3.2 Tuliskan Kode Eror dan Jenis Erornya	17
1.4.3.3 Solusi Pemecahan Masalah Error	19
2 Related Works	20
2.1 Ahmad Syafrizal Huda/1164062	20
2.1.1 Teori	20
2.1.2 Scikit-learn	25
2.1.3 Penanganan Eror	32
3 Methods	35
3.1 Ahmad Syafrizal Huda / 1164062	35
3.1.1 Teori	35
3.1.2 Praktek Program	41
3.1.3 Penanganan Eror	56

4 Experiment and Result	59
4.1 Experiment	59
4.2 Result	59
4.3 Ahmad Syafrizal Huda/1164062	59
4.3.1 Teori	59
4.3.2 Praktek Program	63
4.3.3 Penanganan Eror	66
5 Conclusion	69
5.1 Ahmad Syafrizal Huda/1164062	69
5.1.1 Teori	69
5.1.2 Praktek Program	72
5.1.3 Penanganan Eror	83
6 Discussion	85
6.1 Ahmad Syafrizal Huda/1164062	85
6.1.1 Teori	85
6.1.2 Praktek Program	90
6.1.3 Penanganan Eror	97
7 Discussion	99
7.1 Ahmad Syafrizal Huda/1164062	99
7.1.1 Teori	99
7.1.2 Praktek Program	104
7.1.3 Penanganan Eror	126
8 Discussion	128
9 Discussion	129
10 Discussion	130
11 Discussion	131
12 Discussion	132
13 Discussion	133
14 Discussion	134

A Form Penilaian Jurnal	135
B FAQ	138
Bibliography	140

List of Figures

1.1	Download Anaconda.	4
1.2	Langkah pertama instalasi anaconda.	4
1.3	Langkah kedua instalasi anaconda.	5
1.4	Langkah ketiga instalasi anaconda.	5
1.5	Langkah terakhir instalasi anaconda.	6
1.6	Langkah pertama instalasi scikit pada CMD.	6
1.7	Langkah kedua instalasi scikit pada CMD.	7
1.8	Langkah ketiga instalasi scikit pada CMD.	7
1.9	Langkah compile code pada python anaconda.	7
1.10	Hasil Tampilan 1.	8
1.11	Hasil Tampilan 2.	8
1.12	Hasil Tampilan 3.	9
1.13	Hasil Tampilan 4.	9
1.14	Hasil Tampilan 5.	10
1.15	Hasil Tampilan 6.	10
1.16	Hasil Tampilan 7.	11
1.17	Hasil Tampilan 8.	11
1.18	Hasil Tampilan 9.	12
1.19	Hasil Tampilan 10.	13
1.20	Hasil Tampilan 11.	13
1.21	Hasil Tampilan 12.	14
1.22	Hasil Tampilan 13.	14
1.23	Hasil Tampilan 14.	14
1.24	Hasil Tampilan 15.	14
1.25	Hasil Tampilan 16.	15
1.26	Hasil Tampilan 17.	16
1.27	Hasil Tampilan 18.	16
1.28	Hasil Tampilan 19.	16

1.29	Hasil Tampilan 20.	17
1.30	Hasil Tampilan 21.	17
1.31	Hasil Tampilan 22.	17
1.32	Hasil Tampilan Error.	18
1.33	Hasil Tampilan Install joblib.	19
1.34	Hasil Tampilan Uji coba perintah joblib.	19
2.1	Binary Classification.	21
2.2	Supervised Learning.	22
2.3	Unsupervised Learning.	22
2.4	Clustering.	22
2.5	Evaluasi dan Akurasi.	23
2.6	K-fold Cross Validation.	24
2.7	Decision Tree.	25
2.8	Gain.	26
2.9	Hasil Codingan No 1.	26
2.10	Hasil Codingan No 2.	27
2.11	Hasil Codingan No 3.	28
2.12	Hasil Codingan No 4.	28
2.13	Hasil Codingan No 5.	29
2.14	Hasil Codingan No 6.	29
2.15	Hasil Codingan No 7.	30
2.16	Hasil Codingan No 8.	30
2.17	Hasil Codingan No 9.	30
2.18	Hasil Codingan No 10.	31
2.19	Hasil Codingan No 11.	32
2.20	Hasil Codingan No 12.	33
2.21	Hasil Gambar Eror No 6.	33
2.22	Hasil Gambar Penanganan Eror No 6.	34
3.1	Random Forest	35
3.2	Hasil Dataset	36
3.3	Confusion Matrix	39
3.4	Voting	40
3.5	Aplikasi Menggunakan Pandas	41
3.6	Aplikasi Menggunakan Numpy	42
3.7	Aplikasi Menggunakan Matplotlib	43

3.8	Hasil 1 Random Forest	43
3.9	Hasil 2 Random Forest	44
3.10	Hasil 3 Random Forest	44
3.11	Hasil 4 Random Forest	44
3.12	Hasil 5 Random Forest	45
3.13	Hasil 6 Random Forest	45
3.14	Hasil 7 Random Forest	45
3.15	Hasil 8 Random Forest	46
3.16	Hasil 9 Random Forest	46
3.17	Hasil 10 Random Forest	47
3.18	Hasil 11 Random Forest	47
3.19	Hasil 11 Random Forest	47
3.20	Hasil 12 Random Forest	48
3.21	Hasil 13 Random Forest	49
3.22	Hasil 14 Random Forest	49
3.23	Hasil 15 Random Forest	49
3.24	Hasil 16 Random Forest	50
3.25	Hasil 17 Random Forest	50
3.26	Hasil 1 Confusion Matrix	50
3.27	Hasil 2 Confusion Matrix	51
3.28	Hasil 3 Confusion Matrix	51
3.29	Hasil 4 Confusion Matrix	52
3.30	Hasil 5 Confusion Matrix	53
3.31	Hasil 1 Klasifikasi SVM dan Decision Tree	54
3.32	Hasil 2 Klasifikasi SVM dan Decision Tree	54
3.33	Hasil 1 Cross Validation	54
3.34	Hasil 2 Cross Validation	55
3.35	Hasil 3 Cross Validation	55
3.36	Hasil 1 Pengamatan Komponen Informasi	55
3.37	Hasil 2 Pengamatan Komponen Informasi	56
3.38	Eror	57
4.1	Klasifikasi Teks	60
4.2	Klasifikasi Bunga	60
4.3	Comment Youtube	61
4.4	Bag Of Words	62

4.5	TF-IDF	62
4.6	Data Dummy 500 Data	63
4.7	Membagi 2 Dataframe	63
4.8	Vektorisasi dan Klasifikasi Data	64
4.9	Data Content	64
4.10	DataFrame Kata-kata Pada Content	64
4.11	Klasifikasi SVM Dari Data Vektorisasi	65
4.12	Klasifikasi Decision Tree Dari Data Vektorisasi	65
4.13	Plot Confusion Matrix Menggunakan Matplotlib	66
4.14	Program Cross Validation Pada Data Vektorisasi	66
4.15	Program Pengamatan Komponen Informasi	67
4.16	Eror Pada Coding SVM	67
5.1	Ilustrasi Soal No. 1	69
5.2	Ilustrasi Soal No. 2	70
5.3	Ilustrasi Soal No. 3	71
5.4	Ilustrasi Soal No. 4	71
5.5	Ilustrasi Soal No. 5	72
5.6	Ilustrasi Soal No. 6	72
5.7	Love	73
5.8	Faith	73
5.9	Fall	73
5.10	Sick	74
5.11	Clear	74
5.12	Shine	74
5.13	Bag	74
5.14	Car	75
5.15	Wash	75
5.16	Motor	75
5.17	Cycle	76
5.18	Similariti Pada Kata Motor dan Cycle	76
5.19	Similariti Pada Kata Wash dan Motor	76
5.20	Extract_Words	77
5.21	Permute Sentences	77
5.22	Gensim TaggedDocument dan Doc2vec	78
5.23	Aclimbdb	79

5.24	Aclimbdb	79
5.25	Aclimbdb	80
5.26	Infer Code	81
5.27	Cosine Similarity	82
5.28	Cross Validation Metode 1	83
5.29	Cross Validation Metode 2	83
5.30	Cross Validation Metode 3	83
5.31	eror	84
6.1	Suara ke MFCC	85
6.2	Neural Network	86
6.3	Pembobotan Neural Network	87
6.4	Aktivasi Neural Network	88
6.5	Membaca Hasil Plot dari MFCC	89
6.6	One-Hot Encoding	89
6.7	np.unique	90
6.8	to_categorical	90
6.9	One-Hot Encoding	91
6.10	GTZAN	92
6.11	Display MFCC	92
6.12	Generate Features and Labels	94
6.13	Training Data Sebesar 80%	94
6.14	Fungsi Compile	95
6.15	Fungsi Fit	96
6.16	Fungsi Evaluate	97
6.17	Fungsi Predict	97
6.18	Eror	97
7.1	Tokenizer	100
7.2	StartifiedKFold	100
7.3	Fungsi Tokenizer	102
7.4	Matrix TFID	102
7.5	Matrix Training dan Testing	103
7.6	Konvolusi	105
7.7	Algoritma Perhitungan Konvolusi	105
7.8	In[1]	106
7.9	In[2]	107

7.10	In[3]	108
7.11	In[4]	110
7.12	In[5]	111
7.13	In[6]	111
7.14	In[7]	112
7.15	In[8]	113
7.16	In[9]	114
7.17	In[10]	115
7.18	In[11]	116
7.19	In[12]	117
7.20	In[13]	120
7.21	In[14]	121
7.22	In[15]	122
7.23	In[16]	123
7.24	In[17]	123
7.25	In[18]	124
7.26	In[19]	126
7.27	In[20]	126
7.28	Eror	127
A.1	Form nilai bagian 1.	136
A.2	form nilai bagian 2.	137

Chapter 1

Mengenal Kecerdasan Buatan dan Scikit-Learn

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

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

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

1.1 Teori

Praktek teori penunjang yang dikerjakan :

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

1.2 Instalasi

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

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

1.3 Penanganan Error

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

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

1.4 Ahmad Syafrizal Huda/1164062

1.4.1 Teori

1. 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[1].

2. 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[4].

1.4.2 Instalasi

1.4.2.1 Instalasi Library Scikit dari Anaconda

1. Download aplikasi Anaconda terlebih dahulu. Lihat pada gambar 7.4

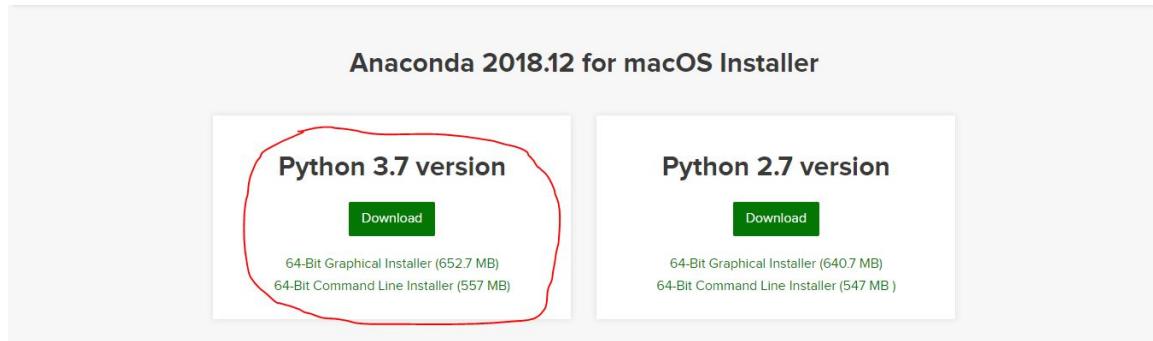


Figure 1.1: Download Anaconda.

2. Install aplikasi Anaconda yang sudah di download tadi. Lihat pada gambar 7.5

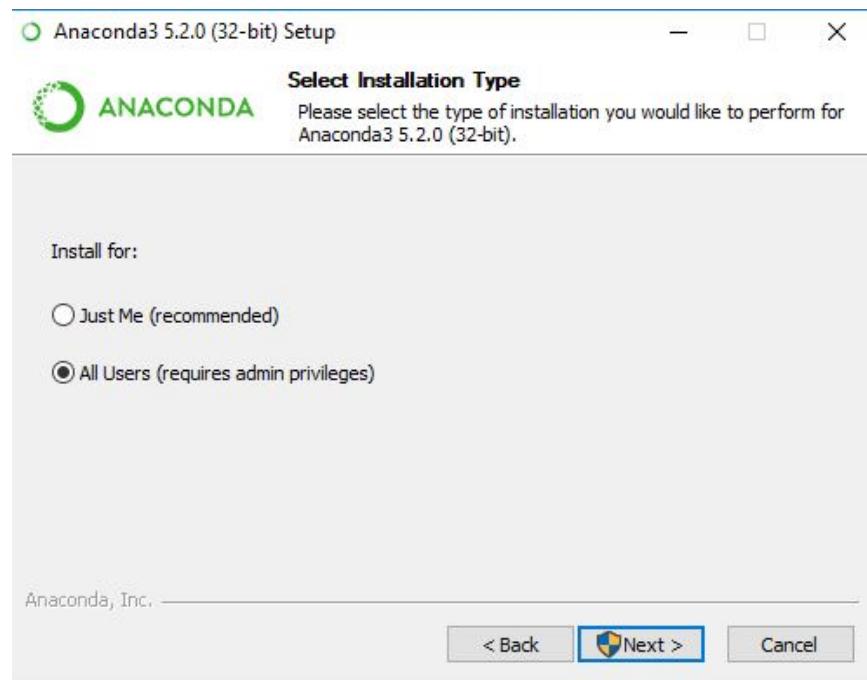


Figure 1.2: Langkah pertama instalasi anaconda.

3. Simpan aplikasi sesuai folder yang kita pilih lalu next. Lihat pada gambar 7.6
4. Centang Keduanya lalu tekan tombol install. Lihat pada gambar 7.7

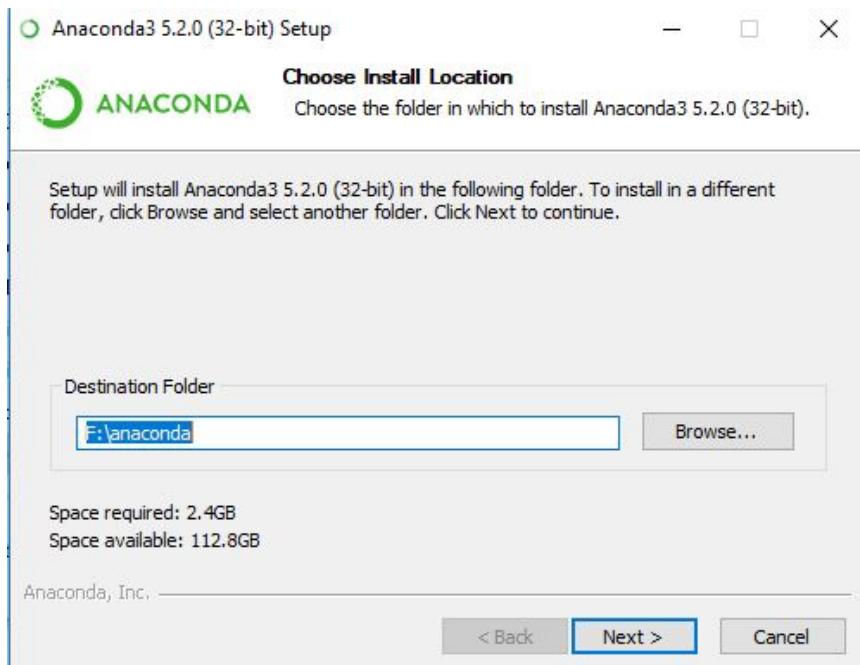


Figure 1.3: Langkah kedua instalasi anaconda.

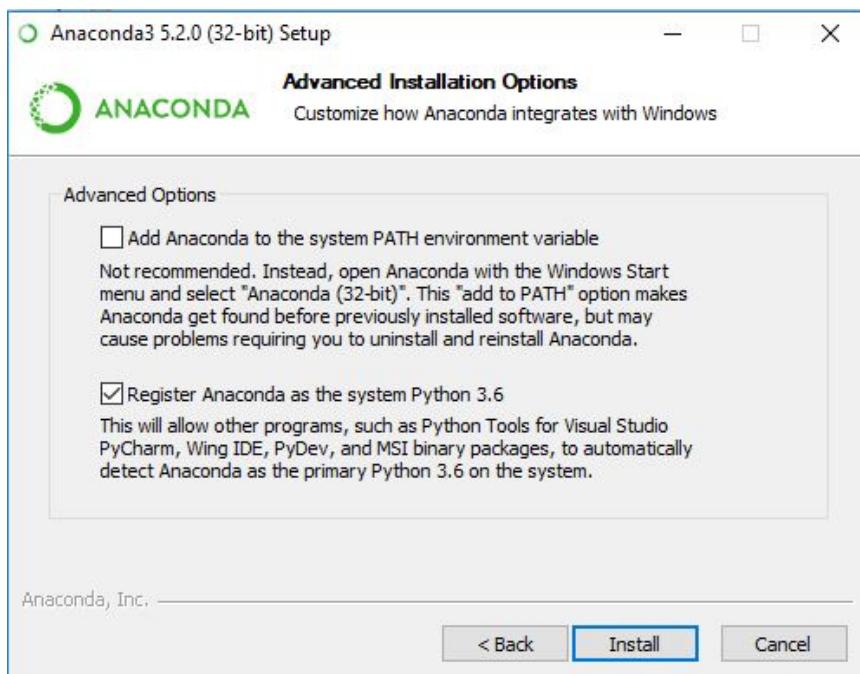


Figure 1.4: Langkah ketiga instalasi anaconda.

5. Setelah itu tunggu sampai proses instalasi selesai lalu jika sudah tekan tombol finish. Lihat pada gambar 7.8

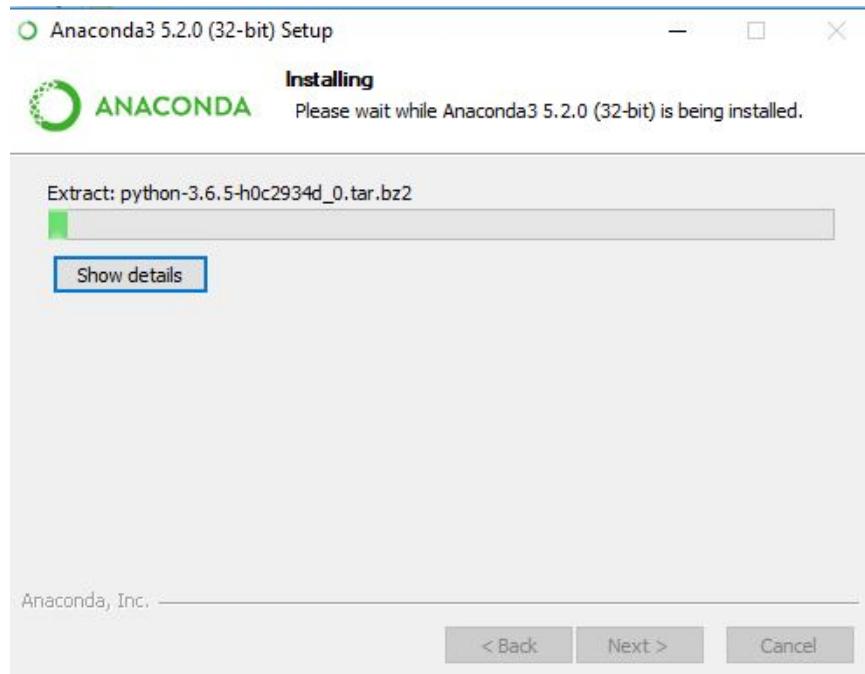


Figure 1.5: Langkah terakhir instalasi anaconda.

6. Lalu buka command prompt anda dan tuliskan perintah berikut ini untuk mengecek apakah aplikasinya sudah terinstall. Lihat pada gambar 7.9

```
PS C:\Users\HUDA> conda --version
conda 4.5.4

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

Figure 1.6: Langkah pertama instalasi scikit pada CMD.

7. Kemudian ketikkan perinta pip install -U scikit-learn seperti gambar berikut. Lihat pada gambar 7.10
8. Lalujika sudah ketikkan juga perintah conda install scikit-learn. Lihat pada gambar 7.11
9. Hasil compile dari beberapa code yang mempunyai variable explorer. Lihat pada gambar 7.12

```
C:\Users\HUDA>pip install -U scikit-learn
Collecting scikit-learn
  Downloading https://files.pythondistro.org/packages/ee/c8/c89ebcd0d7dbba6eef0d222daabd257da3c28a967dd7c352d4272b2e1cef0/scikit_learn-0.20.2-cp36-cp36m-win32.whl (4.3MB)
    100% |████████████████████████████████| 4.3MB 42kB/s
Requirement not upgraded as not directly required: numpy>=1.8.2 in f:\anaconda\lib\site-packages (from scikit-learn) (1.14.3)
Requirement not upgraded as not directly required: scipy>=0.13.3 in f:\anaconda\lib\site-packages (from scikit-learn) (1.1.0)
Installing collected packages: scikit-learn
  Found existing installation: scikit-learn 0.19.1
    Uninstalling scikit-learn-0.19.1:
      Successfully uninstalled scikit-learn-0.19.1
Successfully installed scikit-learn-0.20.2
You are using pip version 10.0.1, however version 19.0.3 is available.
You should consider upgrading via the 'python -m pip install --upgrade pip' command.
```

Figure 1.7: Langkah kedua instalasi scikit pada CMD.

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

## Package Plan ##

environment location: F:\anaconda

added / updated specs:
- scikit-learn

The following packages will be downloaded:

  package          | build
  -- -- -- -- -- -- | --
  conda-4.6.7      | py36_0      1.7 MB

The following packages will be UPDATED:

  conda: 4.5.4-py36_0 --> 4.6.7-py36_0

Proceed ([y]/n)? y

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

Figure 1.8: Langkah ketiga instalasi scikit pada CMD.

```
C:\Users\HUDA>python
Python 3.6.5 |Anaconda, Inc.| (default, Mar 29 2018, 13:23:52) [MSC v.1900 32 bit (Intel)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> from sklearn.metrics import confusion_matrix
>>> y_true = [2, 0, 2, 2, 0, 1]
>>> y_pred = [0, 0, 2, 2, 0, 2]
>>> confusion_matrix(y_true, y_pred)
array([[2, 0, 0],
       [0, 0, 1],
       [1, 0, 2]], dtype=int64)
>>> y_true = ["cat", "ant", "cat", "ant", "bird"]
>>> y_pred = ["ant", "ant", "cat", "ant", "cat"]
>>> confusion_matrix(y_true, y_pred, labels=["ant", "bird", "cat"])
array([[2, 0, 0],
       [0, 0, 1],
       [1, 0, 2]], dtype=int64)
>>> tn, fp, fn, tp = confusion_matrix([0, 1, 0, 1], [1, 1, 1, 0]).ravel()
>>> (tn, fp, fn, tp)
(0, 2, 1, 1)
>>>
```

Figure 1.9: Langkah compile code pada python anaconda.

1.4.2.2 Mencoba Loading an example Dataset

- `from sklearn import datasets`

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

- `iris = datasets.load_iris()`

(pada baris kedua ini dimana iris merupakan suatu estimator/parameter yang berfungsi untuk mengambil data pada item datasets.load_iris).

- `digits = datasets.load_digits()`

(pada baris ketiga ini dimana digits merupakan suatu estimator/parameter yang berfungsi untuk mengambil data pada item datasets.load_digits).

- `print(digits.data)`

(pada baris keempat ini merupakan perintah yang berfungsi untuk menampilkan estimator/parameter yang dipanggil pada item digits.data dan menampilkan outputannya) Lihat gambar 7.13

```
[[ 0.  0.  5. ...  0.  0.  0.]
 [ 0.  0.  0. ... 10.  0.  0.]
 [ 0.  0.  0. ... 16.  9.  0.]
 ...
 [ 0.  0.  1. ...  6.  0.  0.]
 [ 0.  0.  2. ... 12.  0.  0.]
 [ 0.  0. 10. ... 12.  1.  0.]]
```

Figure 1.10: Hasil Tampilan 1.

- `digits.target`

(barisan ini untuk mengambil target pada estimator/parameter digits dan menampilkan outputannya) Lihat gambar 7.14

```
array([0, 1, 2, ..., 8, 9, 8])
```

Figure 1.11: Hasil Tampilan 2.

- `digits.images[0]`

(barisan ini untuk mengambil `images[0]` pada estimator/parameter `digits` dan menampilkan outputannya) Lihat gambar 7.15

```
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.]])
```

Figure 1.12: Hasil Tampilan 3.

1.4.2.3 Learning and Predicting

- `from sklearn import svm`

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

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

(pada baris kedua ini `clf` sebagai estimator/parameter, `svm.SVC` sebagai class, `gamma` sebagai parameter untuk menetapkan nilai secara manual).

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

(pada baris ketiga ini `clf` sebagai estimator/parameter, `fit` sebagai metode, `digits.data` sebagai item, `[:-1]` sebagai syntax pythonnya dan menampilkan outputannya) Lihat gambar 7.16.

```
SVC(C=100.0, cache_size=200, class_weight=None, coef0=0.0,
    decision_function_shape='ovr', degree=3, gamma=0.001, kernel='rbf',
    max_iter=-1, probability=False, random_state=None, shrinking=True,
    tol=0.001, verbose=False)
```

Figure 1.13: Hasil Tampilan 4.

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

(pada baris terakhir ini `clf` sebagai estimator/parameter, `predict` sebagai metode lainnya, `digits.data` sebagai item dan menampilkan outputannya) Lihat gambar 7.17

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

Figure 1.14: Hasil Tampilan 5.

1.4.2.4 Model Persistence

- `from sklearn import svm`

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

- `from sklearn import datasets`

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

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

(pada baris ketiga ini clf sebagai estimator/parameter, svm.SVC sebagai class, gamma sebagai parameter untuk menetapkan nilai secara manual dengan nilai scale).

- `iris = datasets.load_iris()`

(pada baris keempat ini iris sebagai estimator/parameter, datasets.load_iris() sebagai item dari suatu nilai).

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

(pada baris kelima ini X, y sebagai estimator/parameter, iris.data, iris.target sebagai item dari 2 nilai yang ada).

- `clf.fit(X, y)`

(pada baris keenam ini clf sebagai estimator/parameter dengan menggunakan metode fit untuk memanggil estimator X, y dengan outputannya) Lihat gambar 7.18

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

Figure 1.15: Hasil Tampilan 6.

- `import pickle`
(pickle merupakan sebuah class yang di import).
- `s = pickle.dumps(clf)`
(pada baris ini s sebagai estimator/parameter dengan pickle.dumps merupakan suatu nilai/item dari estimator/parameter clf)
- `clf2 = pickle.loads(s)`
(pada baris ini clf2 sebagai estimator/parameter, pickle.loads sebagai suatu item, dan s sebagai estimator/parameter yang dipanggil)
- `clf2.predict(X[0:1])`
(pada baris ini clf2.predict sebagai suatu item dengan menggunakan metode predict untuk menentukan suatu nilai dari (X[0:1])) Lihat gambar 7.19



array([0])

Figure 1.16: Hasil Tampilan 7.

- `y[0]`
(pada estimator/parameter y berapapun angka yang diganti nilainya akan selalu konstan yaitu 0) Lihat gambar 7.20.



>>> y[0]
0

Figure 1.17: Hasil Tampilan 8.

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

- `dump(clf, 'filename.joblib')`
 (pada baris berikutnya dump di sini sebagai class yang didalamnya terdapat nilai dari suatu item clf dan data joblib).
- `clf = load('filename.joblib')`
 (pada baris terakhir clf sebagai estimato/parameter dengan suatu nilai load berfungsi untuk mengulang data sebelumnya)
- dari ketiga baris akhir tersebut jika di jalankan aaau dituliskan perintah seperti itu maka akan menampilkan tampilan eror terlihat pada gambar 7.21

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

Figure 1.18: Hasil Tampilan 9.

1.4.2.5 Conventions

1. Type Casting

- `from sklearn import svm`
 (pada baris ini merupakan sebuah perintah untuk mengimport class svm dari packaged sklearn).
- `from sklearn import random_projection`
 (pada baris ini merupakan sebuah perintah untuk mengimport class random_projection dari packaged sklearn).
- `rng = np.random.RandomState(0)`
 (rng sebagai estimator/parameter dengan nilai suatu itemnya yaitu np.random.RandomState(0)).
- `X = rng.rand(10, 2000)`
 (X sebagai estimator/parameter dengan nilai item rng.rand(10, 2000)).

- `X = np.array(X, dtype='float32')`
(X sebagai estimator/parameter dengan nilai item np.array).
- `X.dtype`
(X.dtype sebagai item pemanggil) Lihat gambar 7.22

```
>>> X.dtype
dtype('float32')
```

Figure 1.19: Hasil Tampilan 10.

- `transformer = random_projection.GaussianRandomProjection()`
(transformer sebagai estimator/parameter dengan memanggil class random_projection).
- `X_new = transformer.fit_transform(X)`
(X_new di sini sebagai estimator/parameter dan menggunakan metode fit)
- `X_new.dtype`
(X_new.dtype sebagai item) Lihat gambar 7.23.

```
>>> X_new.dtype
dtype('float64')
```

Figure 1.20: Hasil Tampilan 11.

- `from sklearn import datasets`
(pada baris ini merupakan sebuah perintah untuk mengimport class datasets dari package sklearn).
- `from sklearn.svm import SVC`
(pada baris ini merupakan sebuah perintah untuk mengimport class SVC dari package sklearn.svm).
- `iris = datasets.load_iris()`
(iris sebagai estimator/parameter dengan item datasets.load_iris()).

- `clf = SVC(gamma='scale')`
(clf sebagai estimator/parameter dengan nilai class SVC pada parameter gamma sebagai set penilaian).
- `clf.fit(iris.data, iris.target)`
(estimator/parameter clf menggunakan metode fit dengan itemnya) Lihat gambar 2.21

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

Figure 1.21: Hasil Tampilan 12.

- `list(clf.predict(iris.data[:3]))`
(menambahkan item list dengan metode predict) Lihat gambar 2.22.

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

Figure 1.22: Hasil Tampilan 13.

- `clf.fit(iris.data, iris.target_names[iris.target])`
(estimator/parameter clf menggunakan metode fit dengan itemnya) Lihat gambar 1.23.

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

Figure 1.23: Hasil Tampilan 14.

- `list(clf.predict(iris.data[:3]))` (menambahkan item list dengan metode predict)
Lihat gambar 1.24.

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

Figure 1.24: Hasil Tampilan 15.

2. Refitting and Updating Parameters

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

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

Figure 1.25: Hasil Tampilan 16.

??

- `clf.predict(X_test)`
(menggunakan metode predict) Lihat gambar 1.26.
- `clf.set_params(kernel='rbf', gamma='scale').fit(X, y)`
Lihat gambar 1.27.
- `clf.predict(X_test)`
Lihat gambar 1.28.

```
>>> clf.predict(X_test)
array([1, 0, 1, 1, 0])
```

Figure 1.26: Hasil Tampilan 17.

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

Figure 1.27: Hasil Tampilan 18.

```
>>> clf.predict(X_test)
array([1, 0, 1, 1, 0])
```

Figure 1.28: Hasil Tampilan 19.

3. Multiclass vs. Multilabel Fitting

- `from sklearn.svm import SVC`
(pada baris ini merupakan sebuah perintah untuk mengimport class SVC dari packaged sklearn.svm).
 - `from sklearn.multiclass import OneVsRestClassifier`
(pada baris ini merupakan sebuah perintah untuk mengimport class OneVsRestClassifier dari packaged sklearn.multiclass).
 - `from sklearn.preprocessing import LabelBinarizer`
(pada baris ini merupakan sebuah perintah untuk mengimport class LabelBinarizer dari packaged sklearn.preprocessing).
 - `X = [[1, 2], [2, 4], [4, 5], [3, 2], [3, 1]]`
 - `y = [0, 0, 1, 1, 2]`
 - `classif = OneVsRestClassifier(estimator=SVC(gamma='scale',random_state=0))`
 - `classif.fit(X, y).predict(X)`
- Lihat gambar 1.29.
- `y = LabelBinarizer().fit_transform(y)`
 - `classif.fit(X, y).predict(X)`
- Lihat gambar 1.30.

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

Figure 1.29: Hasil Tampilan 20.

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

Figure 1.30: Hasil Tampilan 21.

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

Lihat gambar 1.31.

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

Figure 1.31: Hasil Tampilan 22.

1.4.3 Penanganan eror

1.4.3.1 ScreenShoot Eror

1.4.3.2 Tuliskan Kode Eror dan Jenis Erornya

- `from joblib import dump, load`

(Kode baris pertama)

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

Figure 1.32: Hasil Tampilan Error.

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

(Errornya)

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

(Kode baris kedua)

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

(Errornya)

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

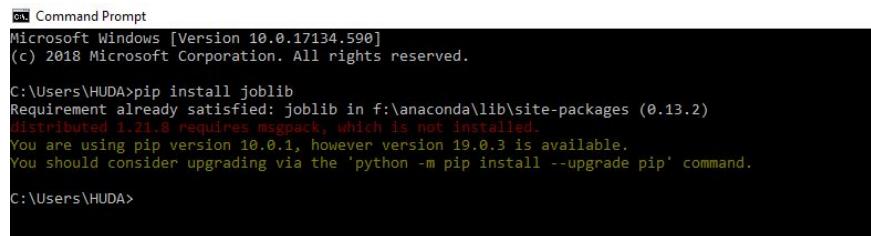
(Kode baris ketiga)

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

(Errornya)

1.4.3.3 Solusi Pemecahan Masalah Error

1. Pada masalah error sebelumnya itu dikarenakan kita belum mempunyai packaged joblib. Jadi solusinya yaitu dengan cara menginstall terlebih dahulu packaged joblibnya setelah itu baru perintah tersebut dapat dijalankan seperti pada gambar 1.33 dan 1.34

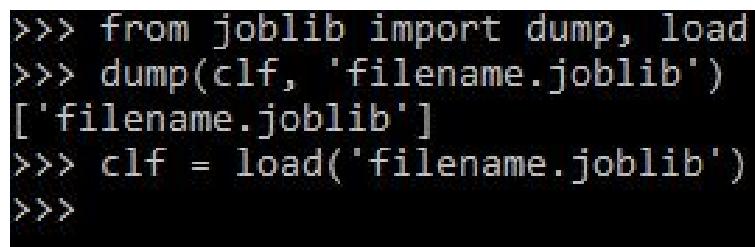


```
Command Prompt
Microsoft Windows [Version 10.0.17134.590]
(c) 2018 Microsoft Corporation. All rights reserved.

C:\Users\HUDA>pip install joblib
Requirement already satisfied: joblib in f:\anaconda\lib\site-packages (0.13.2)
distributed 1.21.8 requires msgpack, which is not installed.
You are using pip version 10.0.1, however version 19.0.3 is available.
You should consider upgrading via the 'python -m pip install --upgrade pip' command.

C:\Users\HUDA>
```

Figure 1.33: Hasil Tampilan Install joblib.



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

Figure 1.34: Hasil Tampilan Uji coba perintah joblib.

Chapter 2

Related Works

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

2.1 Ahmad Syafrizal Huda/1164062

2.1.1 Teori

1. Binary Classification yaitu katakanlah kita memiliki tugas untuk mengklasifikasi objek menjadi dua kelompok berdasarkan beberapa fitur. Sebagai contoh, katakanlah kita diberi beberapa pena dan pensil dari berbagai jenis dan merek, kita dapat dengan mudah memisahkannya menjadi dua kelas, yaitu pena dan pensil.

Contoh ilustrasi gambar bisa dilihat pada gambar 7.4.

2. 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. Dan clustering adalah proses pengelompokan entitas yang sama bersama-sama. Tujuan dari teknik pembelajaran mesin tanpa pengawasan ini adalah untuk menemukan kesamaan pada titik data dan mengelompokkan titik data yang serupa secara bersamaan[4].

Contoh ilustrasi gambar bisa dilihat pada gambar 7.5.

Contoh ilustrasi gambar bisa dilihat pada gambar 7.6.

Contoh ilustrasi gambar bisa dilihat pada gambar 7.7.

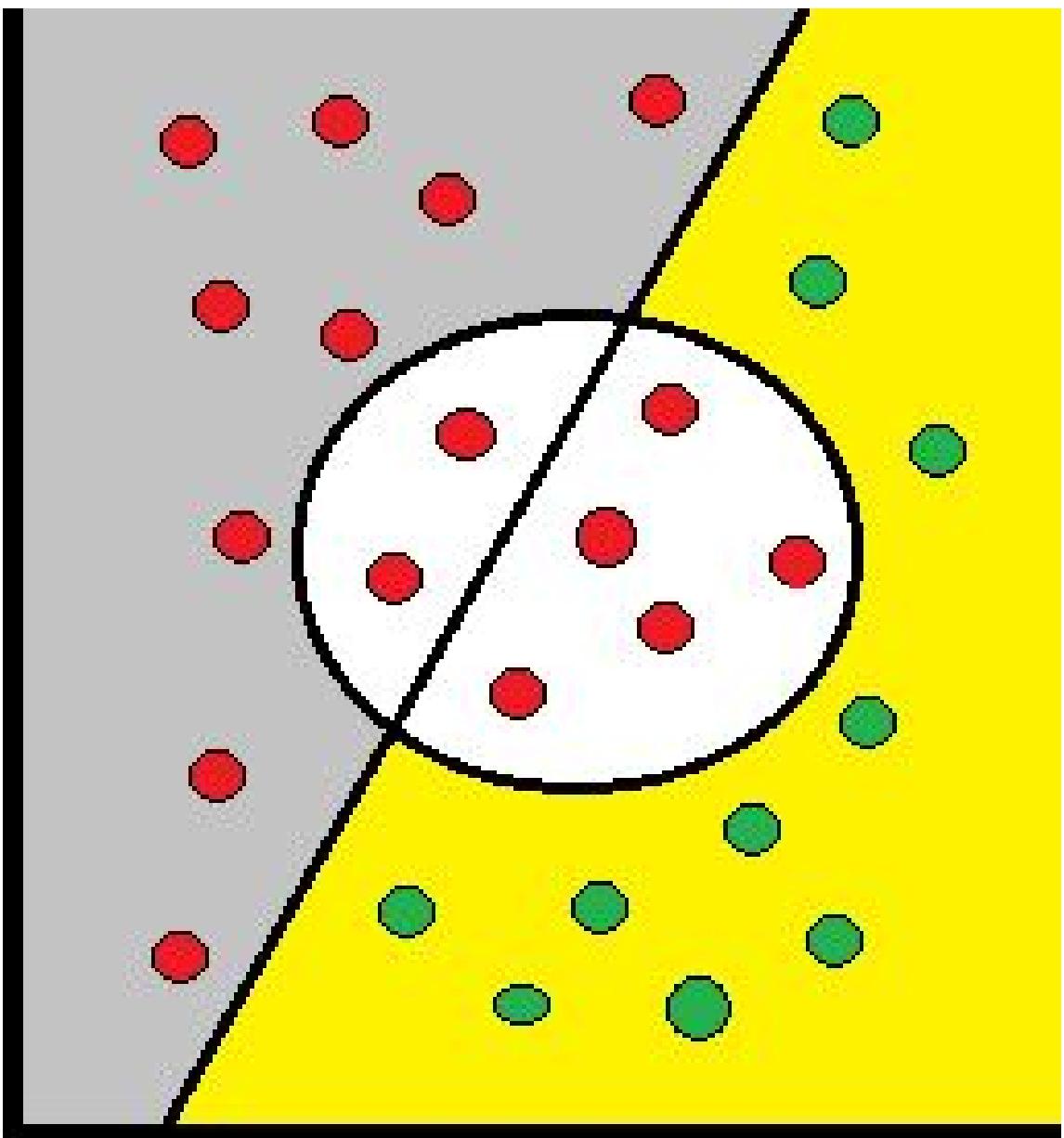


Figure 2.1: Binary Classification.

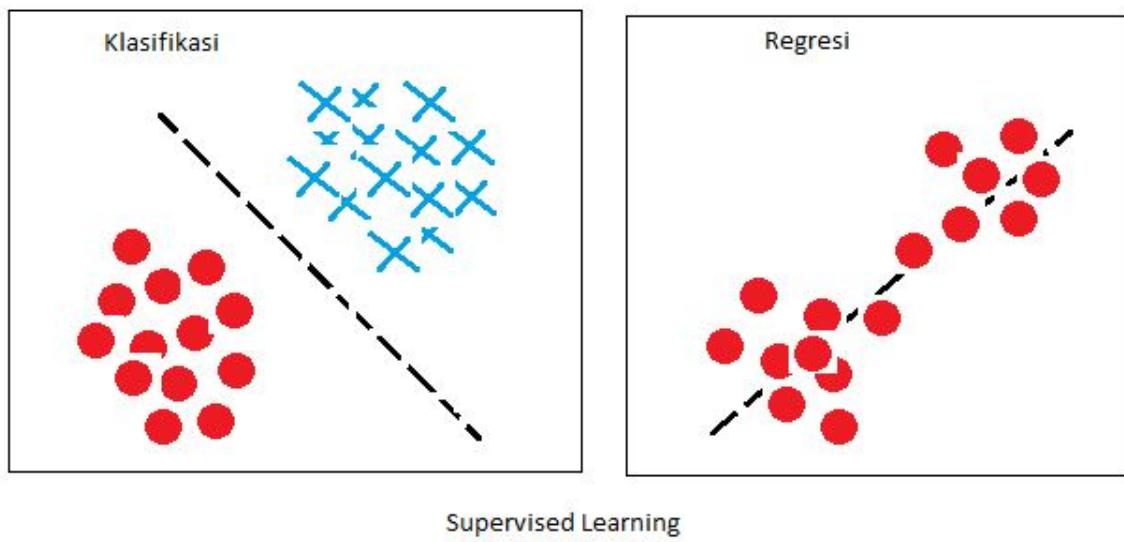


Figure 2.2: Supervised Learning.

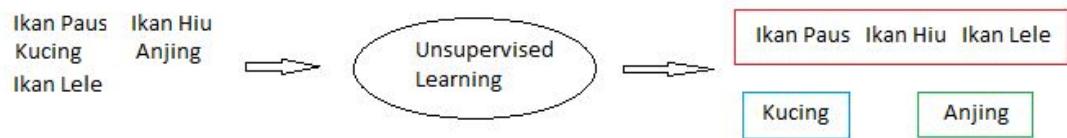


Figure 2.3: Unsupervised Learning.

Clustering

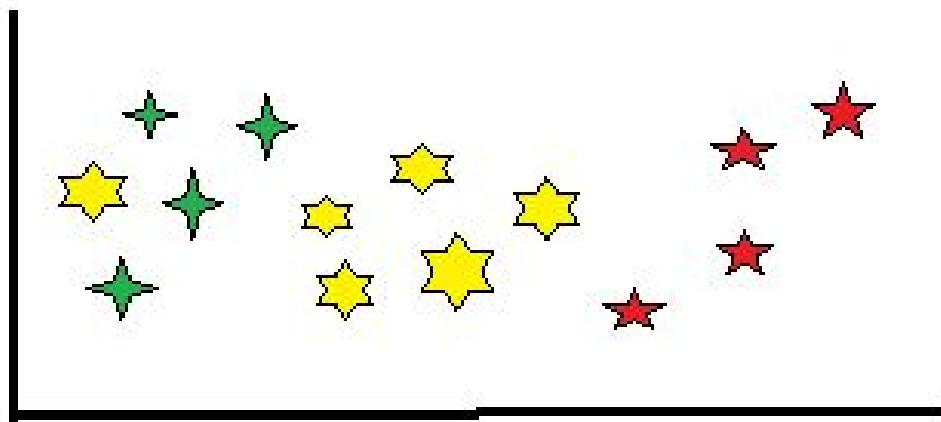


Figure 2.4: Clustering.

3. Evaluasi dan akurasi adalah bagaimana cara kita bisa mengevaluasi seberapa baik model mengerjakan pekerjaannya dengan cara mengukur akurasinya. Akurasi nantinya didefinisikan sebagai persentase kasus yang telah diklasifikasikan dengan benar. Kita dapat melakukan analisis kesalahan yang telah dibuat oleh model.

Contoh ilustrasi gambar bisa dilihat pada gambar 7.8.

		Diprediksi Pena	Diprediksi Pensil
True Pena	10	5	
True Pensil	7	8	

Figure 2.5: Evaluasi dan Akurasi.

4. Cara membuat dan membaca confusion matrix yaitu, menentukan pokok permasalahan serta atributnya, membuat Decision Tree, membuat Data Testing, mencari nilai variabelnya misal a,b,c, dan d, mencari nilai recall, precision, accuracy, dan error rate.

Contoh Confusion Matrix.

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

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

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

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

5. Berikut ini tata cara K-fold Cross Validation:

- Total instance akan dibagi menjadi N bagian.

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

Contoh ilustrasi gambar bisa dilihat pada gambar 7.9.

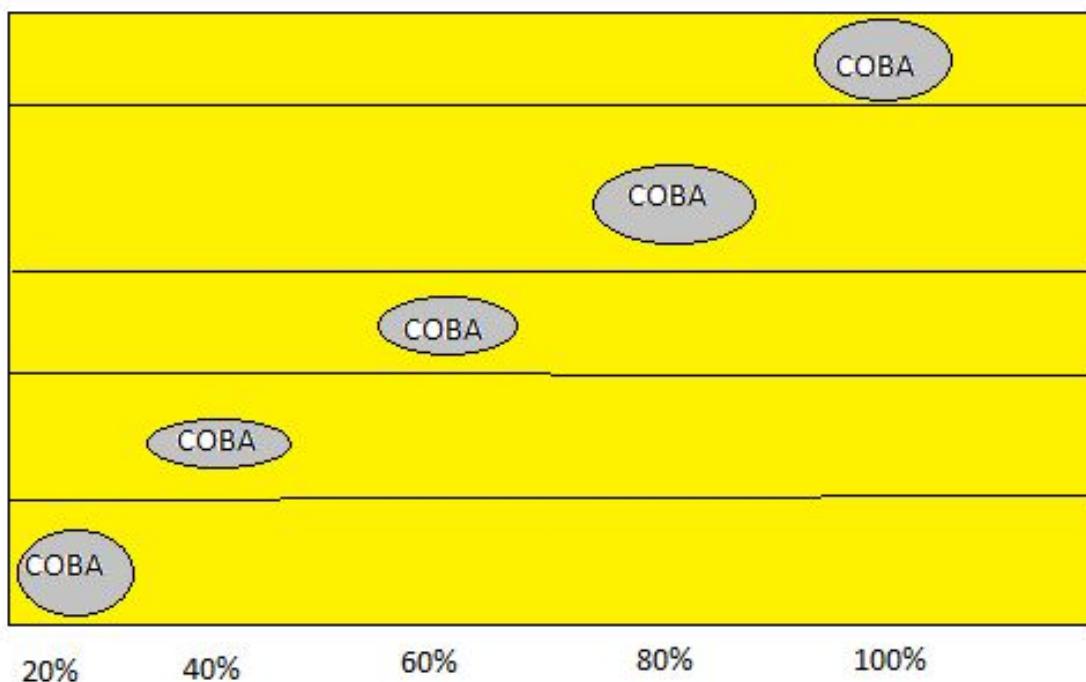


Figure 2.6: K-fold Cross Validation.

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

Contoh ilustrasi gambar bisa dilihat pada gambar 7.10.



Figure 2.7: Decision Tree.

- Gain adalah pengurangan yang diharapkan dalam entropy. Dalam machine learning, gain dapat digunakan untuk menentukan sebuah urutan atribut atau memperkecil atribut yang telah dipilih. Urutan ini akan membentuk decision tree, atribut gain dipilih yang paling besar. Dan Entropi adalah ukuran ketidakpastian sebuah variabel acak sehingga dapat diartikan entropi adalah ukuran ketidakpastian dari sebuah atribut.

Contoh ilustrasi gambar bisa dilihat pada gambar 7.11.

2.1.2 Scikit-learn

- Penjelasan Codingan ini akan menampilkan data pada file yang ditentukan. Untuk codingan ini file yang dieksekusi untuk digunakan ialah student-mat.csv. Secara jelasnya, dalam codingan dapat dilihat bahwa variabel buahpir didefinisikan untuk pembacaan csv dari "buahnaga" dimana untuk pemisahnya yaitu separation berupa ; . Setelah itu variabel buahpir di tampilkan dengan perintah menampilkan len panjang ataupun jumlah dan hasilnya berupa angka 395 .

Gambar Screenshootan codingan dan hasil bisa dilihat pada gambar 7.12.

- Penjelasan codingan ini berfungsi untuk menampilkan baris G1, G2 dan G3 (berdasarkan kriterianya) untuk kolom PASS pada variabel buahpir. Untuk lebih jelasnya, pada codingan terdapat pendefinisian pembacaan lamda (panjang gelombang) dari baris G1, G2 dan G3. Apabila row-row tersebut bernilai

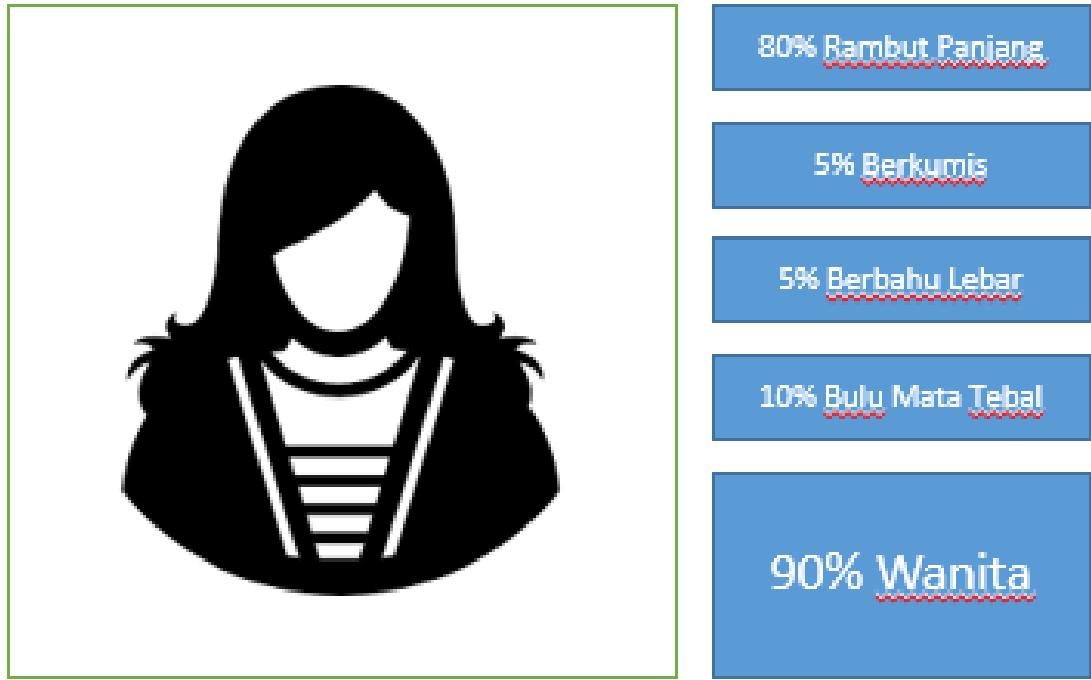


Figure 2.8: Gain.

```
In [10]: import pandas as buahnaga
...: buahpir = buahnaga.read_csv('E:\DATA KULIAH\SEMESTER 6\KECERDASAN BUATAN(PAK
ROLLY)\Python-Artificial-Intelligence-Projects-for-Beginners\Chapter01\dataset\student-
mat.csv', sep=';')
...: len(buahpir)
Out[10]: 395
```

Figure 2.9: Hasil Codingan No 1.

lebih dari 35 maka akan terdefinisikan angka 1 apabila tidak, maka akan terdefinisikan angka 0 pada kolom PASS (sesuai permintaan awal). Selanjutnya variabelnya di ditampilkan sehingga menampilkan keluaran. Tidak lupa terdapat juga jumlah dari baris dan kolom yang terubah sesuai dengan baris yang dieksekusi.

Gambar Screenshootan codingan dan hasil bisa dilihat pada gambar 7.13.

```
In [11]: buahpir['pass'] = buahpir.apply(lambda row: 1 if (row['G1']+row['G2']+row['G3']) >= 35 else 0, axis=1)
.... buahpir = buahpir.drop(['G1', 'G2', 'G3'], axis=1)
.... buahpir.head()
Out[11]:
   school sex age address famsize ... Dalc Walc health absences pass
0      GP   F   18      U    GT3 ...     1     1     3       6     0
1      GP   F   17      U    GT3 ...     1     1     3       4     0
2      GP   F   15      U    LE3 ...     2     3     3      10     0
3      GP   F   15      U    GT3 ...     1     1     5       2     1
4      GP   F   16      U    GT3 ...     1     2     5       4     0
[5 rows x 31 columns]
```

Figure 2.10: Hasil Codingan No 2.

3. Penjelasan codingan ini mendefinisikan pemanggilan get dummies dari buah-naga dalam variabel buahpir. Di dalam get dummies sendiri akan terdefinisikan variabel buahpir dengan kolom-kolom yang akan dieksekusi seperti school, address dll. Kemudian variabel tersebut diartikan untuk mendapatkan kembalian berupa keluaran dari eksekusi perintah variabel buahpir beserta dengan jumlah baris dan kolom data yang dieksekusi.

Gambar Screenshootan codingan dan hasil bisa dilihat pada gambar 7.14.

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

Gambar Screenshootan codingan dan hasil bisa dilihat pada gambar 7.15.

```

In [12]: buahpir = buahnaga.get_dummies(buahpir, columns=['sex', 'school', 'address',
...: 'famsize',
...: 'Pstatus', 'Mjob', 'Fjob',
...: 'reason', 'guardian', 'schoolsup',
...: 'famsup', 'paid', 'activities',
...: 'nursery', 'higher', 'internet',
...: 'romantic'])
...: buahpir.head()
Out[12]:
   age  Medu  Fedu    ...      internet_yes  romantic_no  romantic_yes
0   18     4     4    ...              0            1            0
1   17     1     1    ...              1            1            0
2   15     1     1    ...              1            1            0
3   15     4     2    ...              1            0            1
4   16     3     3    ...              0            1            0
[5 rows x 57 columns]

```

Figure 2.11: Hasil Codingan No 3.

```

In [5]: buahpir = buahpir.sample(frac=1)
...: # split training and testing data
...: buahpir_train = buahpir[:500]
...: buahpir_test = buahpir[500:]
...:
...: buahpir_train_att = buahpir_train.drop(['pass'], axis=1)
...: buahpir_train_pass = buahpir_train['pass']
...:
...: buahpir_test_att = buahpir_test.drop(['pass'], axis=1)
...: buahpir_test_pass = buahpir_test['pass']
...:
...: buahpir_att = buahpir.drop(['pass'], axis=1)
...: buahpir_pass = buahpir['pass']
...:
...: # number of passing students in whole dataset:
...: import numpy as buahpepaya
...: print("Passing: %d out of %d (%.2f%%)" % (buahpepaya.sum(buahpir_pass),
len(buahpir_pass), 100*float(buahpepaya.sum(buahpir_pass)) / len(buahpir_pass)))
Passing: 328 out of 649 (50.54%)

```

Figure 2.12: Hasil Codingan No 4.

5. Penjelasan codingan ini hanya membuktikan pengujian dari Klasifikasi Decision Tree yang ada, apakah true atau tidak dan hasilnya true. Pada codingan ini di definisikan library sklearn untuk mengimport atau menampilkan tree. Variabel buahapel difungsikan untuk membaca klasifikasi decision tree dari tree itu sendiri dengan 2 parameternya yaitu criterion=entropy dan max_depth=5. Maka selanjutnya variabel buahapel akan masuk dan terbaca dalam module fit dengan 2 parameter yaitu buahpir train_att dan buahpir train_pass.

Gambar Screenshootan codingan dan hasil bisa dilihat pada gambar 7.16.

```
In [6]: from sklearn import tree
....: buahapel = tree.DecisionTreeClassifier(criterion="entropy", max_depth=5)
....: buahapel = buahapel.fit(buahpir_train_att, buahpir_train_pass)
```

Figure 2.13: Hasil Codingan No 5.

6. Penjelasan codingan ini memberikan gambaran dari klasifikasi decision tree yaitu pengolahan parameter yang dieksekusi kedalam variabel buahapel. Tentunya dengan pemanfaatan library graphviz yang telah diimport dan difungsikan.

Gambar Screenshootan codingan dan hasil bisa dilihat pada gambar 7.17.

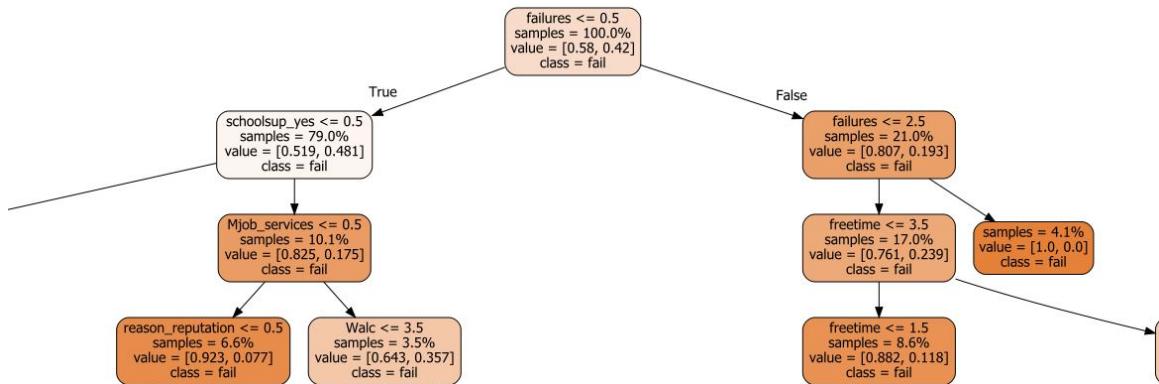


Figure 2.14: Hasil Codingan No 6.

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

Gambar Screenshootan codingan dan hasil bisa dilihat pada gambar 7.18.

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

Figure 2.15: Hasil Codingan No 7.

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

Gambar Screenshootan codingan dan hasil bisa dilihat pada gambar 7.19.

```
In [8]: buahapel.score(buahpir_test_att, buahpir_test_pass)
Out[8]: 0.6644295302013423
```

Figure 2.16: Hasil Codingan No 8.

9. Penjelasan codingan ini membahas mengenai pengkesekusian fungsi dan variabel dari library yang didefinisikan dan yang diimport. Penjelasan lebih jelasnya ialah codingan ini mendefinisikan library sklearn.model.selection kemudian mengimport cross val score. Kemudian variabel score mendefinisikan cross val score yang telah diimport tadi dengan 4 parameter yaitu buahapel, buahpir att, buahpir pass dan cv=5 untuk dieksekusi. Setelah semua pemrosesan tersebut maka hasil yang di tampilkan ialah rata2 perhitungan dari variabel score dimana dan standar dari plus minusnya tentunya dengan ketentuan parameter Accuracy .

Gambar Screenshootan codingan dan hasil bisa dilihat pada gambar 7.20.

```
In [8]: from sklearn.model_selection import cross_val_score
...: scores = cross_val_score(buahapel, buahpir_att, buahpir_pass, cv=5)
...: # show average score and +/- two standard deviations away (covering 95% of
scores)
....: print("Accuracy: %0.2f (+/- %0.2f)" % (scores.mean(), scores.std() * 2))
Accuracy: 0.68 (+/- 0.07)
```

Figure 2.17: Hasil Codingan No 9.

10. Penjelasan Codingan ini mendefinisikan max depth dalam jarak angka antara parameter 1 dan 20. Variabel buahapel mendefinisikan klasifikasi decision tree dengan 2 parameter. Kemudian variabel score mengeksekusi parameter lainnya yaitu seperti buahapel, buahpir att, buahpir pass dan cv=5 . Hasil yang ditampilkan ialah dari max depth, accuracy dan plus minusnya dan akhirnya hasil outputannya keluar.

Gambar Screenshootan codingan dan hasil bisa dilihat pada gambar 7.21.

```
In [10]: for max_depth in range(1, 20):
    .... buahapel = tree.DecisionTreeClassifier(criterion="entropy", max_depth=max_depth)
    .... scores = cross_val_score(buahapel, buahpir_att, buahpir_pass, cv=5)
    .... print("Max depth: %d, Accuracy: %.2f (+/- %.2f)" % (max_depth, scores.mean(),
scores.std() * 2))
Max depth: 1, Accuracy: 0.64 (+/- 0.02)
Max depth: 2, Accuracy: 0.69 (+/- 0.02)
Max depth: 3, Accuracy: 0.69 (+/- 0.07)
Max depth: 4, Accuracy: 0.69 (+/- 0.05)
Max depth: 5, Accuracy: 0.67 (+/- 0.03)
Max depth: 6, Accuracy: 0.66 (+/- 0.08)
Max depth: 7, Accuracy: 0.67 (+/- 0.07)
Max depth: 8, Accuracy: 0.67 (+/- 0.05)
Max depth: 9, Accuracy: 0.65 (+/- 0.04)
Max depth: 10, Accuracy: 0.66 (+/- 0.05)
Max depth: 11, Accuracy: 0.66 (+/- 0.07)
Max depth: 12, Accuracy: 0.62 (+/- 0.09)
Max depth: 13, Accuracy: 0.64 (+/- 0.08)
Max depth: 14, Accuracy: 0.63 (+/- 0.09)
Max depth: 15, Accuracy: 0.64 (+/- 0.07)
Max depth: 16, Accuracy: 0.63 (+/- 0.07)
Max depth: 17, Accuracy: 0.62 (+/- 0.09)
Max depth: 18, Accuracy: 0.63 (+/- 0.08)
Max depth: 19, Accuracy: 0.63 (+/- 0.09)
```

Figure 2.18: Hasil Codingan No 10.

11. Penjelasan codingan ini mengartikan bahwa variabel depth_acc akan mengeksekusi empty dari importan library numphy yang dinamakan buahpepaya dengan 2 parameter yaitu 19,3 dan float. i didefinisikan dengan angka 0 kemudian untuk perhitungan jarak max depth diantara parameter 1 dan 20. Variabel buahapel mengartikan klasifikasi decision tree dengan 2 parameter. setelah itu, variabel score mendefinisikan variabel depth_acc dengan i dan 0, variabel kedua dari depth_acc dengan i dan 1 serta variabel ketiga dari depth_acc dengan i dan 2, maka pengeksekusian akhir bahwa variabel i akan ditambah dengan angka 1 untuk hasil akhirnya. Keluarannya akan berupa array dari perhitungan parameter dan variabel yang telah didefinisikan sebelumnya.

Gambar Screenshootan codingan dan hasil bisa dilihat pada gambar 7.22.

```

In [12]: depth_acc = buahpepaya.empty((19,3), float)
.... i = 0
.... for max_depth in range(1, 20):
....     buahapel = tree.DecisionTreeClassifier(criterion="entropy", max_depth=max_depth)
....     scores = cross_val_score(buahapel, buahpir_att, buahpir_pass, cv=5)
....     depth_acc[i,0] = max_depth
....     depth_acc[i,1] = scores.mean()
....     depth_acc[i,2] = scores.std() * 2
....     i += 1
.....
.....
.....
.....
.....
Out[12]:
array([[ 1.        ,  0.63795172,  0.02257095],
       [ 2.        ,  0.68720762,  0.02295034],
       [ 3.        ,  0.69178704,  0.06621384],
       [ 4.        ,  0.69181089,  0.05105217],
       [ 5.        ,  0.67026014,  0.02969374],
       [ 6.        ,  0.66411731,  0.06858724],
       [ 7.        ,  0.68263885,  0.06498788],
       [ 8.        ,  0.67186961,  0.04285183],
       [ 9.        ,  0.65182173,  0.03602718],
      [10.        ,  0.65173861,  0.04048406],
      [11.        ,  0.65169145,  0.07924518],
      [12.        ,  0.63476783,  0.04265046],
      [13.        ,  0.62398685,  0.06238911],

```

Figure 2.19: Hasil Codingan No 11.

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

Gambar Screenshootan codingan dan hasil bisa dilihat pada gambar 7.23.

2.1.3 Penanganan Eror

1. ScreeShootan Eror pada codingan No 8 dapat dilihat pada gambar 2.21.
2. Codingan eror dan jenis erornya : sebenarnya tidak terdapat eror pada codingan ini namun saat pertama kali di run current cell codingan ini akan eror dan tidak keluar outputannya dikarenakan library graphviz sebelumnya tidak ditemukan atau belum di install terlebih dahulu.

```
import graphviz
```

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

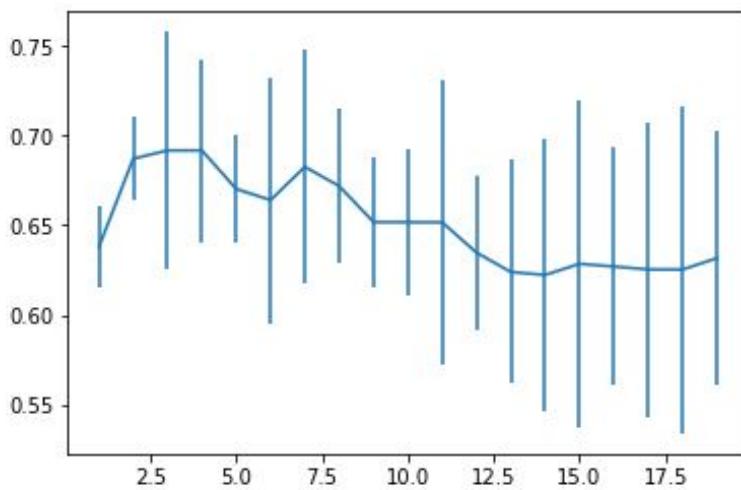


Figure 2.20: Hasil Codingan No 12.

```
In [25]: import graphviz
...: dot_data = tree.export_graphviz(buahpepaya, out_file=None, label="all",
...: impurity=False, proportion=True,
...: feature_names=list(buahpir_train_att),
...: class_names=["fail", "pass"],
...: filled=True, rounded=True)
...: graph = graphviz.Source(dot_data)
...: graph
Traceback (most recent call last):

File "F:\anaconda\lib\site-packages\IPython\core\formatters.py", line 345, in __call__
    return method()

File "F:\anaconda\lib\site-packages\graphviz\files.py", line 106, in __repr_svg__
    return self.pipe(format='svg').decode(self._encoding)

File "F:\anaconda\lib\site-packages\graphviz\files.py", line 128, in pipe
    out = backend.pipe(self._engine, format, data, renderer, formatter)

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

File "F:\anaconda\lib\site-packages\graphviz\backend.py", line 150, in run
    raise ExecutableNotFound(cmd)

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

Figure 2.21: Hasil Gambar Eror No 6.

```

dot_data = tree.export_graphviz(buahapel, out_file=None, label="all", impurity_
                                feature_names=list(buahpir_train_att), class_
                                filled=True, rounded=True)

graph = graphviz.Source(dot_data)

graph

```

3. Solusi pemecahan masalah eror tersebut yaitu dengan cara menginstall terlebih dahulu library graphviznya pada anaconda prompt atau command prompt anda dengan perintah conda install graphviz setelah itu run kembali codingan No 8 maka akan muncul outputan atau tampilan keluarannya.

Berikut gambar cara menginstall graphviz dapat dilihat pada gambar 2.22

```

C:\Users\HUDA>conda install graphviz
Collecting package metadata: done
Solving environment: done

## Package Plan ##

environment location: F:\anaconda
added / updated specs:
- graphviz

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

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

Proceed ([y]/n)?

```

Downloading and Extracting Packages
graphviz-2.38 | 37.7 MB | #####| 100%
Preparing transaction: done
Verifying transaction: done
Executing transaction: done

Figure 2.22: Hasil Gambar Penanganan Eror No 6.

Chapter 3

Methods

3.1 Ahmad Syafrizal Huda / 1164062

3.1.1 Teori

1. Random forest ialah sekumpulan classifier yang terdiri dari banyak pohon keputusan dan mengerjakan klasifikasi berdasarkan keluaran dari hasil klasifikasi setiap pohon keputusan anggota. Klasifikasi random forest dikerjakan melalui penggabungan pohon (tree) dengan melakukan training pada sampel data yang dimiliki. Contoh ilustrasi gambar random forest pada gambar 3.1.

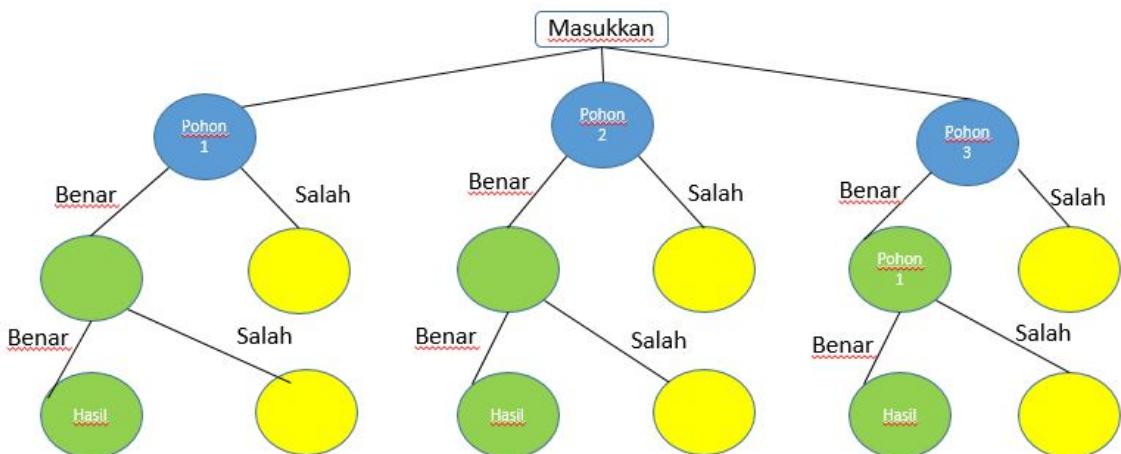


Figure 3.1: Random Forest

2. Cara membaca dataset kasus dan makna setiap file dan isi field masing-masing file

Gunakan librari pandas yang sudah di install sebelumnya pada python untuk dapat membaca dataset dengan format text file.

Selanjutnya buatlah variabel baru misalkan diberi nama dataset yang berisikan perintah untuk membaca file csv.

```
import pandas as data
dataset = data.read_csv("car.txt")
dataset.head()
```

Pada perintah diatas yaitu memanggil library pandas untuk membaca dataset, membuat variabel dataset yang berisikan data.read_csv untuk membaca dataset. Pada contoh ini menggunakan txt tapi tetap bisa membaca datasetnya, karena pada saat dijalankan librari pandas secara otomatis akan mengubah data dalam bentuk text file ke format csv. Hasilnya seperti pada gambar 3.2.

Index	buying	maint	doors	persons	lug_boot	safety	value
0	vhigh	vhigh	2	2	small	low	unacc
1	vhigh	vhigh	2	2	small	med	unacc
2	vhigh	vhigh	2	2	small	high	unacc
3	vhigh	vhigh	2	2	med	low	unacc
4	vhigh	vhigh	2	2	med	med	unacc
5	vhigh	vhigh	2	2	med	high	unacc
6	vhigh	vhigh	2	2	big	low	unacc
7	vhigh	vhigh	2	2	big	med	unacc
8	vhigh	vhigh	2	2	big	high	unacc
9	vhigh	vhigh	2	4	small	low	unacc
10	vhigh	vhigh	2	4	small	med	unacc
11	vhigh	vhigh	2	4	small	high	unacc
12	vhigh	vhigh	2	4	med	low	unacc
13	vhigh	vhigh	2	4	med	med	unacc
14	vhigh	vhigh	2	4	med	high	unacc
15	vhigh	vhigh	2	4	big	low	unacc
16	vhigh	vhigh	2	4	big	med	unacc
17	vhigh	vhigh	2	4	big	high	unacc
18	vhigh	vhigh	2	more	small	low	unacc
19	vhigh	vhigh	2	more	small	med	unacc
20	vhigh	vhigh	2	more	small	high	unacc
21	vhigh	vhigh	2	more	med	low	unacc

Figure 3.2: Hasil Dataset

Penjelasan dari isi field pada hasil gambar 3.2 yaitu: Atribut Index merupakan atribut otomatis untuk penomoran data yang sudah ada, Atribut Buying merupakan harga beli dari mobil tersebut. dengan value : v high/Sangat mahal,high/mahal,med/Cukup, low/Murah, Atribut Maint merupakan harga perawatan dari mobil tersebut, dengan value sama seperti pada atribut Buying, Atribut Doors merupakan jumlah pintu yang terdapat pada mobil, dengan value 2,3,4,5 more atau lebih dari 5, Atribut Persons merupakan kapasitas orang yang bisa masuk kedapalm mobil, dengan value 2,4, more /lebih, Atribut Lug Boot merupakan ukuran bagasi boot mobil, dengan value small,med,big, Atribut Safety merupakan perkiraan keselamatan mobil, dengan value low,med,high, Yang terakhir yaitu Value, yang dimana merupakan merupakan Class nya atau disebut dengan targetnya menyatakan apakah mobil tersebut dapat diterima atau tidak dan apakah mobil tersebut bagus atau tidak, dengan value unacc, acc, good,v good.

3. Cross validation adalah teknik validasi model untuk menilai bagaimana hasil analisis statistik (model) akan digeneralisasi ke kumpulan data independen. Ini terutama digunakan dalam pengaturan di mana tujuannya adalah prediksi, dan orang ingin memperkirakan seberapa akurat model prediksi akan dilakukan dalam praktek.
4. Arti score 44% pada random forest yaitu merupakan outputanya untuk hutan acak, arti score 27% pada decission tree adalah presentasi hasil dari perhitungan dataset acak, dan arti score 29% dari SVM adalah hasil pendekatan jaringan saraf. Hasil tersebut didapat dari hasil valdasi silang untuk memastikan bahwa membagi training test dengan cara yang berbeda. Jadi 44% untuk random forest, 27% untuk pohon keputusan, dan 29% untuk SVM. Itu merupakan presen-tase keakurasian prediksi yang dilakukan pada saat testing menggunakan label pada dataset yang digunakan. Score mendefinisikan aturan evaluasi model.
5. Perhitungan confusion Matriks dapat dilakukan sebagai berikut:

Import librari Pandas, Matplotlib, dan Numpy.

Buat variabel x_aktu yang berisikan data aktual.

Buat variabel x_diksi berisikan data yang akan dijadikan sebagai prediksi.

Buat variabel ak_confusion yang berisikan crosstab untuk membangun tabel tabulasi silang yang dapat menunjukkan frekuensi kemunculan kelompok data tertentu.

Pada variabel ak_confusion definisikan lagi nama baris yaitu Aktual dan kolomnya Prediksi

Kemudian definisikan suatu fungsi yang diberi nama plot confusion matrix yang berisikan pendefinisian confusion matrix dan juga akan di plotting. untuk code perintah lengkapnya sebagai berikut :

```
import numpy as cm1
import matplotlib.pyplot as plt
import pandas as cm2
x_aktu = cm2.Series([3, 1, 3, 3, 1, 2, 2, 3, 3, 1, 2, 3], name='Aktual')
x_diksi = cm2.Series([1, 1, 3, 2, 1, 3, 2, 1, 3, 1, 3, 3], name='Prediksi')
ak_confusion = cm2.crosstab(x_aktu, x_diksi)
ak_confusion = cm2.crosstab(x_aktu, x_diksi, rownames=['Aktual'], colnames=['Prediksi'])
def plot_confusion_matrix(df_confusion, title='Confusion matrix', cmap=plt.cm.Greens):
    plt.matshow(ak_confusion, cmap=cmap) # imshow
    plt.title(title)
    plt.colorbar()
    tick_marks = cm1.arange(len(ak_confusion.columns))
    plt.xticks(tick_marks, ak_confusion.columns, rotation=45)
    plt.yticks(tick_marks, ak_confusion.index)
    #plt.tight_layout()
    plt.ylabel(ak_confusion.index.name)
    plt.xlabel(ak_confusion.columns.name)
plot_confusion_matrix(ak_confusion)
plt.show()
```

Hasilnya akan seperti pada gambar 3.3 :

6. Voting pada random forest sebagaimana voting yaitu suara/hasil untuk setiap target yang diprediksi pada saat melakukan Random Forest. Pertimbangkan target prediksi dengan voting terbanyak/tertinggi sebagai prediksi akhir dari algoritma random forest. Contoh ilustrasi dapat dilihat pada gambar 3.4.

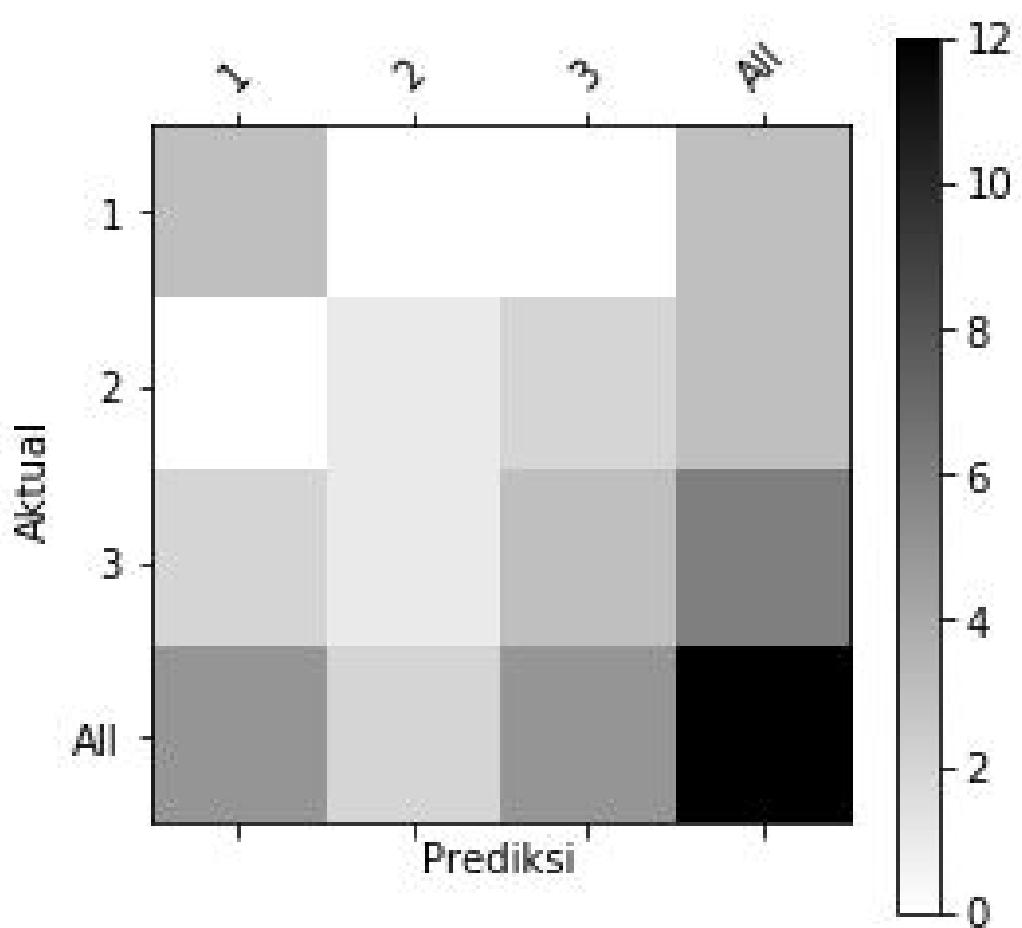


Figure 3.3: Confusion Matrix

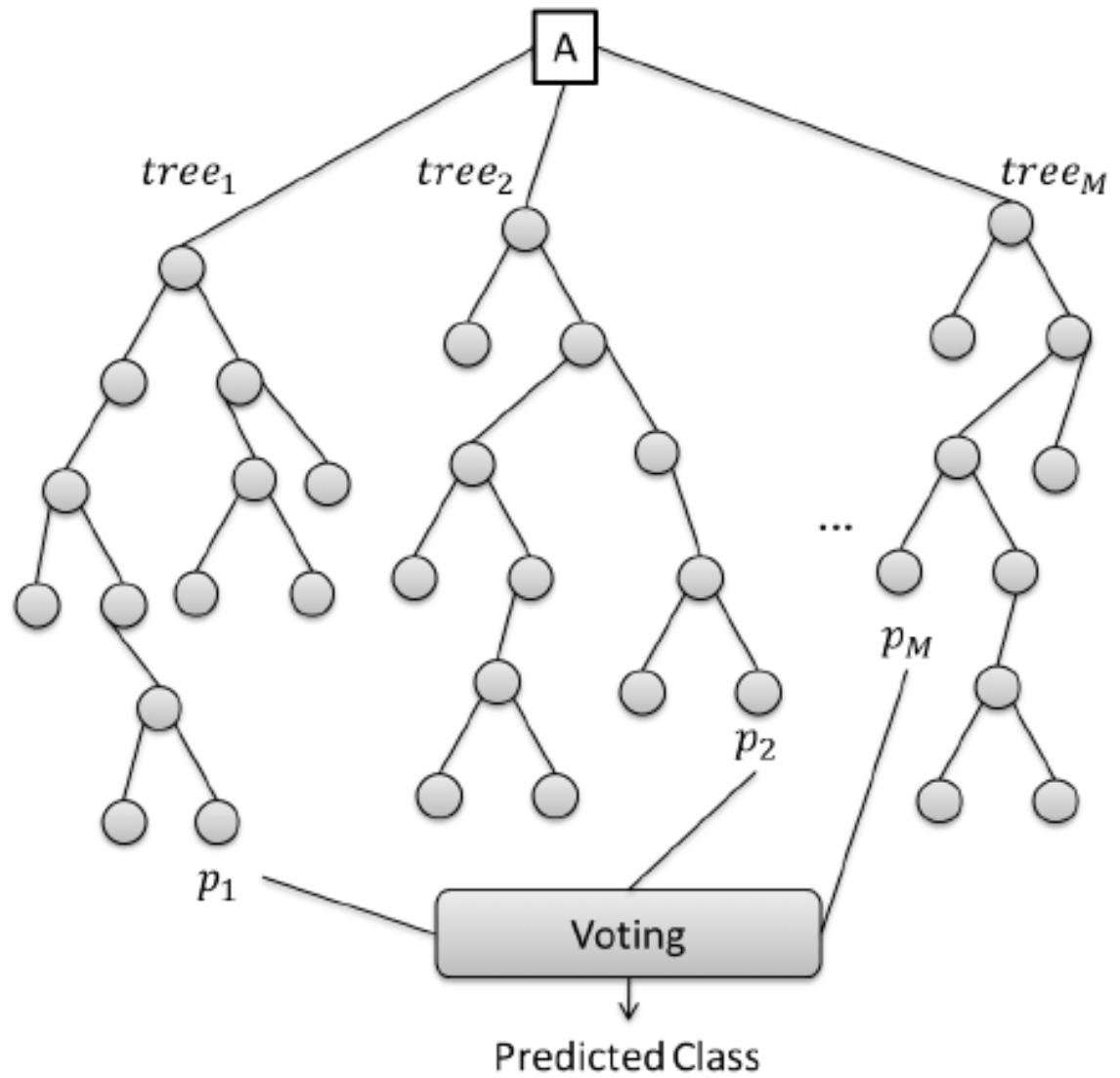


Figure 3.4: Voting

3.1.2 Praktek Program

```
1. import pandas as huda  
    a = huda.Series(['Ahmad', 'Syaf', 'Rizal', 'Huda', 'Mubarrok'],  
    index = [1, 2, 3, 4, 5])  
    print (a)
```

Baris pertama pada codingan, yaitu import pandas as huda yang artinya kita akan mengimport librari pandas dari python dengan inisiasi huda.

Baris kedua pada codingan, yaitu Variabel a didefinisikan data data yang sudah dibuat seperti daftar nama dengan menggunakan huda.Series. Series adalah object satu dimensi yang serupa dengan kolom di dalam tabel.

Baris ketiga pada codingan, yaitu index digunakan untuk melabeli data dengan dimulai dari nomor 1...5, jika tidak label default akan dimulai dari 0,1,2...

Baris keempat pada codingan, yaitu digunakan untuk mencetak atau menampilkan data pada variabel a yang sudah dibuat sebelumnya.

Untuk hasilnya dapat dilihat pada gambar 3.5.

```
In [60]: import pandas as huda  
....: a = huda.Series(['Ahmad', 'Syaf', 'Rizal', 'Huda', 'Mubarrok'],  
....: index = [1, 2, 3, 4, 5])  
....: print (a)  
1    Ahmad  
2    Syaf  
3    Rizal  
4    Huda  
5    Mubarrok  
dtype: object
```

Figure 3.5: Aplikasi Menggunakan Pandas

```
2. import numpy as huda  
    print (huda.arange(50, 101))
```

Baris pertama pada codingan, yaitu import numpy as huda yang artinya kita akan mengimport librari numpy dari python dengan inisiasi huda.

Baris kedua pada codingan, yaitu digunakan untuk mencetak atau menampilkan data dengan menggunakan huda.arange untuk membuat array dengan bilangan sekuensial dari mulai nilai awal 50 sampai sebelum 101 dengan berurutan.

Untuk hasilnya dapat dilihat pada gambar 3.6.

```
In [61]: import numpy as huda
...: print (huda.arange(50, 101))
[ 50  51  52  53  54  55  56  57  58  59  60  61  62  63  64  65  66  67
 68  69  70  71  72  73  74  75  76  77  78  79  80  81  82  83  84  85
 86  87  88  89  90  91  92  93  94  95  96  97  98  99 100]
```

Figure 3.6: Aplikasi Menggunakan Numpy

```
3. import matplotlib.pyplot as huda
huda.plot([0,1,3,5,7,8,10],[25, 35, 30, 45, 50, 45, 30])
huda.show()
```

Baris pertama pada codingan, yaitu import matplotlib.pyplot as huda yang artinya kita akan mengimport librari matplotlib dengan class pyplot dari python dengan inisiasi huda.

Baris kedua pada codingan, yaitu digunakan untuk memasukkan data dengan insiasi huda pada class plot dengan nilai x dan y yg sudah ditentukan.

Baris ketiga pada codingan, yaitu digunakan untuk mencetak atau menampilkan data dengan inisiasi huda.show nantinya keluarannya berupa grafik.

Untuk hasilnya dapat dilihat pada gambar 3.7.

4. Menjalankan Klasifikasi Random Forest.

Pada gambar 3.8 berfungsi untuk membaca data yang berupa image_attribute_labels dengan format text file. Dengan mendefinisikan variabel imgatt yang berisikan value untuk membaca data, juga menggunakan code untuk skip data yang mengandung bad lines agar tidak terjadi eror pada saat pembacaan file.

Pada gambar 3.9 yaitu mengembalikan baris n teratas (5 secara default) dari dataframe imgatt.

Pada gambar 3.10 yaitu menampilkan beberapa baris dan kolom dari dataframe imgatt.

Pada gambar 3.11 yaitu variabel imgatt2 menggunakan function pivot untuk mengubah kolom jadi baris, dan baris jadi kolom dari dataframe sebelumnya.

Pada gambar 3.12 yaitu imgatt2 head berfungsi untuk mengembalikan nilai teratas dari dataframe imgatt2.

```
In [62]: import matplotlib.pyplot as huda  
...: huda.plot([0,1,3,5,7,8,10],[25, 35, 30, 45, 50, 45, 30])  
...: huda.show()
```

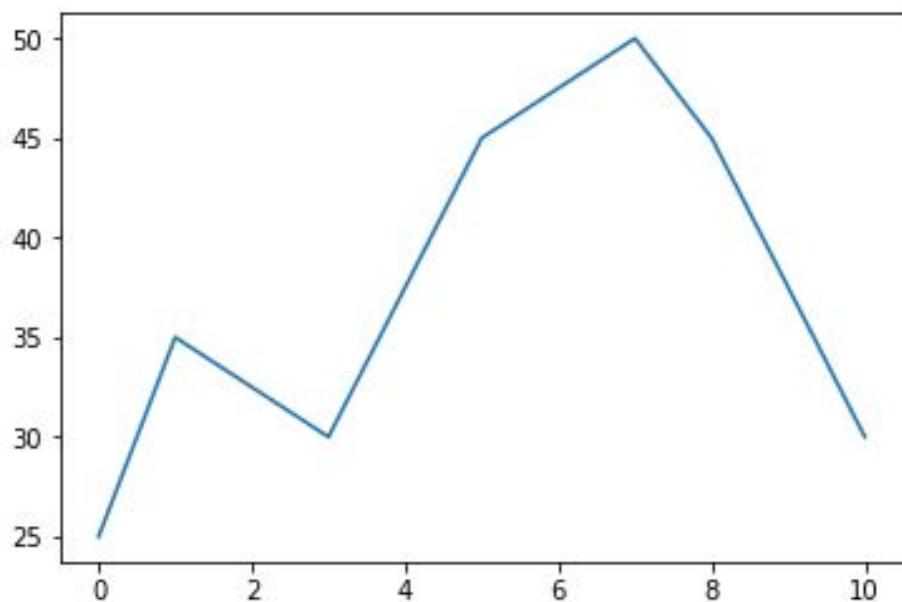


Figure 3.7: Aplikasi Menggunakan Matplotlib

Name	Type	Size	Value
imgatt	DataFrame	(3677856, 3)	Column names: imgid, attid, present

Figure 3.8: Hasil 1 Random Forest

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

Figure 3.9: Hasil 2 Random Forest

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

Figure 3.10: Hasil 3 Random Forest

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

Figure 3.11: Hasil 4 Random Forest

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

Figure 3.12: Hasil 5 Random Forest

Pada gambar 3.13 yaitu menampilkan beberapa baris dan kolom dari dataframe imgatt2.

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

Figure 3.13: Hasil 6 Random Forest

Pada gambar 3.14 yaitu melakukan pivot yang mana imgid menjadi index yang artinya unik.

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

Figure 3.14: Hasil 7 Random Forest

Pada gambar 3.15 yaitu memuat jawabannya yang berisi apakah burung itu termasuk dalam spesies yang mana. Dua kolomnya terdiri dari imgid dan label.

Pada gambar 3.16 yaitu menunjukkan 11788 baris dan 1 kolom. Dimana kolom itu adalah jenis spesies burungnya.

```
In [9]: imglabels.head()
Out[9]:
      label
imgid
3      27.708
3      27.708
3      27.708
3      27.708
3      27.708
```

Figure 3.15: Hasil 8 Random Forest

```
In [10]: imglabels.shape
Out[10]: (3677856, 1)
```

Figure 3.16: Hasil 9 Random Forest

Pada gambar 3.17 yaitu join antara imgatt2 dengan imglabels dikarenakan isinya sama. Sehingga kita bisa mendapatkan data ciri dan data jawabannya/labelnya sehingga bisa dikategorikan supervised learning.

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

Figure 3.17: Hasil 10 Random Forest

Pada gambar 3.18 yaitu menghilangkan label yang didepan, dan menggunakan label yang paling belakang yang baru di join.

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

Figure 3.18: Hasil 11 Random Forest

Pada gambar 3.19 yaitu mengecek 5 data teratas dari df att.

```
In [13]: df_att.head()
Out[13]:
   1    2    3    4    5    6    7    ...    306   307   308   309   310   311   312
imgid
11436  0    0    0    0    0    0    1    ...    0    0    0    1    0    0    0
9629   0    0    0    0    0    0    1    ...    0    0    0    1    0    0    0
1226   0    0    0    0    0    0    1    ...    0    0    1    0    0    1    0
3128   0    0    0    0    0    0    0    ...    0    0    0    0    0    0    1
5720   0    0    0    0    0    0    1    ...    0    0    0    0    0    1    0
```

[5 rows x 312 columns]

Figure 3.19: Hasil 11 Random Forest

```
In [14]: df_label.head()
Out[14]:
      label
imgid
11436    195
9629     164
1226      22
3128      54
5720      98
```

Figure 3.20: Hasil 12 Random Forest

Pada gambar 3.20 yaitu mengecek 5 data teratas dari df label.

Pada gambar 3.21 yaitu 8000 row pertama sebagai data training sisanya sebagai data testing dengan membagi menjadi dua bagian.

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

Figure 3.21: Hasil 13 Random Forest

Pada gambar 3.22 yaitu memanggil kelas RandomForestClassifier. max features diartikan sebagai berapa banyak kolom pada setiap tree dengan isi maksimum 50.

```
In [15]: from sklearn.ensemble import RandomForestClassifier
...: clf = RandomForestClassifier(max_features=50, random_state=0,
n_estimators=100)
```

Figure 3.22: Hasil 14 Random Forest

Pada gambar 3.23 yaitu melakukan fit untuk membangun random forest yang sudah ditentukan dengan maksimum fitur sebanya 50 untuk perpohonnya.

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

Figure 3.23: Hasil 15 Random Forest

Pada gambar 3.24 yaitu menampilkan hasil prediksi dari random forest sebelumnya.

```
In [17]: print(clf.predict(df_train_att.head()))
[199  87 132 119 150]
```

Figure 3.24: Hasil 16 Random Forest

Pada gambar 3.25 yaitu menampilkan besaran akurasinya dari prediksi diatas atau score perolehan dari klasifikasi.

```
In [18]: clf.score(df_test_att, df_test_label)
Out[18]: 0.4429778247096093
```

Figure 3.25: Hasil 17 Random Forest

5. Menjalankan Confusion Matrix.

Pada gambar 3.26 yaitu menyatukan Random Forest ke dalam Confusion Matrix

pred_labels	int64	(3788,)	[22 44 83 ... 143 170 98]
-------------	-------	---------	----------------------------

Figure 3.26: Hasil 1 Confusion Matrix

Pada gambar 3.27 yaitu menampilkan hasil dari gambar sebekumnya dengan array pada perintah cm.

Pada gambar 3.28 yaitu Plotting Confusion Matrix dengan librari Matplotlib.

Pada gambar 3.29 yaitu Set plot sumbunya sesuai dengan nama datanya dan membaca file classes.txt.

Pada gambar 3.30 yaitu Plot hasil perubahan label yang sudah dilakukan.

6. Menjalankan Klasifikasi SVM dan Decision Tree.

Pada gambar 3.31 yaitu klasifikasi dengan decission tree dengan dataset yang sama dan akan muncul akurasi prediksinya.

Pada gambar 3.32 yaitu klasifikasi dengan SVM dengan dataset yang sama dan akan muncul akurasi prediksinya.

```
In [20]: cm
Out[20]:
array([[ 7,  0,  0, ...,  0,  0,  0],
       [ 0, 12,  0, ...,  0,  0,  0],
       [ 2,  0, 10, ...,  0,  0,  0],
       ...,
       [ 0,  0,  0, ...,  3,  0,  0],
       [ 0,  0,  0, ...,  0,  7,  0],
       [ 0,  0,  0, ...,  0,  0, 11]], dtype=int64)
```

Figure 3.27: Hasil 2 Confusion Matrix

```
In [22]: import matplotlib.pyplot as plt
...: import itertools
...: def plot_confusion_matrix(cm, classes,
...:                         normalize=False,
...:                         title='Confusion matrix',
...:                         cmap=plt.cm.Blues):
...:
...:     """
...:     This function prints and plots the confusion matrix.
...:     Normalization can be applied by setting `normalize=True`.
...:
...:     if normalize:
...:         cm = cm.astype('float') / cm.sum(axis=1)[:, np.newaxis]
...:         print("Normalized confusion matrix")
...:     else:
...:         print('Confusion matrix, without normalization')
...:
...:     print(cm)
...:
...:     plt.imshow(cm, interpolation='nearest', cmap=cmap)
...:     plt.title(title)
...:     #plt.colorbar()
...:     tick_marks = np.arange(len(classes))
...:     plt.xticks(tick_marks, classes, rotation=90)
...:     plt.yticks(tick_marks, classes)
...:
...:     fmt = '.2f' if normalize else 'd'
...:     thresh = cm.max() / 2.
```

Figure 3.28: Hasil 3 Confusion Matrix

```
174          175.Pine_Warbler
175          176.Prairie_Warbler
176          177.Prothonotary_Warbler
177          178.Swainson_Warbler
178          179.Tennessee_Warbler
179          180.Wilson_Warbler
180          181.Worm_eating_Warbler
181          182.Yellow_Warbler
182          183.Northern_Waterthrush
183          184.Louisiana_Waterthrush
184          185.Bohemian_Waxwing
185          186.Cedar_Waxwing
186          187.American_Three_toed_Woodpecker
187          188.Pileated_Woodpecker
188          189.Red_bellied_Woodpecker
189          190.Red_cockaded_Woodpecker
190          191.Red_headed_Woodpecker
191          192.Downy_Woodpecker
192          193.Bewick_Wren
193          194.Cactus_Wren
194          195.Carolina_Wren
195          196.House_Wren
196          197.Marsh_Wren
197          198.Rock_Wren
198          199.Winter_Wren
199          200.Common_Yellowthroat
Name: birdname, Length: 200, dtype: object
```

Figure 3.29: Hasil 4 Confusion Matrix

```

[[0.41 0. 0. ... 0. 0. 0.]
 [0. 0.48 0. ... 0. 0. 0.]
 [0.11 0. 0.53 ... 0. 0. 0.]
 ...
 [0. 0. 0. ... 0.2 0. 0.]
 [0. 0. 0. ... 0. 0.35 0.]
 [0. 0. 0. ... 0. 0. 0.79]]
```

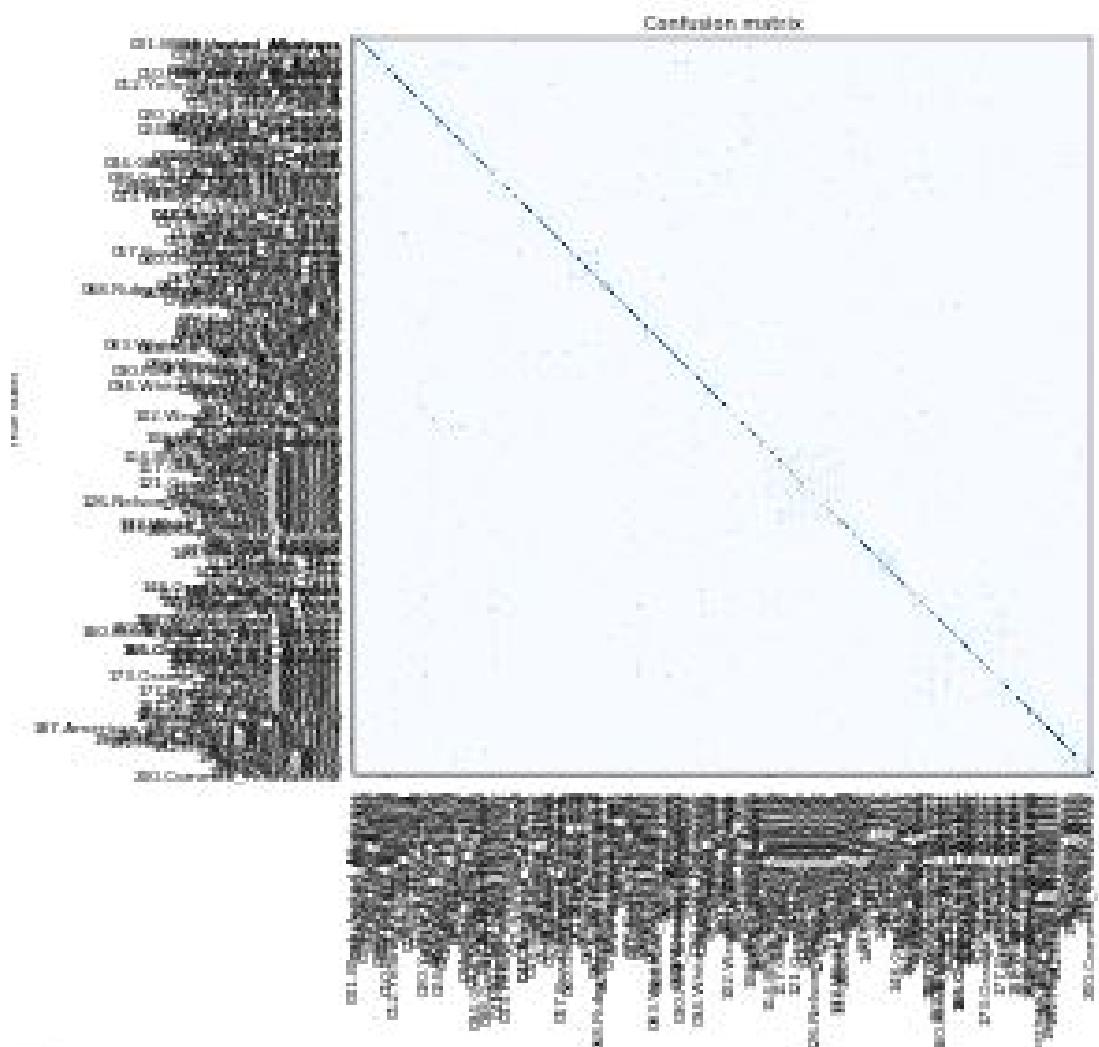


Figure 3.30: Hasil 5 Confusion Matrix

```
In [30]: from sklearn import tree
....: clftree = tree.DecisionTreeClassifier()
....: clftree.fit(df_train_att, df_train_label)
....: clftree.score(df_test_att, df_test_label)
Out[30]: 0.28299894403379094
```

Figure 3.31: Hasil 1 Klasifikasi SVM dan Decision Tree

7. Menjalankan Cross Validation.

```
In [26]: from sklearn import svm
....: clfsvm = svm.SVC()
....: clfsvm.fit(df_train_att, df_train_label)
....: clfsvm.score(df_test_att, df_test_label)
F:\anaconda\lib\site-packages\sklearn\svm\base.py:196: FutureWarning: The default
value of gamma will change from 'auto' to 'scale' in version 0.22 to account better
for unscaled features. Set gamma explicitly to 'auto' or 'scale' to avoid this
warning.
  "avoid this warning.", FutureWarning)
Out[26]: 0.2808870116156283
```

Figure 3.32: Hasil 2 Klasifikasi SVM dan Decision Tree

Pada gambar 3.33 yaitu Cross Validation untuk Random Forest.

```
In [26]: from sklearn.model_selection import cross_val_score
....: scores = cross_val_score(clf, df_train_att, df_train_label, cv=5)
....: # show average score and +/- two standard deviations away (covering 95% of
scores)
....: print("Accuracy: %.2f (+/- %.2f)" % (scores.mean(), scores.std() * 2))
F:\anaconda\lib\site-packages\sklearn\model_selection\_split.py:652: Warning: The
least populated class in y has only 1 members, which is too few. The minimum number
of members in any class cannot be less than n_splits=5.
  % (min_groups, self.n_splits)), Warning)
Accuracy: 0.30 (+/- 0.05)
```

Figure 3.33: Hasil 1 Cross Validation

Pada gambar 3.34 yaitu Cross Validation untuk Decission Tree.

Pada gambar 3.35 yaitu Cross Validation untuk SVM.

8. Menjalankan Pengamatan Komponen Informasi.

Pada gambar 3.36 yaitu untuk mengetahui berapa banyak tree yang dibuat, berapa banyak atribut yang dipakai dan informasi lainnya.

```
In [27]: scorestree = cross_val_score(clftree, df_train_att, df_train_label, cv=5)
...: print("Accuracy: %0.2f (+/- %0.2f)" % (scorestree.mean(), scorestree.std() * 2))
F:\anaconda\lib\site-packages\sklearn\model_selection\_split.py:652: Warning: The
least populated class in y has only 1 members, which is too few. The minimum number
of members in any class cannot be less than n_splits=5.
% (min_groups, self.n_splits)), Warning)
Accuracy: 0.15 (+/- 0.04)
```

Figure 3.34: Hasil 2 Cross Validation

```
In [28]: scoressvm = cross_val_score(clfsvm, df_train_att, df_train_label, cv=5)
...: print("Accuracy: %0.2f (+/- %0.2f)" % (scoressvm.mean(), scoressvm.std() * 2))
F:\anaconda\lib\site-packages\sklearn\model_selection\_split.py:652: Warning: The least populated class in y has only 1 members, which is too few. The minimum number
of members in any class cannot be less than n_splits=5.
% (min_groups, self.n_splits)), Warning)
F:\anaconda\lib\site-packages\sklearn\svm\base.py:196: FutureWarning: The default value of gamma will change from 'auto' to 'scale' in version 0.22 to account better
for unscaled features. Set gamma explicitly to 'auto' or 'scale' to avoid this warning.
"avoid this warning.", FutureWarning)
F:\anaconda\lib\site-packages\sklearn\svm\base.py:196: FutureWarning: The default value of gamma will change from 'auto' to 'scale' in version 0.22 to account better
for unscaled features. Set gamma explicitly to 'auto' or 'scale' to avoid this warning.
"avoid this warning.", FutureWarning)
F:\anaconda\lib\site-packages\sklearn\svm\base.py:196: FutureWarning: The default value of gamma will change from 'auto' to 'scale' in version 0.22 to account better
for unscaled features. Set gamma explicitly to 'auto' or 'scale' to avoid this warning.
"avoid this warning.", FutureWarning)
F:\anaconda\lib\site-packages\sklearn\svm\base.py:196: FutureWarning: The default value of gamma will change from 'auto' to 'scale' in version 0.22 to account better
for unscaled features. Set gamma explicitly to 'auto' or 'scale' to avoid this warning.
"avoid this warning.", FutureWarning)
F:\anaconda\lib\site-packages\sklearn\svm\base.py:196: FutureWarning: The default value of gamma will change from 'auto' to 'scale' in version 0.22 to account better
for unscaled features. Set gamma explicitly to 'auto' or 'scale' to avoid this warning.
"avoid this warning.", FutureWarning)
Accuracy: 0.01 (+/- 0.00)
```

Figure 3.35: Hasil 3 Cross Validation

```
In [29]: max_features_opts = range(1, 10, 1)
...: n_estimators_opts = range(2, 40, 4)
...: rf_params = np.empty((len(max_features_opts)*len(n_estimators_opts),4), float)
...: i = 0
...: for max_features in max_features_opts:
...:     for n_estimators in n_estimators_opts:
...:         clf = RandomForestClassifier(max_features=max_features, n_estimators=n_estimators)
...:         scores = cross_val_score(clf, df_train_att, df_train_label, cv=5)
...:         rf_params[i,0] = max_features
...:         rf_params[i,1] = n_estimators
...:         rf_params[i,2] = scores.mean()
...:         rf_params[i,3] = scores.std() * 2
...:         i += 1
...: print("Max features: %d, num estimators: %d, accuracy: %0.2f (+/- %0.2f)" % (max_features, n_estimators, scores.mean(), scores.std() * 2))
F:\anaconda\lib\site-packages\sklearn\model_selection\_split.py:652: Warning: The least populated class in y has only 1 members, which is too few. The minimum number
of members in any class cannot be less than n_splits=5.
% (min_groups, self.n_splits)), Warning)
F:\anaconda\lib\site-packages\sklearn\model_selection\_split.py:652: Warning: The least populated class in y has only 1 members, which is too few. The minimum number
of members in any class cannot be less than n_splits=5.
% (min_groups, self.n_splits)), Warning)
Max features: 1, num estimators: 2, accuracy: 0.06 (+/- 0.04)
Max features: 1, num estimators: 6, accuracy: 0.08 (+/- 0.03)
F:\anaconda\lib\site-packages\sklearn\model_selection\_split.py:652: Warning: The least populated class in y has only 1 members, which is too few. The minimum number
of members in any class cannot be less than n_splits=5.
% (min_groups, self.n_splits)), Warning)
Max features: 1, num estimators: 10, accuracy: 0.12 (+/- 0.03)
F:\anaconda\lib\site-packages\sklearn\model_selection\_split.py:652: Warning: The least populated class in y has only 1 members, which is too few. The minimum number
of members in any class cannot be less than n_splits=5.
% (min_groups, self.n_splits)), Warning)
```

Figure 3.36: Hasil 1 Pengamatan Komponen Informasi

Pada gambar 3.37 yaitu hasil dari plotting komponen informasi agar dapat dibaca.

```
....: ax = fig.gca(projection='3d')
....: x = rf_params[:,0]
....: y = rf_params[:,1]
....: z = rf_params[:,2]
....: ax.scatter(x, y, z)
....: ax.set_zlim(0.02, 0.3)
....: ax.set_xlabel('Max features')
....: ax.set_ylabel('Num estimators')
....: ax.set_zlabel('Avg accuracy')
....: plt.show()
```

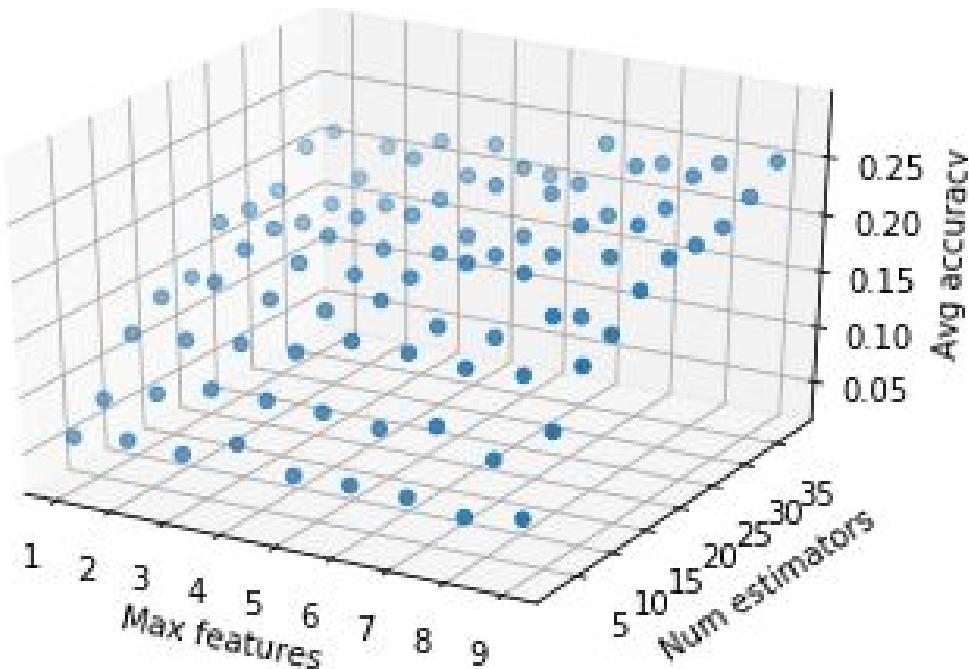


Figure 3.37: Hasil 2 Pengamatan Komponen Informasi

3.1.3 Penanganan Eror

1. Screenshootan eror ada pada gambar 3.38.
2. `from sklearn.model_selection import cross_val_score`

MemoryError

Figure 3.38: Eror

```
scores = cross_val_score(clf, df_train_att, df_train_label, cv=5)
print("Accuracy: %0.2f (+/- %0.2f)" % (scores.mean(), scores.std() * 2))

scorestree = cross_val_score(clftree, df_train_att, df_train_label, cv=5)
print("Accuracy: %0.2f (+/- %0.2f)" % (scorestree.mean(), scorestree.std() * 2))

scoressvm = cross_val_score(clfsvm, df_train_att, df_train_label, cv=5)
print("Accuracy: %0.2f (+/- %0.2f)" % (scoressvm.mean(), scoressvm.std() * 2))

max_features_opts = range(1, 10, 1)
n_estimators_opts = range(2, 40, 4)
rf_params = np.empty((len(max_features_opts)*len(n_estimators_opts),4), float)
i = 0
for max_features in max_features_opts:
    for n_estimators in n_estimators_opts:
        clf = RandomForestClassifier(max_features=max_features, n_estimators=n_estimators)
        scores = cross_val_score(clf, df_train_att, df_train_label, cv=5)
        rf_params[i,0] = max_features
        rf_params[i,1] = n_estimators
        rf_params[i,2] = scores.mean()
        rf_params[i,3] = scores.std() * 2
        i += 1
print("Max features: %d, num estimators: %d, accuracy: %0.2f (+/- %0.2f)" % (rf_params[:,0].mean(), rf_params[:,1].mean(), rf_params[:,2].mean(), rf_params[:,3].mean()))
```

3. Solusi pemecahan masalah tersebut yaitu dengan cara mengubah pada codingan dibawah ini yang awalnya datanya 8000 menjadi 1000 semua dan max_featurenya awalnya 50 menjadi 25.

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

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

from sklearn.ensemble import RandomForestClassifier
clf = RandomForestClassifier(max_features=25, random_state=0, n_estimators=10)
```

Jika sudah dilakukan maka langkah selanjutnya yaitu kita tinggal merunning ulang dari awal hingga data tersebut berhasil diajalankan

Chapter 4

Experiment and Result

brief of experiment and result.

4.1 Experiment

Please tell how the experiment conducted from method.

4.2 Result

Please provide the result of experiment

4.3 Ahmad Syafrizal Huda/1164062

4.3.1 Teori

1. Klasifikasi teks adalah proses pemberian kategori ke dalam teks/dokumen sesuai dengan tipikal dalam supervised machine

learning (ML) yang bisa berupa buku perpustakaan, halaman web, artikel media, galeri, dan lain sebagainya. Tujuannya

untuk memberikan label pada setiap teks/dokumen.

Contoh ilustrasi gambar dapat dilihat pada gambar 4.1

2. Mengapa klasifikasi bunga tidak dapat menggunakan machine learning? itu dikarenakan memiliki masalah masukan(input)

yang sama tetapi keluarannya (output) yang berbeda, biasanya output yang berbeda ini disebut dengan istilah 'noise'.



Figure 4.1: Klasifikasi Teks

Noise berarti output yang disimpan bukan seperti seharusnya (keluaran yang tidak diinginkan).

Contoh ilustrasi gambar dapat dilihat pada gambar 4.2

	Sepal.Length	Sepal.Width	Petal.Length	Petal.Width	Species
1	5.1	3.5	1.4	0.2	setosa
2	4.9	3.0	1.4	0.2	setosa
3	4.7	3.2	1.3	0.2	setosa
4	4.6	3.1	1.5	0.2	setosa
5	5.0	3.6	1.4	0.2	setosa

Figure 4.2: Klasifikasi Bunga

3. Teknik pembelajaran mesin pada teks biasanya menggunakan teknik bag-of-words pada klasifikasi berbasis text dan kata untuk mengklasifikasikan komentar yang ada diinternet sebagai spam atau bukan. Misalkan pada kolom komentar di youtube dapat di cek seberapa sering suatu kata muncul dalam kalimat. Setiap kata dapat dijadikan baris dan kolom yang

merupakan kategori kata tersebut, apakah masuk kedalam spam atau tidak. dan contoh lainnya yaitu pada Caption. dimana akan muncul subtitle secara otomatis dari youtube menggunakan sensor suara yang disesuaikan dengan kata yang telah ditentukan.

Contoh ilustrasi gambar dapat dilihat pada gambar 4.3

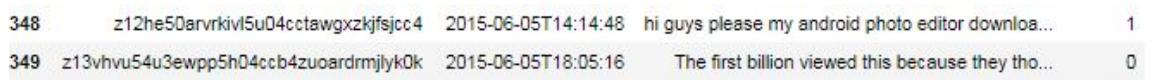


Figure 4.3: Comment Youtube

4. Vektorisasi data merupakan pembagian dan pemecahan data yang kemudian nantinya data tersebut akan menjadi beberapa data

Contoh misalkan dari sebuah paragraph nantinya kan di pecah menjadi beberapa kalimat dari kalimat tersebut dibagi lagi menjadi beberapa kata.

5. Bag-of-words adalah cara untuk merepresentasikan data teks saat memodelkan teks yang menggambarkan kemunculan

kata-kata dalam dokumen dengan algoritma pembelajaran mesin.

Contoh ilustrasi gambar dapat dilihat pada gambar 4.4

6. TF-IDF memberi kita frekuensi kata dalam setiap dokumen atau mengganti data jadi number. Ini adalah rasio berapa

kali kata itu muncul dalam dokumen dibandingkan dengan jumlah total kata dalam dokumen itu. Itu meningkat seiring jumlah kemunculan kata itu di dalam dokumen meningkat. Setiap dokumen memiliki tf sendiri.

Contoh ilustrasi gambar dapat dilihat pada gambar 4.5



Figure 4.4: Bag Of Words

Kalimat 1		Kalimat 2	
Term	TF	Term	IDF
	Kalimat 1	Kalimat 2	df
Pagi	1	1	2
nanti	1	1	2
jangan	1	1	2
lupa	1	0	1
makan	1	0	1
pergi	0	1	1
dulu	0	1	1

Figure 4.5: TF-IDF

Name	Type	Size	Value
huda	DataFrame	(500, 7)	Column names: User Id, Sports, Religious, Nature, Theatre, Shopping, P ...

Variable explorer File explorer Help

IPython console

Console 1/A

```
In [1]: import pandas as praktek
...: huda=praktek.read_csv("E:\DATA KULIAH\SEMESTER 6\KECERDASAN BUATAN(PAK ROLLY)\Holiday.csv")
```

Figure 4.6: Data Dummy 500 Data

4.3.2 Praktek Program

1. Kali ini kita akan membuat data dummy dengan format csv di sini saya mengambil data dari web UCI Machine Learning

Repository dengan nama file Holiday.csv seperti pada gambar 4.6

Pada codingan 4.6 yaitu mengimport librari pandas dimana kita mengaliaskan praktek sebagai pandas. Pandas berguna untuk mengelola dataframe = matrix = tabel kemudian memanggil nama alias untuk membaca format csvnya.

2. Dari dataframe yang sudah ada sebelumnya kita akan membagi menjadi 2 yaitu 450 row pertama dan 50 row sisanya dapat dilihat pada gambar 4.7

Name	Type	Size	Value
d_test	DataFrame	(50, 7)	Column names: User Id, Sports, Religious, Nature, Theatre, Shopping, P ...
d_train	DataFrame	(450, 7)	Column names: User Id, Sports, Religious, Nature, Theatre, Shopping, P ...
huda	DataFrame	(500, 7)	Column names: User Id, Sports, Religious, Nature, Theatre, Shopping, P ...

Variable explorer File explorer Help

IPython console

Console 1/A

```
In [2]: d_train=huda[:450]
...: d_test=huda[450:]
```

Figure 4.7: Membagi 2 Dataframe

Pada codingan 4.7 yaitu baris pertama dimana d_train untuk membagi data training sebanyak 450 row dan pada

baris kedua dimana d_test untuk data sisa atau data yang baru sebanyak 50 row.

3. Melakukan Vektorisasi dan Klasifikasi Data pada data Eminem pada gambar 4.8
huda merupakan dataframe keseluruhan dari file csv yang sudah dimasukkan dengan jumlah 448 baris dan 5 kolom. nospam merupakan dataframe yang isinya hanya data yang bukan termasuk spam dengan inisial angka 0. Sedangkan spam merupakan dataframe yang isinya hanya data spam dengan inisial angka 1.

huda	DataFrame	(448, 5)	Column names: COMMENT_ID, AUTHOR, DATE, CONTENT, CLASS
nospam	DataFrame	(203, 5)	Column names: COMMENT_ID, AUTHOR, DATE, CONTENT, CLASS
spam	DataFrame	(245, 5)	Column names: COMMENT_ID, AUTHOR, DATE, CONTENT, CLASS

Figure 4.8: Vektorisasi dan Klasifikasi Data

Pada gambar 4.9 merupakan hasil output content dimana terdapat 448 baris/data yang mempunyai 1602 kata-kata

yang digunakan pada komentar yang ada di content tersebut.

```
<448x1602 sparse matrix of type '<class 'numpy.int64'>'  
with 7105 stored elements in Compressed Sparse Row format>
```

Figure 4.9: Data Content

Pada gambar 4.10 maksud dari outputannya merupakan dataframe kata-kata tadi yang berjumlah 1602 kata orang

yang komen pada data eminem.

Name	Type	Size	Value
dk	list	1602	['00', '000', '047000', '09', '10', '100', '1000', '100877300245414', ...]

Figure 4.10: DataFrame Kata-kata Pada Content

4. Mengklasifikasikan dari data vektorisasi dengan menggunakan klasifikasi svm(support vektorisasi machine) dapat dilihat pada gambar 4.11.

```
In [20]: from sklearn import svm
....: clfsvm = svm.SVR(gamma='auto')
....: clfsvm.fit(huda_train_att, huda_train_label)
....: clfsvm.score(huda_test_att, huda_test_label)
Out[20]: 0.3360974270084298
```

Figure 4.11: Klasifikasi SVM Dari Data Vektorisasi

Jadi pada gambar 4.11 merupakan hasil dari memprediksi data score vektorisasi dengan svm menggunakan metode fit dimana digunakan untuk data training atau data pelatihannya saja.

5. Mengklasifikasikan dari data vektorisasi dengan menggunakan klasifikasi Decision Tree dapat dilihat pada gambar 4.12.

```
In [19]: from sklearn import tree
....: clftree = tree.DecisionTreeClassifier()
....: clftree.fit(huda_train_att, huda_train_label)
....: clftree.score(huda_test_att, huda_test_label)
Out[19]: 0.9324324324324325
```

Figure 4.12: Klasifikasi Decision Tree Dari Data Vektorisasi

Jadi pada gambar 4.12 merupakan hasil dari memprediksi data score vektorisasi dengan Decision Tree menggunakan metode fit dimana digunakan untuk data training atau data pelatihannya saja.

6. Mengeplot confusion matrix menggunakan matplotlib dapat dilihat pada gambar 4.13.

Pada gambar 4.13 sebelumnya kita harus mengimport matplotlibnya terlebih dahulu setelah itu di sini saya menggunakan numpy untuk mengeluarkan hasil plot confusion matrix pada matplotlibnya nantinya akan keluar normalisasi dari confusion matrix berupa data baris dan kolom.

7. Menjalankan program cross validation pada data vektorisasi dapat dilihat pada gambar 4.14.

```

In [22]: import numpy as np
....: np.set_printoptions(precision=2)
....: plot_confusion_matrix(cm, classes=huda, normalize=True)
....: plt.show()
Normalized confusion matrix
[[0.97 0.03]
 [0.06 0.94]]

```

Figure 4.13: Plot Confusion Matrix Menggunakan Matplotlib

```

In [23]: from sklearn.model_selection import cross_val_score
....: scores = cross_val_score(clf, huda_train_att, huda_train_label, cv=5)
....: # show average score and +/- two standard deviations away (covering 95% of scores)
....: print("Accuracy: %.2f (+/- %.2f)" % (scores.mean(), scores.std() * 2))
Accuracy: 0.95 (+/- 0.03)

In [24]: scorestree = cross_val_score(clftree, huda_train_att, huda_train_label, cv=5)
....: print("Accuracy: %.2f (+/- %.2f)" % (scorestree.mean(), scorestree.std() * 2))
Accuracy: 0.95 (+/- 0.05)

In [25]: scoressvm = cross_val_score(clfsvm, huda_train_att, huda_train_label, cv=5)
....: print("Accuracy: %.2f (+/- %.2f)" % (scoressvm.mean(), scoressvm.std() * 2))
Accuracy: 0.30 (+/- 0.42)

```

Figure 4.14: Program Cross Validation Pada Data Vektorisasi

Pada gambar4.14 yang pertama yaitu memunculkan akurasi cross validation dari random forest pada data yang sudah divektorisasi, yang kedua yaitu memunculkan akurasi cross validation dari decision tree pada data yang sudah divektorisasi, dan yang ketiga yaitu memunculkan akurasi cross validation dari svm pada data yang sudah divektorisasi.

8. Membuat program pengamatan komponen informasi pada data yang sudah di vektorisasi dapat dilihat pada gambar 4.15.

Pada gambar 4.15 merupakan hasil outputan yang mana max features di sana terdapat 9 data dari 10 yang sudah kita masukkan ke dalam codingan sebelumnya dan penomoran estimatornya merupakan data per 5 dari angka 40 sedangkan rata-rata akurasinya kita tuliskan datanya mulai dari 0.9 sampai dengan 1. Titik-titik yang didalam tersebut merupakan data vektorisasi dari pengamatan komponen informasinya.

4.3.3 Penanganan Eror

1. Pada gambar 4.16 merupakan ScreeShootan dari data yang eror.

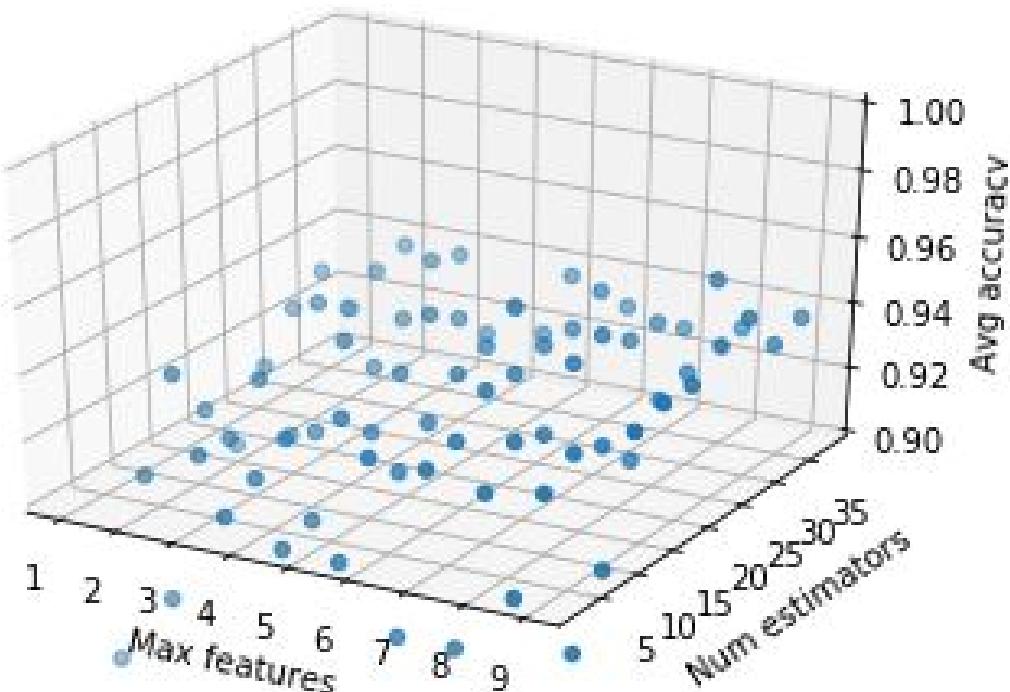


Figure 4.15: Program Pengamatan Komponen Informasi

```
In [28]: from sklearn import svm
....: clfsvm = svm.SVC()
....: clfsvm.fit(huda_train_att, huda_train_label)
....: clfsvm.score(huda_test_att, huda_test_label)
F:\anaconda\lib\site-packages\sklearn\svm\base.py:196: FutureWarning: The default value of
gamma will change from 'auto' to 'scale' in version 0.22 to account better for unscaled
features. Set gamma explicitly to 'auto' or 'scale' to avoid this warning.
    "avoid this warning.", FutureWarning)
Out[28]: 0.5945945945945946
```

Figure 4.16: Eror Pada Coding SVM

2. `clfsvm = svm.SVC()`
F:\anaconda\lib\site-packages\sklearn\svm\base.py:196: FutureWarning: The def
"avoid this warning.", FutureWarning)

3. Solusi pemecah masalah eror tersebut dengan mengganti dan menambahkan
codingan tersebut seperti berikut.

```
from sklearn import svm
clfsvm = svm.SVR(gamma='auto')
clfsvm.fit(huda_train_att, huda_train_label)
clfsvm.score(huda_test_att, huda_test_label)
```

Chapter 5

Conclusion

brief of conclusion

5.1 Ahmad Syafrizal Huda/1164062

5.1.1 Teori

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

Kata kata harus dilakukan vektorisasi dikarenakan untuk mengukur nilai kemunculan suatu kata yang sama dari sebuah kalimat sehingga kata-kata tersebut dapat di prediksi berapa kemunculannya. Atau juga di buatkan vektorisasi data yang digunakan untuk memprediksi bobot dari suatu kata misalkan mobil dan motor sama-sama kendaraan maka akan dibuat prediksi apakah kata tersebut akan muncul pada kalimat yang kira-kira memiliki bobot yang sama.

Untuk ilustrasinya dapat dilihat pada gambar 5.1



Figure 5.1: Ilustrasi Soal No. 1

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

Dimensi dari vektor dataset google bisa mencapai 300 karena dimensi dari vektor digunakan untuk membandingkan bobot dari setiap kata, misalkan terdapat kata mobil dan motor pada dataset google tersebut setiap kata tersebut di buat dimensi vektor 300 untuk kata mobil dan 300 dimensi vektor juga untuk kata motor kemudian kata tersebut di bandingkan bobot kesamaan katanya maka akan muncul akurasi sekitar 70an persen kesamaan bobot dikarenakan kata mobil dan motor sama sama di gunakan sebagai kendaraan.

Untuk ilustrasinya dapat dilihat pada gambar 5.2

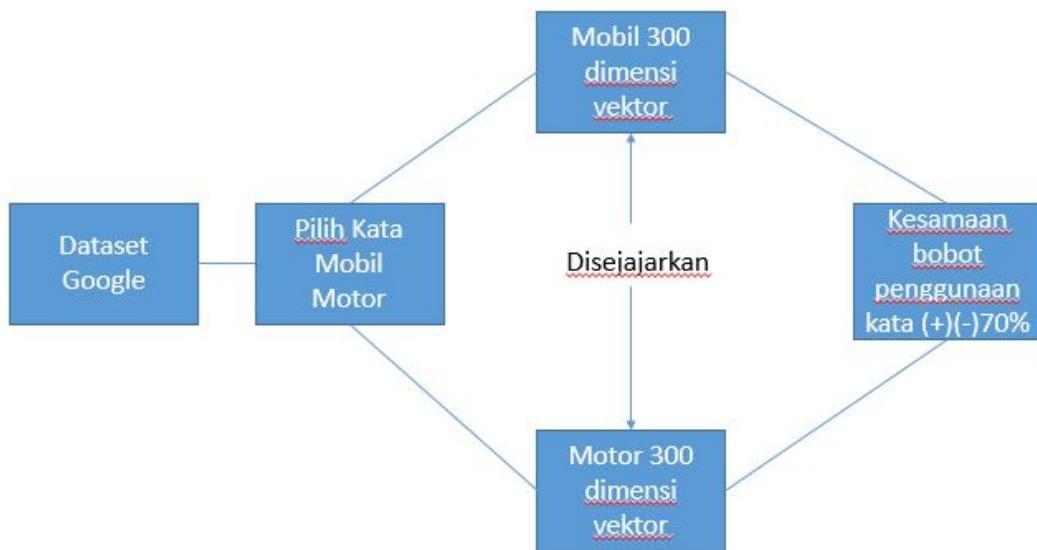


Figure 5.2: Ilustrasi Soal No. 2

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

Konsep vektorisasi untuk kata yaitu mengetahui kata tengah dari suatu kalimat utama dengan suatu kalimat contoh (Jangan lupa like dan comment yah makasih) kata tengah tersebut merupakan (dan) yang memiliki bobot sebagai kata tengah dari suatu kalimat atau bobot sebagai objek dari suatu kalimat. hal ini sangat berkaitan dengan dimensi vektor pada dataset google yang 300 tadi karena untuk mendapatkan nilai atau bobot dari kata tengah tersebut di dapatkan dari proses dimensi dari kata tersebut.

Untuk ilustrasinya dapat dilihat pada gambar 5.3



Figure 5.3: Ilustrasi Soal No. 3

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

Konsep vektorisasi untuk dokumen hampir sama seperti vektorisasi untuk kata hanya saja pemilihan kata utama atau kata tengah terdapat pada satu dokumen jadi mesin akan membuat dimensi vektor 300 untuk dokumen dan nanti kata tengahnya akan di sandingkan pada dokumen yang terdapat pada dokumen tersebut.

Untuk ilustrasinya dapat dilihat pada gambar 5.4



Figure 5.4: Ilustrasi Soal No. 4

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

Mean adalah nilai rata-rata dari suatu data. Mean disini merupakan petunjuk terhadap kata-kata yang diolah jika kata-kata itu akurasinya tinggi berarti kata tersebut sering muncul begitu juga sebaliknya. Standar deviasi merupakan standar untuk menimbang kesalahan. Misalkan kita memperkirakan kedalaman dari dataset merupakan 2 atau 3 tapi pada kenyataannya merupakan 5 itu merupakan kesalahan tapi masih bisa dianggap wajar karena masih mendekati perkiraan awal.

Untuk ilustrasinya dapat dilihat pada gambar 5.5

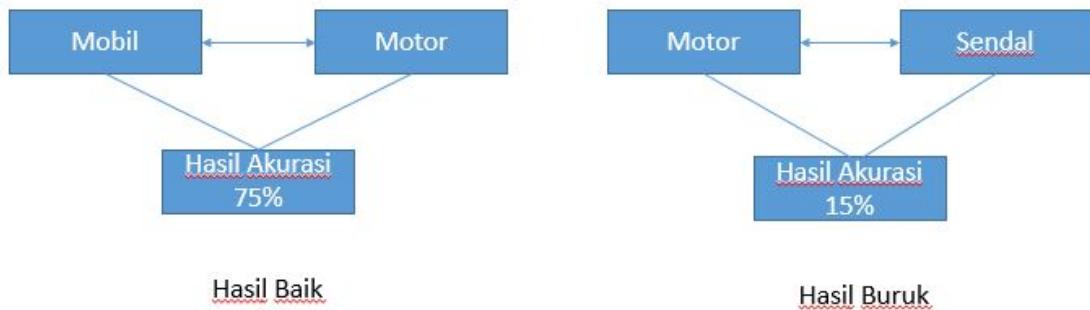


Figure 5.5: Ilustrasi Soal No. 5

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

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

Untuk ilustrasinya dapat dilihat pada gambar 5.6

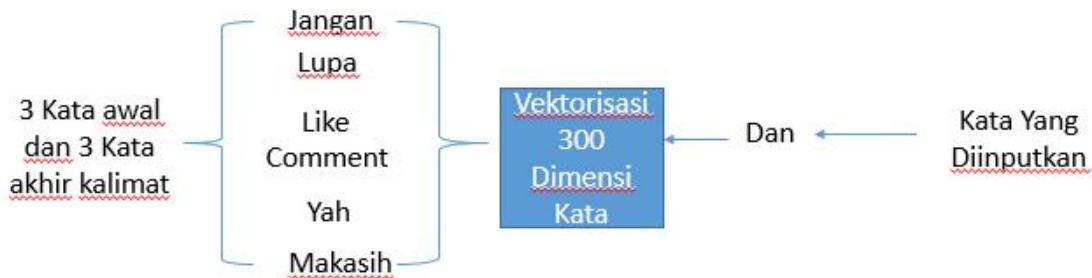


Figure 5.6: Ilustrasi Soal No. 6

5.1.2 Praktek Program

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

Output source code dibawah akan memunculkan data vektor untuk kata love. bahwa vektor memiliki array sebanyak 300 dimensi. Hasil pada source code tersebut dapat dilihat pada gambar 5.7.

```
In [34]: gmodel['love']
Out[34]:
array([ 0.10302734, -0.15234375,  0.02587891,  0.16503906, -0.16503906,
```

Figure 5.7: Love

```
gmodel['love']
```

Output source code dibawah akan memunculkan data vektor untuk kata faith. bahwa vektor memiliki array sebanyak 300 dimensi. Hasil pada source code tersebut dapat dilihat pada gambar 5.8.

```
gmodel['faith']
```

```
In [35]: gmodel['faith']
Out[35]:
array([ 0.26367188, -0.04150391,  0.1953125 ,  0.13476562, -0.14648438,
```

Figure 5.8: Faith

Output source code dibawah akan memunculkan data vektor untuk kata fall. bahwa vektor memiliki array sebanyak 300 dimensi. Hasil pada source code tersebut dapat dilihat pada gambar 5.9.

```
gmodel['fall']
```

```
In [36]: gmodel['fall']
Out[36]:
array([-0.04272461,  0.10742188, -0.09277344,  0.16894531, -0.1328125 ,
```

Figure 5.9: Fall

Output source code dibawah akan memunculkan data vektor untuk kata sick. bahwa vektor memiliki array sebanyak 300 dimensi. Hasil pada source code tersebut dapat dilihat pada gambar 5.10.

```
gmodel['sick']
```

Output source code dibawah akan memunculkan data vektor untuk kata clear. bahwa vektor memiliki array sebanyak 300 dimensi. Hasil pada source code tersebut dapat dilihat pada gambar 5.11.

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

Figure 5.10: Sick

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

Figure 5.11: Clear

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

Output source code dibawah akan memunculkan data vektor untuk kata shine. bahwa vektor memiliki array sebanyak 300 dimensi. Hasil pada source code tersebut dapat dilihat pada gambar 5.12.

```
gmodel['shine']
```

```
In [39]: gmodel['shine']
Out[39]:
array([-0.12402344,  0.25976562, -0.15917969, -0.27734375,  0.30273438,
```

Figure 5.12: Shine

Output source code dibawah akan memunculkan data vektor untuk kata bag. bahwa vektor memiliki array sebanyak 300 dimensi. Hasil pada source code tersebut dapat dilihat pada gambar 5.13.

```
gmodel['bag']
```

```
In [40]: gmodel['bag']
Out[40]:
array([-0.03515625,  0.15234375, -0.12402344,  0.13378906, -0.11328125,
```

Figure 5.13: Bag

Output source code dibawah akan memunculkan data vektor untuk kata car. bahwa vektor memiliki array sebanyak 300 dimensi. Hasil pada source code tersebut dapat dilihat pada gambar 5.14.

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

Figure 5.14: Car

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

Output source code dibawah akan memunculkan data vektor untuk kata wash. bahwa vektor memiliki array sebanyak 300 dimensi. Hasil pada source code tersebut dapat dilihat pada gambar 5.15.

```
gmodel['wash']
```

```
In [42]: gmodel['wash']
Out[42]:
array([ 9.46044922e-03,  1.41601562e-01, -5.46875000e-02,  1.34765625e-01,
```

Figure 5.15: Wash

Output source code dibawah akan memunculkan data vektor untuk kata motor. bahwa vektor memiliki array sebanyak 300 dimensi. Hasil pada source code tersebut dapat dilihat pada gambar 5.16.

```
gmodel['motor']
```

```
In [43]: gmodel['motor']
Out[43]:
array([ 5.73730469e-02,  1.50390625e-01, -4.61425781e-02, -1.32812500e-01,
```

Figure 5.16: Motor

Output source code dibawah akan memunculkan data vektor untuk kata cycle. bahwa vektor memiliki array sebanyak 300 dimensi. Hasil pada source code tersebut dapat dilihat pada gambar 5.17.

```
gmodel['cycle']
```

Pada source code dibawah menunjukkan hasil score perbandingan kata apakah kata motor dan cycle memiliki ke samaan atau tidak. Hasil pada source code tersebut dapat dilihat pada gambar 5.18.

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

Figure 5.17: Cycle

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

Figure 5.18: Similariti Pada Kata Motor dan Cycle

```
gmodel.similarity('motor','cycle')
```

Pada source code dibawah menunjukkan hasil score perbandingan kata apakah kata wash dan motor memiliki ke samaan atau tidak. Hasil pada source code tersebut dapat dilihat pada gambar 5.19.

```
gmodel.similarity('wash','motor')
```

```
In [46]: gmodel.similarity('wash','motor')
Out[46]: 0.10280077965607967
```

Figure 5.19: Similariti Pada Kata Wash dan Motor

Untuk Motor dan Cycle hasilnya adalah 17%

Untuk Wash dan Motor hasilnya adalah 10%

Artinya Motor dan Cyle memang dalam kategori yang sama misalnya dalam kategori kata-kata yang disatukan/berpasangan. Mesin sudah mengetahui bahwa keduanya dapat dikategorikan sebagai sepasang kata.

2. Jelaskan dengan kata dan ilustrasi fungsi dari extract_words dan PermuteSentences.

Extract_Words merupakan function untuk menambahkan, menghilangkan atau menghapuskan, hal hal yang tidak penting atau tidak perlu di dalam teks. Pada gambar 5.20 berikut ini menggunakan function extract_words untuk

```
In [48]: import re
.... def extract_words(luarbiasa):
....     luarbiasa = luarbiasa.lower()
....     luarbiasa = re.sub(r'<[^>]+>', ' ', luarbiasa) #hapus tag html
....     luarbiasa = re.sub(r'(\w)\\'(\w)', ' ', luarbiasa) #hapus petik satu
....     luarbiasa = re.sub(r'\W', ' ', luarbiasa) #hapus tanda baca
....     luarbiasa = re.sub(r'\s+', ' ', luarbiasa) #hapus spasi yang berurutan
....     return luarbiasa.split()
```

Figure 5.20: Extract_Words

menghapus komen dengan python style , mencari data yang diinginkan, dan memberikan spasi pada teks.

PermuteSentences berfungsi untuk melakukan pengacakan data supaya memperoleh data yang teratur. Ini merupakan class yang digunakan untuk melakukan pengocokan secara acak pada data yang ada. Digunakan cara ini agar tidak terjadi kelebihan memori pada saat dijalankan. Hasilnya dapat dilihat pada gambar 5.21.

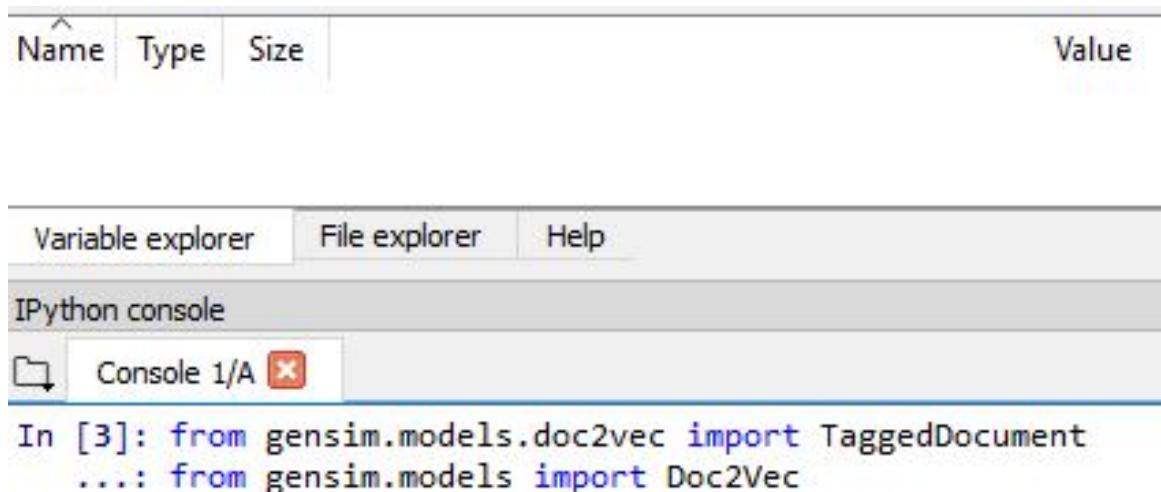
```
In [49]: import random
.... class PermuteSentences(object):
....     def __init__(self, luarbiasas):
....         self.luarbiasas = luarbiasas
....     def __iter__(self):
....         shuffled = list(self.luarbiasas)
....         random.shuffle(shuffled)
....         for luarbiasa in shuffled:
....             yield luarbiasa
```

Figure 5.21: Permute Sentences

3. Jelaskan fungsi dari librari gensim TaggedDocument dan Doc2Vec disertai praktik pemakaiannya.

Fungsi dari library gensim yaitu sebagai pemodelan topik tanpa pengawasan dan pemrosesan bahasa alami, atau bisa kita sebut dengan unsupervised. tagged document itu sendiri untuk memasukan kata-kata pada setiap dokumennya yang akan di vektorisasi. Fungsi dari doc2vec itu sendiri ialah untuk membandingkan

bobot data yang terdapat pada dokumen lainnya, apakah kata-kata didalamnya ada yang sama atau tidak. Ketika di running maka tidak terjadi apa-apa, seperti pada gambar 5.22



The screenshot shows a Jupyter Notebook interface. At the top, there is a toolbar with 'Variable explorer', 'File explorer', and 'Help' buttons. Below the toolbar, it says 'IPython console'. A tab labeled 'Console 1/A' is selected. In the main area, the code cell contains:

```
In [3]: from gensim.models.doc2vec import TaggedDocument  
....: from gensim.models import Doc2Vec
```

Figure 5.22: Gensim TaggedDocument dan Doc2vec

4. Jelaskan dengan kata dan praktek cara menambahkan data training dari file yang dimasukkan kepada variabel dalam rangka melatih model doc2vec.

Untuk menambahkan data training yaitu melakukan import library os, library os berfungsi untuk melakukan interaksi antara python dengan os laptop kita masing-masing, setelah itu kita buat variabel unsup sentences. Kemudian pilih direktori tempat data kita disimpan. Lalu menyortir data yang terdapat pada folder aclImdb dan membaca file tersebut dengan ekstensi .txt.

```
import os  
unsup_sentences = []  
for dirname in ["train/pos","train/neg","train/unsup","test/pos","test/neg"]:  
    for fname in sorted(os.listdir("aclImdb/"+dirname)):  
        if fname[-4:] == '.txt':  
            with open("aclImdb/"+dirname+"/"+fname,encoding='UTF-8') as f:  
                sent = f.read()  
                words = extract_words(sent)  
                unsup_sentences.append(TaggedDocument(words,[dirname+"/"+fname]))
```

Hasil dari code yang diatas ialah terdapatnya data training dengan jumlah 10000 data dari variabel unsup_sentences hasil running dari folder aclImdb dapat dilihat pada gambar 5.23

Name	Type	Size	Value
dirname	str	1	test/neg
fname	str	1	9_4.txt
sent	str	1	David Bryce's comments nearby are exceptionally well written and infor ...
unsup_sentences	list	100000	[TaggedDocument, TaggedDocument, TaggedDocument, TaggedDocument, Tagge ...]
words	list	391	['david', 'bryc', 'comments', 'nearby', 'are', 'exceptionally', 'well' ...]

Figure 5.23: Aclimbdb

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

Untuk code berikutnya akan menambahkan data training pada variabel unsup_sentences sekitar 64.720 data, seperti pada gambar 5.24

Name	Type	Size	Value
dirname	str	1	txt_sentoken/neg
fname	str	1	cv999_14636.txt
i	int	1	24
sent	str	1	after watching _a_night_at_the_roxbury_ , you'll be left with exactly ...
unsup_sentences	list	164720	[TaggedDocument, TaggedDocument, TaggedDocument, TaggedDocument, Tagge ...]
words	list	11	['after', 'watching', '_a_night_at_the_roxbury_', 'yo', 'l', 'be', 'le' ...]

Figure 5.24: Aclimbdb

```
with open("stanfordSentimentTreebank/original_rt_snippets.txt", encoding='UTF-8') as f:
    for i, sent in enumerate(f):
```

```

words = extract_words(sent)
unsup_sentences.append(TaggedDocument(words, ["rt-%d" % i]))

```

Untuk code berikutnya akan menambahkan data training pada variabel unsup_sentences sekitar 10.605 data, seperti pada gambar 5.25

Name	Type	Size	Value
dirname	str	1	txt_sentoken/neg
fname	str	1	cv999_14636.txt
i	int	1	10604
sent	str	1	Her fans walked out muttering words like ``horrible'' and ``terrible,' ...
unsup_sentences	list	175325	[TaggedDocument, TaggedDocument, TaggedDocument, TaggedDocument, Tagge ...]
words	list	29	['her', 'fans', 'walked', 'out', 'muttering', 'words', 'like', 'horrib ...']

Figure 5.25: Aclimbdb

5. Jelaskan dengan kata dan praktek kenapa harus dilakukan pengocokan dan pembersihan data.

Pengocokan data itu berguna untuk mengacak data supaya pada saat data di running bisa berjalan lebih baik dan hasil presentase akhirnya bisa lebih bagus. Sedangkan pembersihan data untuk memberikan ruang bagi ram komputer kita setelah melakukan running data sebanyak 3 juta lebih, agar lebih ringan saat proses selanjutnya. Hasil dari pengacakan data tidak ditampilkan pada spyder. Dan sebelumnya memori yang terpakai itu sekitar 87% , setelah dikosongkan jadi 71%. Untuk source codenya dapat dilihat sebagai berikut:

```

# Pengocokan data
mute = PermuteSentences(unsup_sentences)
# Pembersihan data
model.delete_temporary_training_data(keep_inference=True)

```

6. Jelaskan dengan kata dan praktek kenapa model harus di save dan kenapa temporari training harus dihapus.

Save data untuk menyimpan file hasil dari proses training data sebelumnya, model tersebut dilakukan penyimpanan untuk memberikan keringanan pada ram agar saat kita akan melakukan training lagi, model tersebut tinggal di muat saja tanpa harus melakukan training dari awal dan bisa menghemat waktu.

Sedangkan untuk delete temporary training data untuk menghapus data training yang sebelumnya sudah dilakukan dan disimpan, tujuannya memberikan keringinan pada ram. Karena setelah melakukan proses training, ram biasanya jadi berat untuk membaca sampai komputer kita jadi lemot. Untuk source codenya dapat dilihat sebagai berikut:

```
# Save data  
model.save('ocean.d2v')  
# Delete temporary data  
model.delete_temporary_training_data(keep_inference=True)
```

7. Jalankan dengan kata dan praktek maksud dari infer code.

Infer vector yaitu membandingkan kata yang tercantum dengan vektor pada dokumen yang sudah di muat sebelumnya. Selain itu infer_vector juga menghitung vektor dari kata yang dicantumkan pada model yang telah kita buat. Seharusnya kata yang dicantumkan itu lebih panjang lagi agar hasilnya bisa lebih baik lagi.

```
model.infer_vector(extract_words("I will go home"))
```

Pada source code tersebut menghasilkan keluaran seperti pada gambar 5.26.

```
In [14]: model.infer_vector(extract_words("I will go home"))  
Out[14]:  
array([-0.13122962, -0.2173184 , -0.14073701,  0.47829896,  0.10263892,  
-0.12006506, -0.02554635, -0.06321128,  0.01674332,  0.21312888,  
-0.03759272, -0.21068032,  0.08137176, -0.04404473,  0.23163871,  
-0.11791784, -0.02098055, -0.44984344, -0.21420991,  0.10826829,  
-0.24330835, -0.13386108,  0.05421483, -0.10329439, -0.16347748,  
0.03265174, -0.18829556,  0.14504528, -0.03915659, -0.39758494,  
0.05695166,  0.02784907, -0.15537408,  0.14412752, -0.0224928 ,  
-0.40351996, -0.11055487,  0.18106677,  0.09285883, -0.2614472 ,  
0.24014267,  0.15855208, -0.1287663 ,  0.53696984,  0.08572944,  
0.2602722 ,  0.3341717 ,  0.12752089,  0.15490048,  0.25898734,  
-0.23501003, -0.03285267], dtype=float32)
```

Figure 5.26: Infer Code

8. Jelaskan dengan praktek dan kata maksud dari cosine similarity.

Cosine similarity yaitu membandingkan vektorisasi data diantara kedua kata yang di masukkan, Jika hasil presentase dari kedua kata tersebut lebih dari 50 persen kemungkinan kata tersebut terdapat dalam 1 file. Tapi jika kurang dari 50 persen kata tersebut tidak terdapat dalam 1 file. Hasil yang didapatkan pada code tersebut hanya 0.4 persen itu dikarenakan kata pertama dan kedua tidak memiliki kesamaan vektorisasi dan tidak terdapat pada salah satu dokumen.

```
from sklearn.metrics.pairwise import cosine_similarity
cosine_similarity(
[model.infer_vector(extract_words("she going to school, after wash hand"))],
[model.infer_vector(extract_words("Services sucks."))])
```

Pada source code tersebut menghasilkan keluaran seperti pada gambar 5.27.

```
In [15]: from sklearn.metrics.pairwise import cosine_similarity
...: cosine_similarity(
...:     [model.infer_vector(extract_words("she going to school, after wash hand"))],
...:     [model.infer_vector(extract_words("Services sucks."))])
Out[15]: array([[0.46642035]], dtype=float32)
```

Figure 5.27: Cosine Similarity

9. Jelaskan dengan praktek score dari cross validation masing-masing metode.

```
scores = cross_val_score(clf, sentvecs, sentiments, cv=5)
np.mean(scores), np.std(scores)
```

Pada source code tersebut melakukan perhitungan presentase dengan menggunakan cross validation yang mana mengecek data score menggunakan metode kneighborsClassifier untuk menghasilkan typedata float64 dimana terdapat 5 data dari score yang typedatanya float64 tersebut. menghasilkan keluaran seperti pada gambar 5.28.

```
scores = cross_val_score(clfrf, sentvecs, sentiments, cv=5)
np.mean(scores), np.std(scores)
```

The screenshot shows a Jupyter Notebook interface. At the top, there's a toolbar with 'Variable explorer', 'File explorer', and 'Help'. Below that is the 'IPython console' header with 'Console 1/A' selected. The code cell In [20] contains the command to calculate cross-validation scores. The output Out[20] shows the mean score of 0.765 and its standard deviation of 0.016532795690182997.

```
In [20]: scores = cross_val_score(clf, sentvecs, sentiments, cv=5)
....: np.mean(scores), np.std(scores)
Out[20]: (0.765, 0.016532795690182997)
```

Figure 5.28: Cross Validation Metode 1

```
Out[21]: (0.7150000000000001, 0.02378141197564929)
```

Figure 5.29: Cross Validation Metode 2

Pada source code tersebut melakukan perhitungan persentase dengan menggunakan cross validation yang mana mengecek data score menggunakan metode Random forest menghasilkan keluaran seperti pada gambar 5.29.

```
from sklearn.pipeline import make_pipeline
from sklearn.feature_extraction.text import CountVectorizer, TfidfTransformer
pipeline = make_pipeline(CountVectorizer(), TfidfTransformer(), RandomForestClassifier())
scores = cross_val_score(pipeline, sentences, sentiments, cv=5)
np.mean(scores), np.std(scores)
```

Pada source code tersebut melakukan skoring dari vektorisasi, tfidf, dan rf lalu dibuat perbandingan yang menghasilkan keluaran seperti pada gambar 5.30.

```
Out[22]: (0.749, 0.01271918935222596)
```

Figure 5.30: Cross Validation Metode 3

5.1.3 Penanganan Eror

1. ScreenShoot Eror dapat dilihat pada gambar 5.31
2. Tuliskan kode eror dan jenis erornya.

Kode eror

```
In [1]: import gensim, logging
F:\anaconda\lib\site-packages\gensim\utils.py:1197: UserWarning: detected Windows; aliasing
chunkize to chunkize_serial
    warnings.warn("detected Windows; aliasing chunkize to chunkize_serial")
```

Figure 5.31: eror

```
import gensim, logging
```

Jenis eronya.

```
F:\anaconda\lib\site-packages\gensim\utils.py:1197: UserWarning: detected Windows
warnings.warn("detected Windows; aliasing chunkize to chunkize_serial")
```

3. Solusi Pemecahan Eror Tersebut.

yaitu dengan cara memberikan source code seperti dibawah ini:

```
import warnings
warnings.filterwarnings(action='ignore', category=UserWarning, module='gensim')
```

Kenapa demikian itu karena gensim memberi tahu bahwa ia akan melakukan chunkize ke fungsi yang berbeda karena kita menggunakan os tertentu.

Chapter 6

Discussion

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

6.1 Ahmad Syafrizal Huda/1164062

6.1.1 Teori

1. Jelaskan kenapa file suara harus di lakukan MFCC.

File suara harus dilakukan MFCC(Mel Frequency Cepstral Coeficients) karena MFCC dapat mengubah file suara/frekuensi suara ke dalam bentuk data vektor yang nantinya akan diolah menjadi outputan dimana telah dilakukan ekstraksi oleh MFCC kemudian direalisasikan sebagai data matrix. Contoh ilustrasi dapat dilihat pada gambar 6.1.

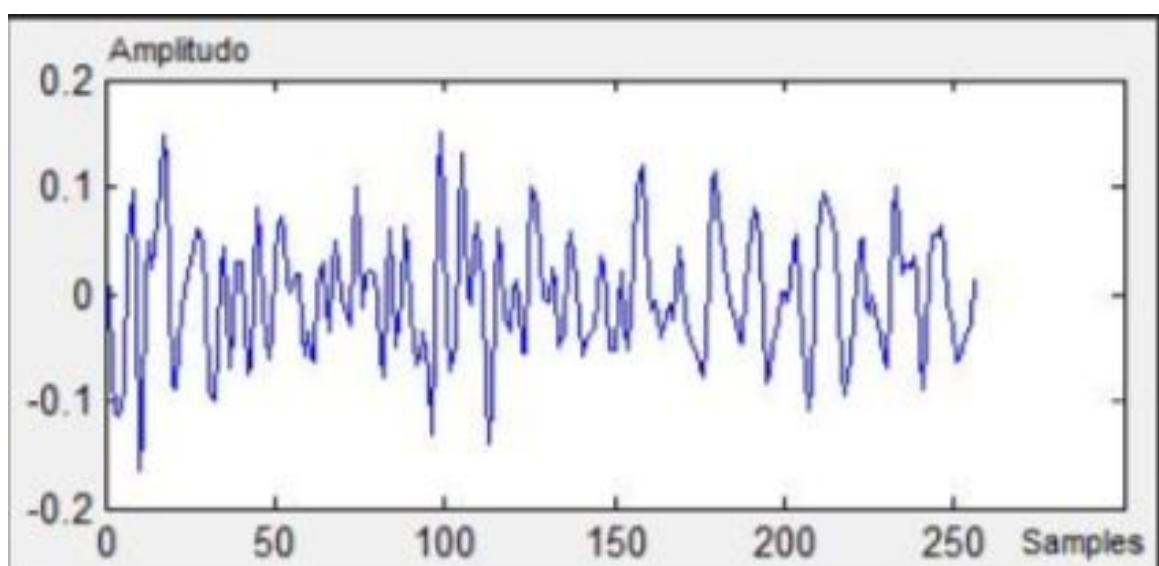


Figure 6.1: Suara ke MFCC

2. Jelaskan konsep dasar neural network.

Neural Network merupakan sistem komputasi yang efisien dimana tema utamanya dipinjam dari analogi jaringan saraf biologis. Neural Network biasa disebut dengan jaringan saraf tiruan dimana Neural Network sebenarnya mengadopsi dari kemampuan otak manusia yang mampu memberikan stimulasi/rangsangan, melakukan proses, dan memberikan output. Contoh ilustrasi dapat dilihat pada gambar 6.2.

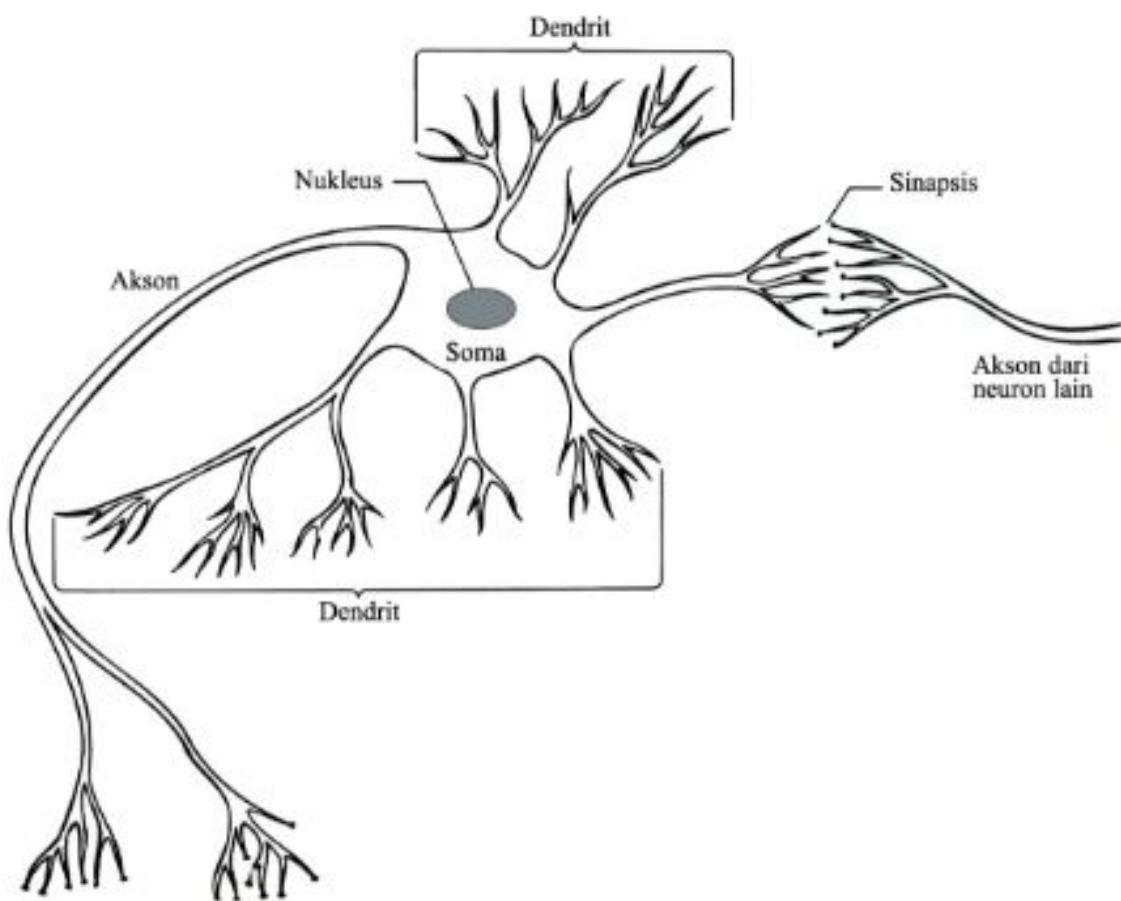


Figure 6.2: Neural Network

3. Jelaskan konsep pembobotan dalam neural network.

Sebuah Neural Network dikonfigurasi untuk aplikasi tertentu, seperti pengenalan pola atau klasifikasi data, contohnya pada Neural Network melakukan penyesuaian koneksi sinaptik antar neuron dilakukan dengan menyesuaikan nilai bobot yang ada pada tiap koneksi baik dari input, neuron maupun out-

put disinkronkan dengan penyesuaian koneksi sinaptik antar neuron itu sendiri. Contoh ilustrasi dapat dilihat pada gambar 6.3.

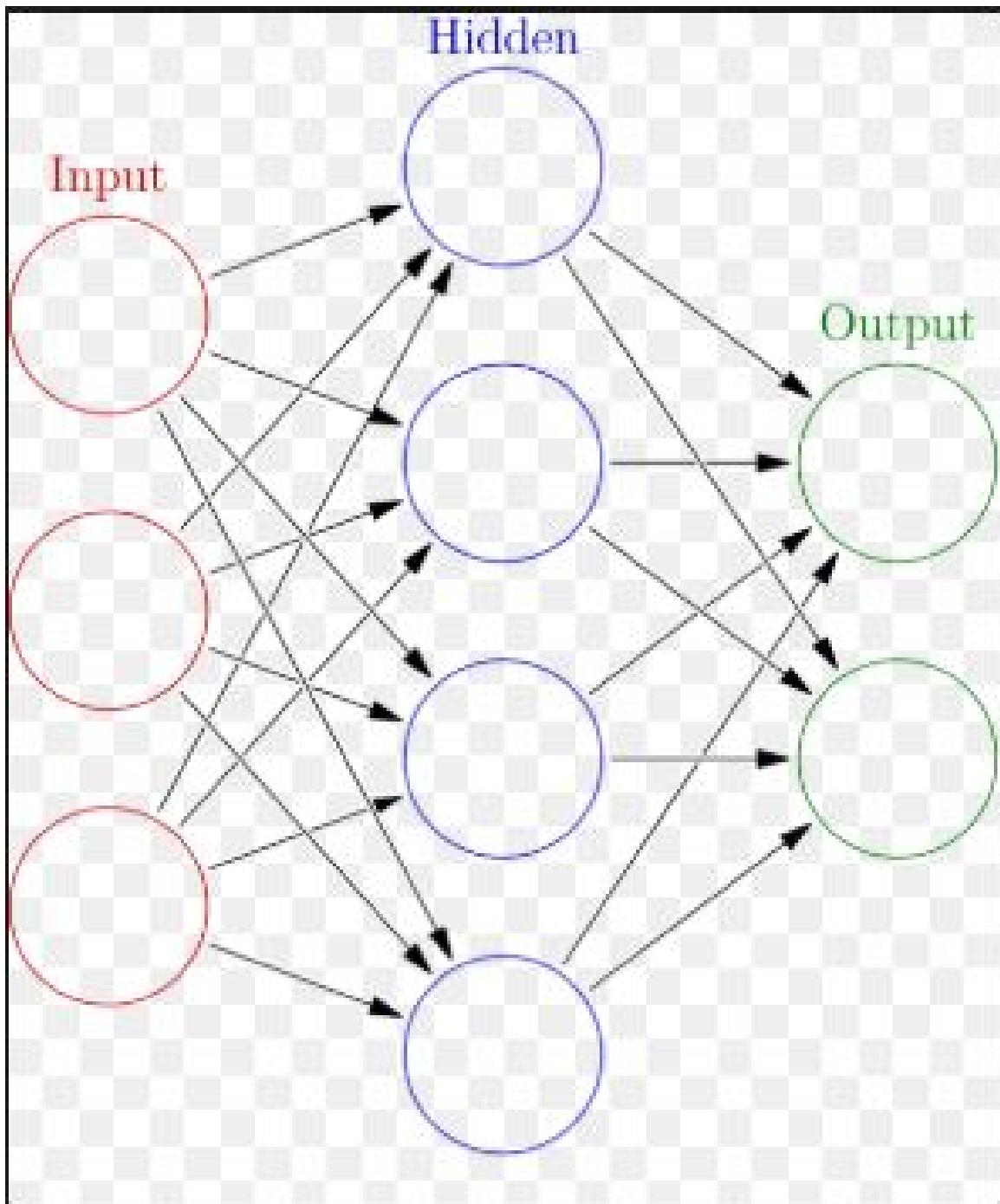


Figure 6.3: Pembobotan Neural Network

4. Jelaskan konsep fungsi aktifasi dalam neural network.

Fungsi aktivasi dalam neural network yaitu setiap neuron mempunyai tingkat aktivasi yang merupakan fungsi dari input yang masuk padanya. Aktivasi yang dikirim suatu neuron ke neuron lain berupa sinyal dan hanya dapat mengirim sekali dalam satu waktu, meskipun sinyal tersebut disebarluaskan pada beberapa neuron yang lain. Contoh ilustrasi dapat dilihat pada gambar 6.4.

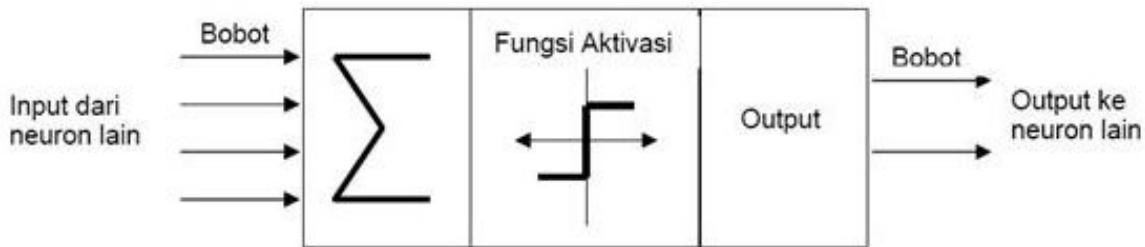


Figure 6.4: Aktivasi Neural Network

5. Jelaskan cara membaca hasil plot dari MFCC.

Cara membaca hasil plot dari MFCC yaitu Nanti akan ada outputan berbentuk grafik setelah melakukan plot dari MFCC. Kemudian terdapat frekuensi/Hz pada suara frekuensi biasanya vertikal atau bisa disimbolkan dengan sumbu y, Lalu terdapat waktu yang mana waktu diartikan dalam simbol sumbu x. Sedangkan pada bagian dalam atau bisa disimbolkan dengan sumbu z merupakan power atau kekuatan dari lagu atau suara atau desibel yang dihasilkan. Untuk warna biru itu merupakan suara rendah, yang merah merupakan tinggi apabila daya frekuensi nya misalkan suara siul berarti dominan warna merah karena siul biasanya pada nada yang tinggi sedangkan jika bass dominan biru karena bass merupakan nada rendah. Contoh ilustrasi dapat dilihat pada gambar 6.5.

6. Jelaskan apa itu one-hot encoding.

One-hot encoding adalah representasi dari variabel kategori sebagai vektor biner. Yaitu nilai kategorika harus dipetakan ke nilai integer. Kemudian, setiap nilai integer direpresentasikan sebagai vektor biner yang semuanya bernilai nol kecuali indeks integer, yang ditandai dengan 1. Contoh ilustrasi dapat dilihat pada gambar 6.6.

7. Jelaskan apa fungsi dari np.unique dan to_categorical dalam kode program.

- np.unique untuk mengekstraksi elemen-elemen unik tertentu dalam array.

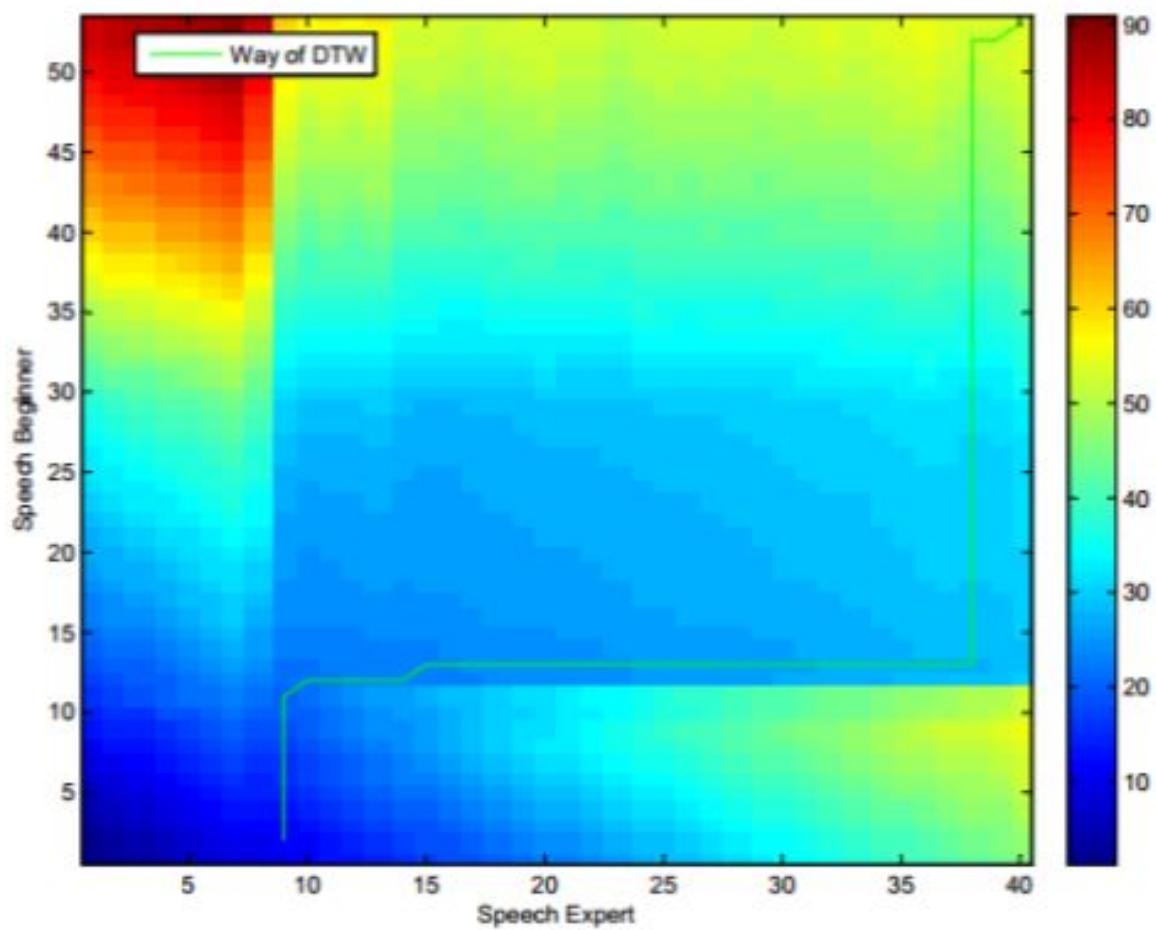


Figure 6.5: Membaca Hasil Plot dari MFCC

Original data:		One-hot encoding format:						
id	Color		id	White	Red	Black	Purple	Gold
1	White		1	1	0	0	0	0
2	Red		2	0	1	0	0	0
3	Black		3	0	0	1	0	0
4	Purple		4	0	0	0	1	0
5	Gold		5	0	0	0	0	1

Figure 6.6: One-Hot Encoding

- `to_categorical` untuk mengubah vektor kelas yang berupa integer menjadi matriks kelas biner.

```
>>> a = np.array([1,1,1,2,2,3,4,4,4,4,5,5,5,5,5], float)
>>> np.unique(a)
array([ 1.,  2.,  3.,  4.,  5.])
```

Figure 6.7: `np.unique`

```
> labels
array([0, 2, 1, 2, 0])
> to_categorical(labels)
array([[ 1.,  0.,  0.],
       [ 0.,  0.,  1.],
       [ 0.,  1.,  0.],
       [ 0.,  0.,  1.],
       [ 1.,  0.,  0.]], dtype=float32)
```

Figure 6.8: `to_categorical`

8. Jelaskan apa fungsi dari Sequential dalam kode program.

Fungsi dari Sequential dalam kode program yaitu merupakan sebuah jenis model yang digunakan dalam perhitungan ataupun code program yang direalisasikan. Neural Networks Sequential membangun fitur tingkat tinggi melalui lapisan yang berurutan. Sequential juga merupakan proses dimana membandingkan setiap elemen larik satu per satu secara beruntun, mulai dari elemen pertama, sampai dengan elemen terakhir atau elemen yang dicari sudah ditemukan. Contoh ilustrasi dapat dilihat pada gambar 6.9.

6.1.2 Praktek Program

1. Jelaskan isi dari data GTZAN Genre Collection dan data dari freesound. Buat kode program untuk meload data tersebut untuk digunakan pada MFCC.

```

Pencarian Sequential
Jumlah data : 5
Indeks [1] : 1
Indeks [2] : 8
Indeks [3] : 4
Indeks [4] : 6
Indeks [5] : 9
Masukkan Data Yang Ingin Dicari : 6
6 Ditemukan pada indeks ke-4

```

Figure 6.9: One-Hot Encoding

Isi dari data GTZAN Genre Collection itu isinya berupa data musik yang sudah di folderkan berdasarkan genre lagunya (dikelompokkan) dengan ekstensi .au yang akan kita lakukan proses MFCC. Sedangkan freesound hanya untuk 1 lagu saja dengan ekstensi .wav.

```

filename_jazz = 'genres/jazz/jazz.00062.au'
x_jazz, sr_jazz = librosa.load(filename_jazz, duration=10)

```

- Code pada baris pertama yaitu Filename jazz merupakan variabel yang berisikan direktori dari file yang dituju, pada code ini digunakan file audio dari genre jazz.
- Code pada baris kedua yaitu Membuat variabel X jazz dan sr jazz yang digunakan untuk meload file dari variabel filename jazz menggunakan librari Librosa dengan durasi 10, yang nantinya akan digunakan pada Mfcc.

Hasilnya dapat dilihat pada gambar 6.10.

2. Jelaskan perbaris kode program dengan kata-kata dan dilengkapi ilustrasi gambar fungsi dari display mfcc() .

```
display_mfcc('genres/hiphop/hiphop.00028.au')
```

Pada kode tersebut menjelaskan display_mfcc untuk menampilkan vektorisasi dari sebuah suara yang akan menampilkan Plot dan Bar dari inputan code dan file suara hiphop.00028.au. Hasilnya dapat dilihat pada gambar 6.11.

Name	Type	Size	Value
filename_jazz	str	1	genres/jazz/jazz.00062.au
sr_jazz	int	1	22050
x_jazz	float32	(220500,)	[-0.28155518 -0.3609314 -0.4019165 ... -0.04586792 -0.052... -0.0 ...]

Figure 6.10: GTZAN

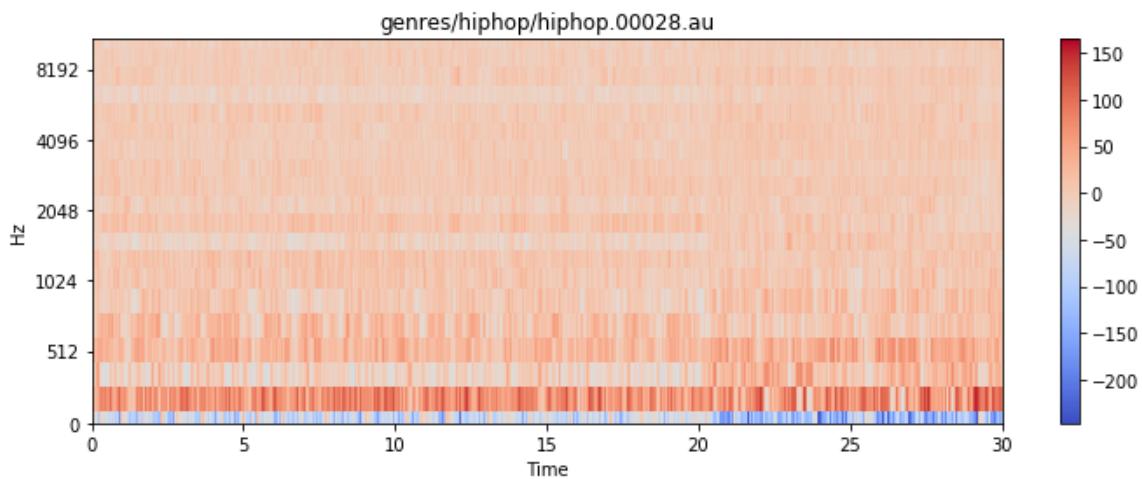


Figure 6.11: Display MFCC

3. Jelaskan perbaris kode program dengan kata-kata dan dilengkapi ilustrasi gambar fungsi dari extract features song(). Jelaskan juga mengapa yang diambil 25.000 baris pertama?

```
display_mfcc('genres/hiphop/hiphop.00028.au')
```

- Pada kode baris pertama berfungsi untuk memuat inputan.
- Pada kode baris kedua itu akan memuat data inputan dengan menggunakan librosa.
- Kemudian pada parameter inputan yaitu y untuk membuat sebuah fitur pada mfcc.
- Selanjutnya me-return menjadi array dan akan mengambil 25000 data saja dari hasil vektorisasi dalam 1 lagu yang sudah di display_mfcc.
- Mengapa yang dimabil 25.000 baris pertama karena nemuat nilai-nilai MFCC untuk lagu, tetapi karena nilai-nilai ini mungkin antara negatif 250 hingga positif 150, mereka tidak baik untuk jaringan saraf.

4. Jelaskan perbaris kode program dengan kata-kata dan dilengkapi ilustrasi gambar fungsi dari generate features and labels().

```
def generate_features_and_labels():
    all_features = []
    all_labels = []

    genres = [ 'blues' , 'classical' , 'country' , 'disco' , 'hiphop' , 'jazz' , 'metal' , 'pop' , 'reggae' , 'rock' ]
    for genre in genres:
        sound_files = glob.glob( 'genres/' + genre + '/*.au' )
        print( 'Processing %d songs in %s genre...' , % (len(sound_files) , genre) )
        for f in sound_files:
            features = extract_features_song(f)
            all_features.append(features)
            all_labels.append(genre)

    # convert labels to one-hot encoding cth blues : 1000000000 classical : 0000000001
    label_uniq_ids , label_row_ids = np.unique(all_labels , return_index=True)
    label_row_ids = label_row_ids.astype(np.int32 , copy=False)
    onehot_labels = to_categorical(label_row_ids , len(label_uniq_ids))
    return np.stack(all_features) , onehot_labels
```

Pada kode diatas berfungsi melakukan fungsi yang sebelumnya kita telah jalankan. Lalu pada bagian genres itu disesuaikan dengan nama folder dataset. Untuk baris selanjutnya itu akan melakukan looping dari folder genres dengan ekstensi .au. Lalu akan memanggil fungsi ekstrak lagu. Pada setiap file yang terdapat pada folder itu akan di ekstrak menjadi vektor dan akan dimasukan kedalam fitur. Dan fungsi append itu menumpuk file-file yang telah di vektorisasi.

5. Jelaskan dengan kata dan praktek kenapa penggunaan fungsi generate features and labels() sangat lama ketika meload dataset genre.

```
features , labels = generate_features_and_labels()
```

Pada kode diatas menjelaskan dikarenakan terdapat 10 folder dengan genre berbeda, dan didalamnya terdapat 100 audio maka itu akan lama untuk meload dataset genre. Dari setiap folder itu akan dilakukan features dan perubahan label dengan ekstraksi data menggunakan mfcc. Karena banyaknya jumlah file maka proses loadnya pun lama. Hasil dapat dilihat pada gambar 6.12.

6. Jelaskan kenapa harus dilakukan pemisahan data training dan data set sebesar 80 persen?

Name	Type	Size	Value
features	float64	(1000, 25000)	[[-0.81999071 -0.81014303 -0.75184401 ... -0.01818394... 0 ...
██████████	██████████	██████████	██████████
labels	float32	(1000, 10)	[[1. 0. 0. ... 0. 0. 0.] [1. 0. 0. ... 0. 0. 0.]

Figure 6.12: Generate Features and Labels

```
training_split = 0.8
```

Pada kode diatas berfungsi untuk memisahkan data training dan dataset sebesar 80% digunakan untuk memudahkan dalam melakukan pengacakan atau pengocokan nantinya. Dimana 80% merupakan data training dan sisanya 20% merupakan datatestnya. data training perlu lebih banyak agar saat dilakukan pengocokan tidak teracak dalam urutan yang berbeda. Hasilnya dapat dilihat pada gambar 6.13.

training_split	float	1	0.8
----------------	-------	---	-----

Figure 6.13: Training Data Sebesar 80%

7. Praktekkan dan jelaskan masing-masing parameter dari fungsi Sequential().

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

Pada kode diatas dijelaskan bahwa:

- Layer pertama dense dari 100 neuron untuk inputan
- Activationnya menggunakan fungsi relu yaitu jika ada inputan dengan nilai maksimum maka inputan itu yang akan terpilih.
- Dense 10 mengkategorikan 10 neuron untuk jenis genrenya untuk outputnya.
- Untuk dense diatas aktivasinya menggunakan fungsi Softmax

8. Praktekkan dan jelaskan masing-masing parameter dari fungsi compile() dan tunjukkan keluarannya dengan fungsi summary.

```

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

```

Pada kode diatas dijelaskan bahwa:

- Menggunakan algortima adam sebagai optimizer. Adam yaitu algoritme pengoptimalan yang dapat digunakan sebagai ganti dari prosedur penu-runan gradien stokastik klasik untuk memperbarui bobot jaringan yang berulang berdasarkan data training.
- Loss nya menggunakan categorical_crossentropy untuk fungsi optimasi skor

Hasilnya dapat dilihat pada gambar 6.14.

Layer (type)	Output Shape	Param #
<hr/>		
dense_1 (Dense)	(None, 100)	2500100
activation_1 (Activation)	(None, 100)	0
dense_2 (Dense)	(None, 10)	1010
activation_2 (Activation)	(None, 10)	0
<hr/>		
Total params: 2,501,110		
Trainable params: 2,501,110		
Non-trainable params: 0		
<hr/>		
None		

Figure 6.14: Fungsi Compile

9. Praktekkan dan jelaskan masing-masing parameter dari fungsi fit().

```

model.fit(train_input, train_labels, epochs=10, batch_size=32,
          validation_split=0.2)

```

Pada kode tersebut dapat dilihat bahwa pada model fit digunakan untuk melatih mesin dengan data training input dan training label. Epochs ini merupakan iterasi atau pengulangan berapa kali data tersebut akan dilakukan. Batch_size ini adalah jumlah file yang akan dilakukan pelatihan pada setiap

1 kali pengulangan. Sedangkan validation_split itu untuk menentukan presen-tase dari cross validation atau k-fold sebanyak 20% dari masing-masing data pengulangan. Hasilnya dapat dilihat pada gambar 6.15.

```
640/640 [=====] - 2s 3ms/step - loss: 0.9898 - acc: 0.6703 -
val_loss: 1.6596 - val_acc: 0.4500
Epoch 4/10
640/640 [=====] - 2s 3ms/step - loss: 0.7642 - acc: 0.7688 -
val_loss: 1.5874 - val_acc: 0.4625
Epoch 5/10
640/640 [=====] - 2s 3ms/step - loss: 0.6307 - acc: 0.8141 -
val_loss: 1.5605 - val_acc: 0.4437
Epoch 6/10
640/640 [=====] - 2s 3ms/step - loss: 0.4229 - acc: 0.9109 -
val_loss: 1.5566 - val_acc: 0.5000
Epoch 7/10
640/640 [=====] - 2s 3ms/step - loss: 0.3339 - acc: 0.9516 -
val_loss: 1.6867 - val_acc: 0.4750
Epoch 8/10
640/640 [=====] - 2s 3ms/step - loss: 0.2515 - acc: 0.9719 -
val_loss: 1.6458 - val_acc: 0.4813
Epoch 9/10
640/640 [=====] - 2s 3ms/step - loss: 0.1881 - acc: 0.9828 -
val_loss: 1.5771 - val_acc: 0.4813
Epoch 10/10
640/640 [=====] - 2s 3ms/step - loss: 0.1470 - acc: 0.9984 -
val_loss: 1.5798 - val_acc: 0.4813
Out[18]: <keras.callbacks.History at 0x25a2985f390>
```

Figure 6.15: Fungsi Fit

10. Praktekkan dan jelaskan masing-masing parameter dari fungsi evaluate().

```
loss , acc = model.evaluate( test_input , test_labels , batch_size=32)
```

Pada kode diatas maka dapat dilihat bahwa dengan menggunakan test input dan test label dilakukan evaluasi atau proses menemukan model terbaik yang mewakili data dan seberapa baik model yang dipilih akan dijalankan kedepannya. Hasilnya dapat dilihat pada gambar 6.16.

11. Praktekkan dan jelaskan masing-masing parameter dari fungsi predict().

```
model.predict( test_input [:1])
```

Pada kode diatas akan melakukan testing satu data lagu. file yang di jalankan tersebut termasuk ke dalam genre apa, hasilnya bisa dilihat pada gambar tersebut presentase yang paling besar yakni genre hiphop. Maka lagu tersebut termasuk ke dalam genre hiphop dengan perbandingan presentase hasil prediksi. Hasilnya dapat dilihat pada gambar 6.17.

Name	Type	Size	Value
acc	float64	1	0.48
loaded	float64	(1000, 25010)	[-0.40208789 -0.42075999 -0.51786171 ... 0. ... 0. ...]
loaded	float64	(1000, 25000)	[-0.81999071 -0.81014303 -0.75184401 ... -0.0181839... 0 ...]
loaded	float64	1	genres/jazz/jazz.00062.au
loss	float32	(1000, 10)	[1. 0. 0. ... 0. 0. 0.] [1. 0. 0. ... 0. 0. 0.]
loss	float64	1	1.5005969190597535

Figure 6.16: Fungsi Evaluate

```
In [21]: model.predict(test_input[:1])
Out[21]:
array([[1.4394021e-02, 1.3317568e-06, 1.1783892e-02, 8.4673949e-03,
       9.1628902e-02, 1.5559867e-02, 8.3826762e-06, 3.7754746e-03,
       8.2906008e-01, 2.5320653e-02]], dtype=float32)
```

Figure 6.17: Fungsi Predict

6.1.3 Penanganan Eror

1. ScreenShoot Error dapat dilihat pada gambar 6.18.

```
In [1]: import librosa
....: import librosa.feature
....: import librosa.display
....: import glob
....: import numpy as np
....: import matplotlib.pyplot as plt
....: from keras.models import Sequential
....: from keras.layers import Dense, Activation
....: from keras.utils.np_utils import to_categorical
Traceback (most recent call last):

File "<ipython-input-1-736e292567e0>", line 7, in <module>
    from keras.models import Sequential

ModuleNotFoundError: No module named 'keras.models'
```

Figure 6.18: Eror

2. Tuliskan kode eror dan jenis errornya.

ModuleNotFoundError: No Module named 'keras.models'

3. Solusi pemecahan masalah error tersebut.

```
# Cara penanganannya adalah install modul keras  
# dengan perintah seperti berikut
```

```
#dengan pip  
pip install keras  
#dengan conda  
conda install keras
```

Chapter 7

Discussion

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

7.1 Ahmad Syafrizal Huda/1164062

7.1.1 Teori

1. Jelaskan kenapa file teks harus di lakukan tokenizer.

Tokenizer adalah untuk membuat vektor dari teks. Dan mengapa harus dilakukan tokenizer pada file teks? itu karena dengan memfungsiakan tokenizer, teks dapat divektorkan. Sehingga teks yang telah telah divektorkan tersebut dapat terbaca pada Machine Learning. Proses tokenizer itu hanya memenggal kata-kata yang ada didalam suatu frasa atau kalimat yang terdapat didalam suatu text dataset. Ilustrasi gambar dapat dilihat pada gambar 7.1.

2. Jelaskan konsep dasar K Fold Cross Validation pada dataset komentar Youtube pada kode listing 7.1.

Listing 7.1: K Fold Cross Validation

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

StartifiedKFold berisikan presentasi sampel untuk setiap kelas. Dimana dalam ilustrasi ini sampel dibagi menjadi 5 dalam setiap class nya. Kemudian sampel tadi akan dimasukan kedalam class dari dataset youtube tadi. Contoh dapat dilihat pada gambar 7.2.

3. Jelaskan apa maksudnya kode program for train, test in splits.

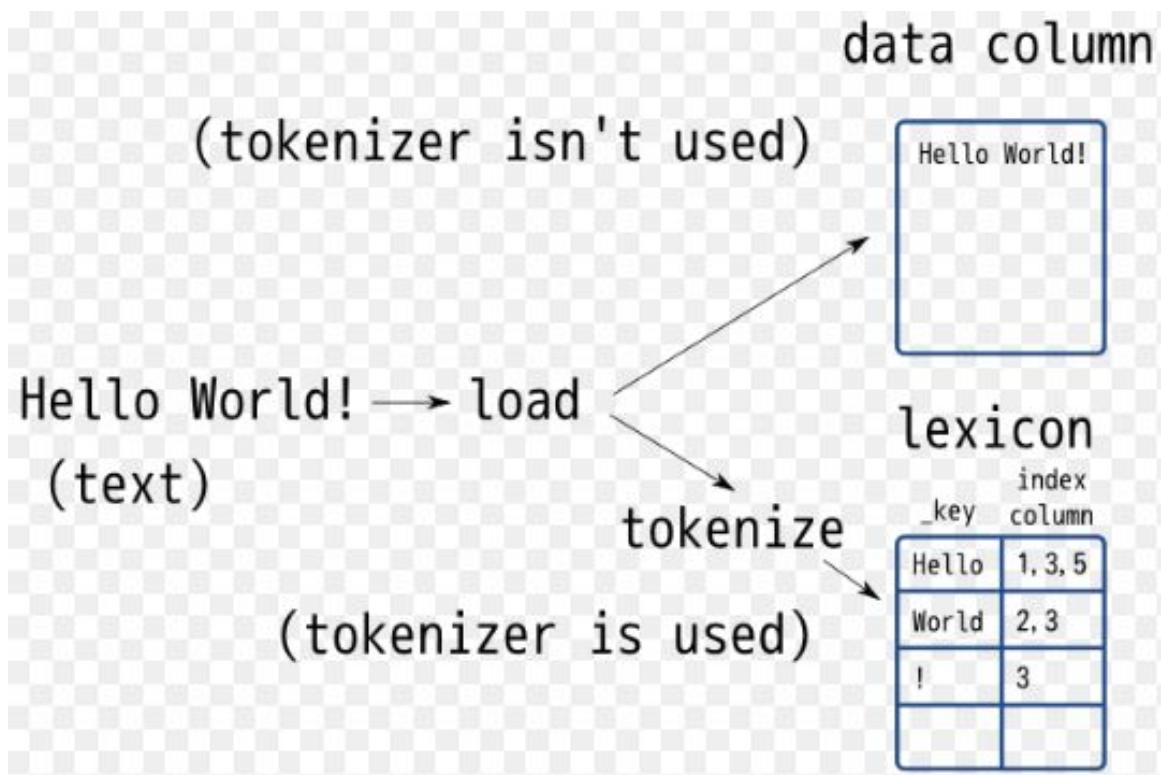


Figure 7.1: Tokenizer

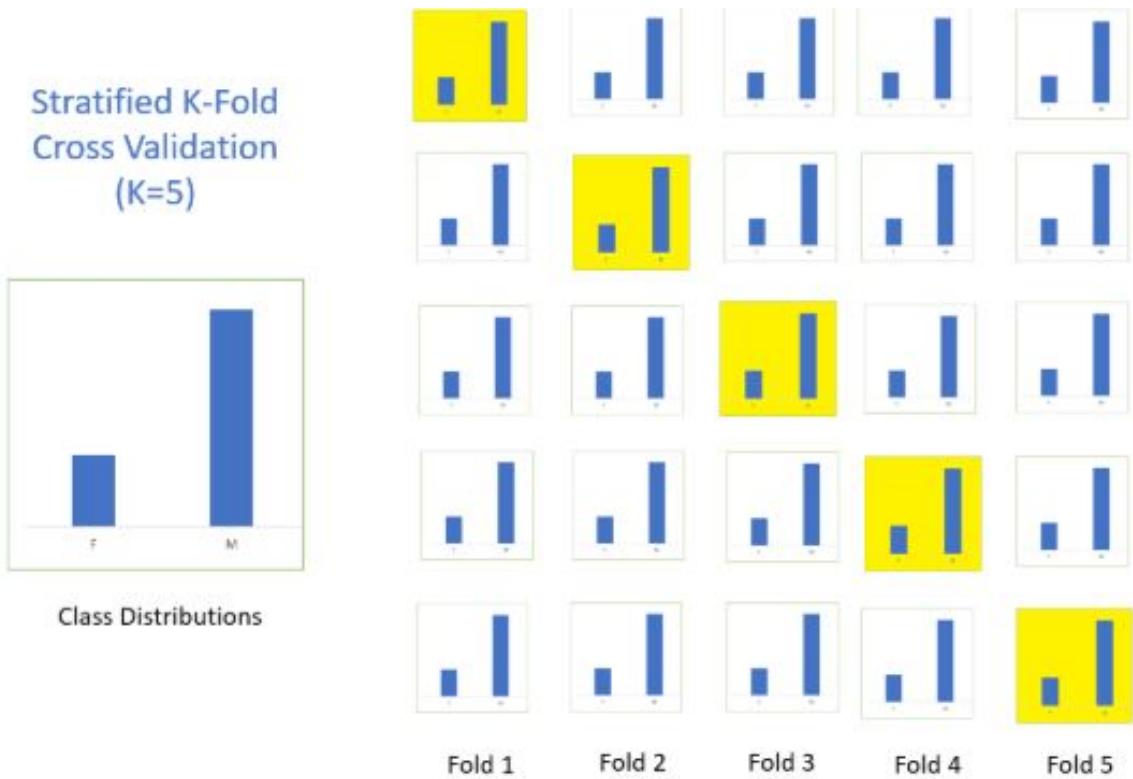


Figure 7.2: StartifedKFold

Maksudnya yaitu untuk menguji apakah setiap data pada dataset sudah di split dan tidak terjadi penumpukan. Yang dimana maksudnya di setiap class tidak akan muncul id yang sama. Ilustrasinya misalkan kita memiliki 4 baju dengan model yang berbeda. Kemudian kita bagikan kedua anak, tentunya setiap anak yang menerima baju tidak memiliki baju yang sama modelnya.

4. Jelaskan apa maksudnya kode program $train_content = d['CONTENT'].iloc[train_idx]$ dan $test_content = d['CONTENT'].iloc[test_idx]$.

Maksudnya yaitu mengambil data pada kolom atau index CONTENT yang merupakan bagian dari train_idx dan test_idx. Ilustrasinya, ketika data telah diubah menjadi train dan test maka kita dapat memilihnya untuk ditampilkan pada kolom yang diinginkan.

5. Jelaskan apa maksud dari fungsi $tokenizer = Tokenizer(num_words=2000)$ dan $tokenizer.fit_on_texts(train_content)$.

Dimana variabel tokenizer akan melakukan vektorisasi kata menggunakan fungsi Tokenizer yang dimana jumlah kata yang ingin diubah kedalam bentuk token adalah 2000 kata. Dan untuk $tokenizer.fit_on_texts(train_content)$ maksudnya kita akan melakukan fit tokenizer hanya untuk dat trainnya saja tidak dengan data test nya untuk kolom CONTENT. Ilustrasinya, Jadi, jika Anda memberikannya sesuatu seperti, "Kucing itu duduk di atas tikar." Ini akan membuat kamus s.t. word_index ["the"] = 0; word_index ["cat"] = 1 itu adalah kata -*i* kamus indeks sehingga setiap kata mendapat nilai integer yang unik.

6. Jelaskan apa maksud dari fungsi $d_train_inputs = tokenizer.texts_to_matrix(train_content, mode='tfidf')$ dan $d_test_inputs = tokenizer.texts_to_matrix(test_content, mode='tfidf')$.

Maksudnya yaitu untuk variabel d_train_inputs akan melakukan tokenizer dari bentuk teks ke matrix dari data train_content dengan mode matriksnya yaitu tfidf begitu juga dengan variabel d_test_inputs untuk data test. Contoh dapat dilihat pada gambar 7.3.

7. Jelaskan apa maksud dari fungsi $d_train_inputs = d_train_inputs/np.amax(np.absolute(d_train_inputs))$ dan $d_test_inputs = d_test_inputs/np.amax(np.absolute(d_test_inputs))$.

Fungsi tersebut akan bagi matrix tfidf tadi dengan amax yaitu mengembalikan maksimum array atau maksimum sepanjang sumbu. Yang hasilnya akan dimasukan kedalam variabel d_train_inputs untuk data train dan d_test_inputs

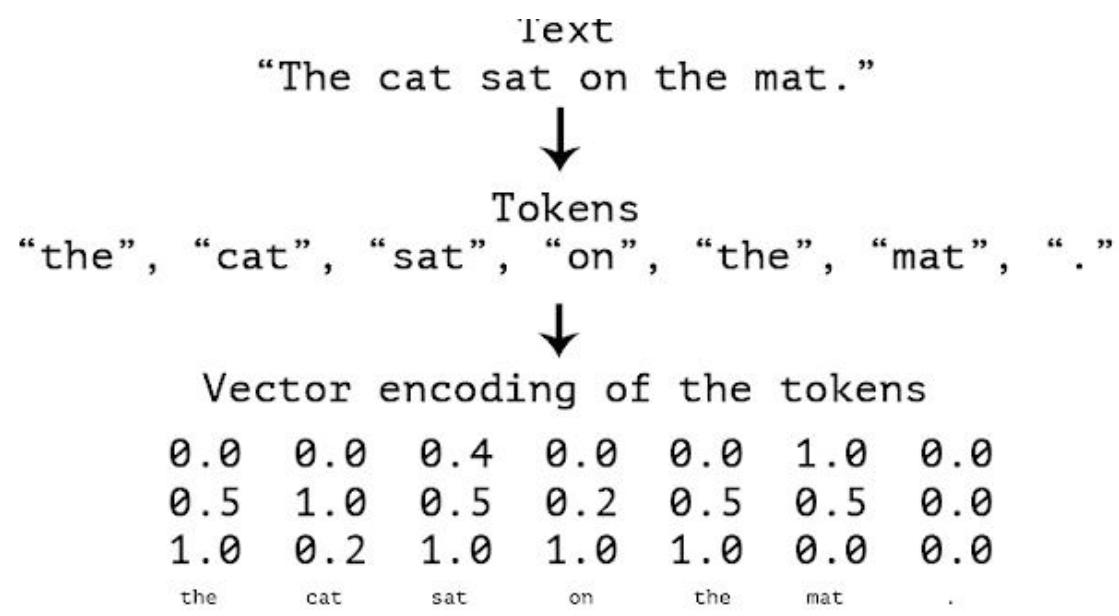


Figure 7.3: Fungsi Tokenizer

untuk data test dengan nominal absolut atau tanpa ada bilangan negatif dan koma. Contoh dapat dilihat pada gambar 7.4.

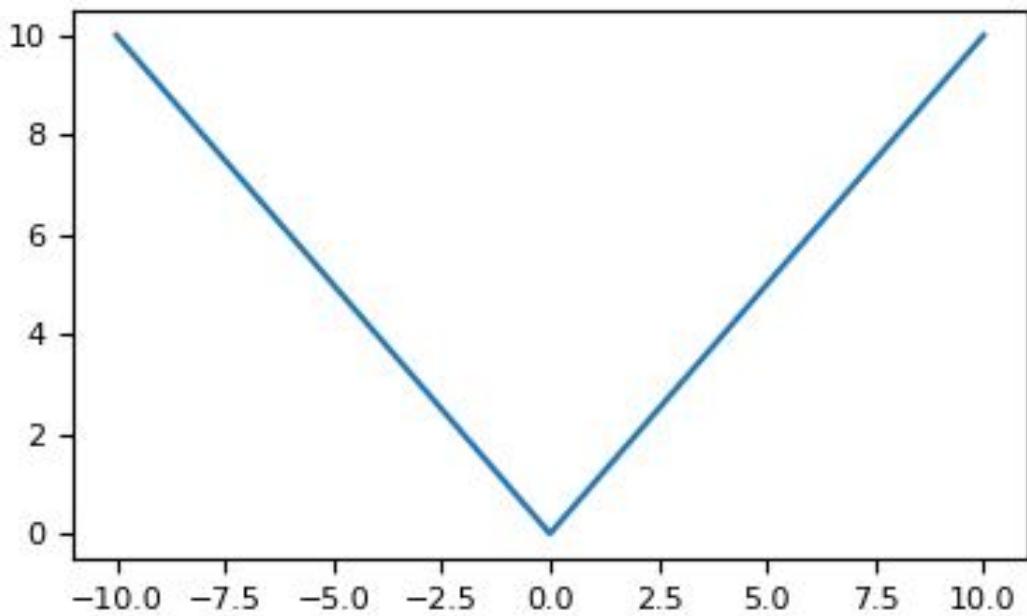


Figure 7.4: Matrix TFID

8. Jelaskan apa maksud fungsi dari `d train outputs = np utils.to categorical(d['CLASS'].iloc[train]`

dan d test outputs = np utils.to categorical(d['CLASS'].iloc[test idx]) dalam kode program.

maksud dari fungsi tersebut yaitu untuk merubah nilai vektor yang ada pada atribut class menjadi bentuk matrix dengan pengurutan berdasarkan data index training dan testing. Contoh dapat dilihat pada gambar 7.5

```
# Consider an array of 5 labels out of a set of 3 classes {0, 1, 2}:
> labels
array([0, 2, 1, 2, 0])
# `to_categorical` converts this into a matrix with as many
# columns as there are classes. The number of rows
# stays the same.
> to_categorical(labels)
array([[ 1.,  0.,  0.],
       [ 0.,  0.,  1.],
       [ 0.,  1.,  0.],
       [ 0.,  0.,  1.],
       [ 1.,  0.,  0.]], dtype=float32)

>>> type(df.iloc[0])
<class 'pandas.core.series.Series'>
>>> df.iloc[0]
a    1
b    2
c    3
d    4
Name: 0, dtype: int64
```

Figure 7.5: Matrix Training dan Testing

9. Jelaskan apa maksud dari fungsi di listing 7.2. Gambarkan ilustrasi Neural Network nya dari model kode tersebut.

Listing 7.2: Neural Network

```
model=Sequential()
model.add(Dense(512, input_shape=(2000,)))
model.add(Activation('relu'))
model.add(Dropout(0.5))
model.add(Dense(2))
model.add(Activation('softmax'))
```

10. Jelaskan apa maksud dari fungsi di listing 7.3 dengan parameter tersebut.

Listing 7.3: Model Compile Metric

```
model . compile(loss='categorical_crossentropy', optimizer='adamax',  
metrics=[ 'accuracy' ])
```

11. Jelaskan apa itu Deep Learning.

Deep Learning merupakan cabang dari Machine Learning atau bagian keluarga yang lebih luas dari method machine learning berdasarkan pada representasi data pembelajaran dan memiliki konsep serupa, tapi dilakukan dengan metode yang lebih cerdas. Deep Learning menggunakan Deep Neural Network dalam menyelesaikan suatu masalah yang terjadi pada Machine Learning.

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

- Deep Neural Network atau DNN merupakan algoritma yang berbasis neural network yang digunakan untuk mengambil keputusan.
- Yang membedakan Deep Learning dengan Deep Neural Network (DNN) adalah DNN merupakan algoritma yang digunakan pada Deep Learning, sedangkan Deep Learning merupakan model yang menggunakan algoritma DNN.

13. Jelaskan dengan ilustrasi gambar buatan sendiri(langkah per langkah) bagaimana perhitungan algoritma konvolusi dengan ukuran stride (NPM mod3+1) x (NPM mod3+1) yang terdapat max pooling.

Konvolusi pada gambar dilakukan dalam image processing untuk menerapkan operator yang memiliki nilai output dari piksel gambar yang berasal dari kombinasi linear nilai input piksel, semakin nilai piksel tersebut maka kualitas gambar bisa semakin bagus. Contoh ilustrasi gambar dapat dilihat pada gambar 7.6 dan 7.7.

7.1.2 Praktek Program

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

Berikut adalah kode program yang digunakan :

Listing 7.4: Kode Program 1

```
import csv  
from PIL import Image as pil_image  
import keras.preprocessing.image
```

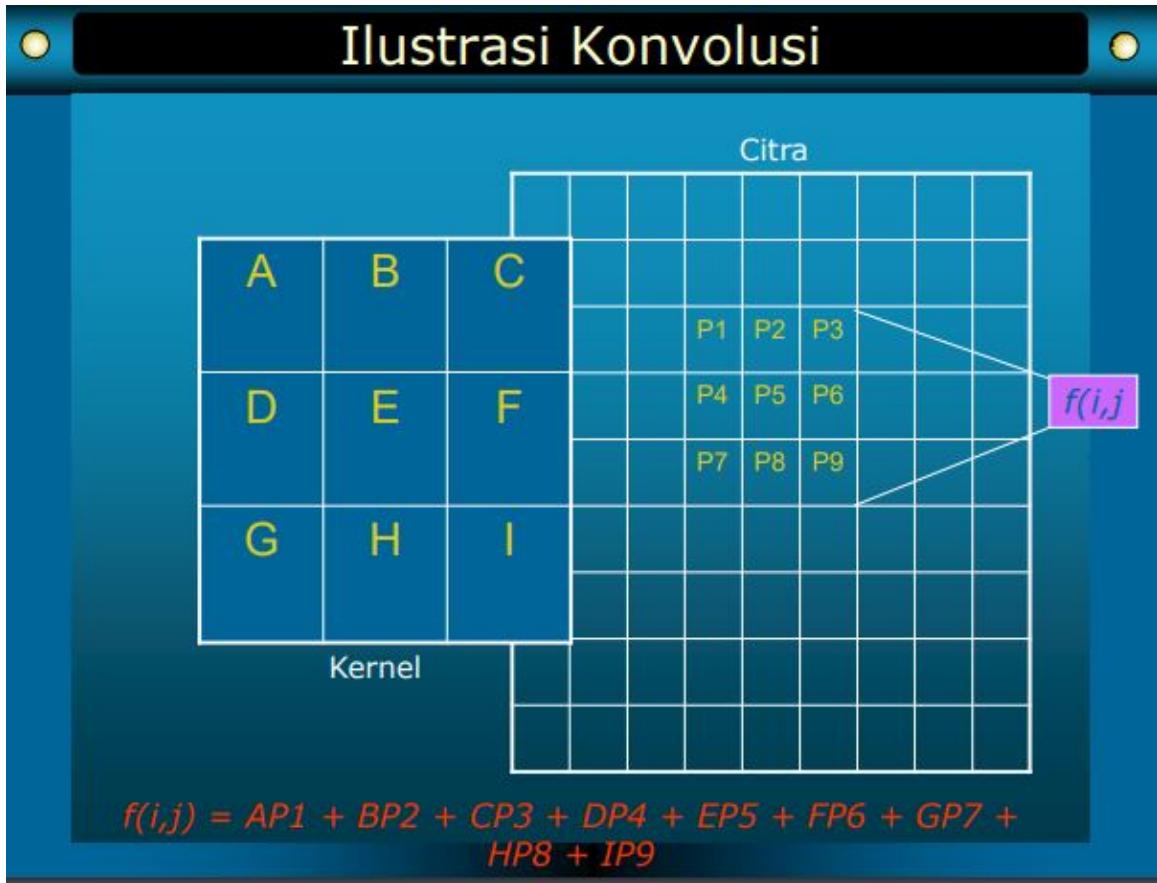


Figure 7.6: Konvolusi

Npm saya adalah 1164062 dan hasil dari $(NPM \bmod 3+1)=3$, maka saya menggunakan matrik kernel berukuran 3×3 . Misal $f(x,y)$ yang digunakan berukuran 4×4 dan kernel atau mask berukuran 3×3 dengan data masing-masing sebagai berikut:

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

Penyelesaian pada algoritma perhitungan konvolusi antara $f(x,y)$ dan $g(x,y)$ yaitu $f(x,y)xg(x,y)$. cara menghitungnya dari ujung kiri $f(x,y)$ ke $g(x,y)$ jika sudah geser pada kernel yang berbeda selanjutnya seperti berikut ini:

1. $(2 \times 1) + (4 \times 2) + (6 \times 1) + (1 \times 2) + (3 \times 1) + (5 \times 2) + (1 \times 3) + (2 \times 2) + (3 \times 3) = 47$
2. $(4 \times 1) + (6 \times 2) + (8 \times 1) + (3 \times 2) + (5 \times 1) + (7 \times 2) + (2 \times 3) + (3 \times 2) + (4 \times 3) = 73$
3. $(1 \times 1) + (3 \times 2) + (5 \times 1) + (1 \times 2) + (2 \times 1) + (3 \times 2) + (4 \times 3) + (3 \times 2) + (2 \times 3) = 46$
4. $(3 \times 1) + (5 \times 2) + (7 \times 1) + (2 \times 2) + (3 \times 1) + (4 \times 2) + (3 \times 3) + (2 \times 2) + (1 \times 3) = 51$

Hasil dari perhitungan diatas menghasilkan matrik kernel dengan 2×2 seperti berikut ini:

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

Figure 7.7: Algoritma Perhitungan Konvolusi

Dari kode listing pada kode program 1, dapat dijelaskan seperti berikut :

- Baris 1 : Melakukan pengimportan file csv
- Baris 2 : Melakukan pemanggilan atau memasukkan module image sebagai pil_image dari library PIL
- Baris 3 : Melakukan pengimportan fungsi keras.preprocessing.image

Sehingga dari kode program tersebut bila dijalankan, maka menghasilkan seperti pada gambar 7.8.

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

Figure 7.8: In[1]

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

Berikut adalah kode program yang digunakan :

Listing 7.5: Kode Program 2

```
imgs = []
classes = []
with open('Chapter04/hasy-data-labels.csv') as csvfile:
    csvreader = csv.reader(csvfile)
    i = 0
    for row in csvreader:
        if i > 0:
            img = keras.preprocessing.image.img_to_array(pil_image.
# neuron activation functions behave best when input va
# so we rescale each pixel value to be in the range 0.0
            img /= 255.0
            imgs.append((row[0], row[2], img))
            classes.append(row[2])
        i += 1
```

Dari kode listing pada kode program 2, dapat dijelaskan seperti berikut :

- Baris 1 : Membuat variabel imgs tanpa ada parameter di dalamnya

- Baris 2 : Membuat variabel classes tanpa ada parameter didalamnya
- Baris 3 : Membuka file csv dari HASYv2/hasy-data-labels.csv sebagai csv-file
- Baris 4 : Membuat variabel csvreader yang difungsikan untuk membaca dari file csv yang dimasukkan
- Baris 5 : Membuat variabel i dengan parameter 0 atau nilai 0
- Baris 6 : Digunakan untuk melakukan eksekusi baris dari pembacaan csv
- Baris 7 : Mengaplikasikan atau menggunakan perintah "if" dengan variabel i lebih besar dari 0, yang selanjutnya akan dilanjutkan ke perintah berikutnya
- Baris 8 : Membuat variabel img yang berfungsi untuk mengubah image atau gambar menjadi bentuk array (bilangan) dari file HASYv2 yang dibuka dengan row berparameter 0.
- Baris 9 : Membuat variabel img dengan nilai bukan sama dengan 255.0
- Baris 10 : Mendefinisikan fungsi imgs.append yang digunakan untuk melakukan proses peng gabungan data dengan file lain atau dataset lain yang telah ditentukan dengan 3 parameter yaitu row[0], row[2] dan variabel img.
- Baris 11 : Mendefinisikan fungsi append dari variabel classes dengan menggunakan parameter row[2].
- Baris 12 : Mengartikan $i = i + 1$ yang dimana nilai sari variabel i akan ditambah 1 sehingga akan bernilai 1.

Sehingga dari kode program tersebut bila dijalankan, maka menghasilkan seperti pada gambar 7.9.

Name	Type	Size	Value
classes	list	168233	['A', 'A', ...]
i	int	1	168234
img	float32	(32, 32, 3)	[[[1. 1. 1.], [1. 1. 1.]]]
imgs	list	168233	[('hasy-data/v2-00000.png', 'A', Numpy array), ('hasy-data/v2-00001.pn...']
row	list	4	['hasy-data/v2-168232.png', '1400', '\guillemotleft', '16925']

Figure 7.9: In[2]

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

Berikut adalah kode program yang digunakan :

Listing 7.6: Kode Program 3

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

Dari kode listing pada kode program 3, dapat dijelaskan seperti berikut :

- Baris 1 : Memanggil dan menggunakan module random
- Baris 2 : Melakukan pengocokan menggunakan module random pada parameter variabel imgs
- Baris 3 : Membagi index data kedalam bentuk integer dengan mengalikan 0,8 dan len yang berfungsi mengembalikan jumlah item dalam datanya dari variabel imgs
- Baris 4 : Membuat variabel train yang digunakan untuk mengeksekusi imgs serta pemecahan index awal pada data untuk digunakan sebagai data training
- Baris 5 : Membuat variabel test yang digunakan untuk mengeksekusi imgs serta pemecahan index akhir pada data untuk digunakan sebagai data testing

Sehingga dari kode program tersebut bila dijalankan, maka menghasilkan seperti pada gambar 7.10.

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

Figure 7.10: In[3]

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

Berikut adalah kode program yang digunakan :

Listing 7.7: Kode Program 4

```
import numpy as np

train_input = np.asarray(list(map(lambda row: row[2], train)))
test_input = np.asarray(list(map(lambda row: row[2], test)))

train_output = np.asarray(list(map(lambda row: row[1], train)))
test_output = np.asarray(list(map(lambda row: row[1], test)))
```

Dari kode listing pada kode program 4, dapat dijelaskan seperti berikut :

- Baris 1 : Melakukan import library numpy sebagai np
- Baris 2 : Membuat variabel train_input untuk mengubah inputan menjadi array menggunakan fungsi np.asarray dan fungsi list untuk mengkoleksi data yang dipilih serta data dapat diubah. Dan didalamnya melakukan penerapan fungsi map yang berfungsi untuk mengembalikan iterator dari data yang digunakan dan fungsi lamda pada row berparameter [2] difungsikan untuk membuat objek menjadi lebih kecil sehingga mudah dieksekusi dari variabel train.
- Baris 3 : Membuat variabel test_input untuk mengubah inputan menjadi array menggunakan fungsi np.asarray dan fungsi list untuk mengkoleksi data yang dipilih serta data dapat diubah. Dan didalamnya melakukan penerapan fungsi map yang berfungsi untuk mengembalikan iterator dari data yang digunakan dan fungsi lamda pada row berparameter [2] difungsikan untuk membuat objek menjadi lebih kecil sehingga mudah dieksekusi dari variabel test.
- Baris 4 : Membuat variabel train_input untuk mengubah inputan menjadi array menggunakan fungsi np.asarray dan fungsi list untuk mengkoleksi data yang dipilih serta data dapat diubah. Dan didalamnya melakukan penerapan fungsi map yang berfungsi untuk mengembalikan iterator dari data yang digunakan dan fungsi lamda pada row berparameter [1] difungsikan untuk membuat objek menjadi lebih kecil sehingga mudah dieksekusi dari variabel train.
- Baris 5 : Membuat variabel test_input untuk mengubah inputan menjadi array menggunakan fungsi np.asarray dan fungsi list untuk mengkoleksi data yang dipilih serta data dapat diubah. Dan didalamnya melakukan penerapan fungsi map yang berfungsi untuk mengembalikan iterator dari

data yang digunakan dan fungsi lamda pada row berparameter [1] difungsikan untuk membuat objek menjadi lebih kecil sehingga mudah dieksekusi dari variabel test.

Sehingga dari kode program tersebut bila dijalankan, maka menghasilkan seperti pada gambar 7.11.

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

Figure 7.11: In[4]

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

Berikut adalah kode program yang digunakan :

Listing 7.8: Kode Program 5

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

Dari kode listing pada kode program 5, dapat dijelaskan seperti berikut :

- Baris 1 : Menggunakan fungsi LabelEncoder dari sklearn.preprocessing yang berfungsi untuk menormalkan label dimana label encoder hanya didefinisikan dengan nilai antara 0 dan -1.
- Baris 2 : Menggunakan fungsi OneHotEncoder dari sklearn.preprocessing yang berfungsi untuk mendefinisikan fitur input yang dimana mengambil nilai dalam kisaran [0, nilai maksimal].

Sehingga dari kode program tersebut bila dijalankan, maka menghasilkan seperti pada gambar 7.12.

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

Berikut adalah kode program yang digunakan :

Listing 7.9: Kode Program 6

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

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

Figure 7.12: In[5]

Dari kode listing pada kode program 6, dapat dijelaskan seperti berikut :

- Baris 1 : Membuat variabel label_encoder dengan penerapan modul / fungsi LabelEncoder tanpa parameter
- Baris 2 : Membuat variabel integer_encoded dengan penerapan fungsi label_encoder.fit_transform yang berfungsi untuk melakukan ekstrasi fitur object dari variabel classes yang akan mengembalikan beberapa data yang diubah kembali.

Sehingga dari kode program tersebut bila dijalankan, maka menghasilkan seperti pada gambar 7.13.

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

Figure 7.13: In[6]

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

Berikut adalah kode program yang digunakan :

Listing 7.10: Kode Program 7

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

Dari kode listing pada kode program 7, dapat dijelaskan seperti berikut :

- Variabel onehot_encoder akan memanggil fungsi OneHotEncoder dimana tidak berisikan matriks sparse.
- Pada variabel integer_encoded akan diubah bentuknya dimana setiap nilai integer akan direpresentasikan sebagai vektor binari dengan nilai 0 kecuali index dari integer tersebut ditandai dengan 1.
- Melakukan fit untuk one hot encoder kedalam integer_encoder.

Sehingga dari kode program tersebut bila dijalankan, maka menghasilkan seperti pada gambar 7.14.

```
In [8]: onehot_encoder = OneHotEncoder(sparse=False)
.... integer_encoded = integer_encoded.reshape(len(integer_encoded), 1)
.... onehot_encoder.fit(integer_encoded)
C:\Users\HUDA\Anaconda3\lib\site-packages\sklearn\preprocessing\_encoders.py:368:
FutureWarning: The handling of integer data will change in version 0.22. Currently, the
categories are determined based on the range [0, max(values)], while in the future they will
be determined based on the unique values.
If you want the future behaviour and silence this warning, you can specify
"categories='auto'".
In case you used a LabelEncoder before this OneHotEncoder to convert the categories to
integers, then you can now use the OneHotEncoder directly.
    warnings.warn(msg, FutureWarning)
Out[8]:
OneHotEncoder(categorical_features=None, categories=None,
               dtype=<class 'numpy.float64'>, handle_unknown='error',
               n_values=None, sparse=False)
```

Figure 7.14: In[7]

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

Berikut adalah kode program yang digunakan :

Listing 7.11: Kode Program 8

```
train_output_int = label_encoder.transform(train_output)
train_output = onehot_encoder.transform(train_output_int.reshape(len(train_output_int), 1))
test_output_int = label_encoder.transform(test_output)
test_output = onehot_encoder.transform(test_output_int.reshape(len(test_output_int), 1))

num_classes = len(label_encoder.classes_)
print("Number of classes: %d" % num_classes)
```

Dari kode listing pada kode program 8, dapat dijelaskan seperti berikut :

- Variabel train_output_int akan mengubah data dari train_output menjadi LabeEncoder
- Dimana pada train_output setelah diubah labelnya menjadi integer dilakukan one hot encoding diambil dari train_output_int dan menggunakan .reshape untuk memberikan bentuk baru ke array tanpa mengubah datanya dengan keterangan jika index dari integer tersebut ditandai dengan 1 dan sisanya yang bukan nol.
- Variabel test_output_int akan mengubah data dari test_output menjadi LabeEncoder

- Dimana pada train_output setelah diubah labelnya menjadi integer dilakukan one hot encoding diambil dari test_output_int dan menggunakan .reshape untuk memberikan bentuk baru ke array tanpa mengubah datanya dengan keterangan jika index dari integer tersebut ditandai dengan 1 dan sisanya yang bukan nol.
- Variabel num_classes akan menampilkan jumlah data dari classes yang telah dilakukan label encoder
- Menampilkan tulisan "Number of classes : %d" dmana mengembalikan nilai integer dari num_classes.

Sehingga dari kode program tersebut bila dijalankan, maka menghasilkan seperti pada gambar 7.15.



Figure 7.15: In[8]

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

Berikut adalah kode program yang digunakan :

Listing 7.12: Kode Program 9

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

Dari kode listing pada kode program 9, dapat dijelaskan seperti berikut :

- Impor Sequential dari model pada librari Keras.
- Impor Dense, Dropout, Flatten dari modul Layers pada librari Keras.
- Impor Conv2D, MaxPooling2D dari modul Layers pada librari Keras.

Sehingga dari kode program tersebut bila dijalankan, maka menghasilkan seperti pada gambar 7.16.

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

Berikut adalah kode program yang digunakan :

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

Figure 7.16: In[9]

Listing 7.13: Kode Program 10

```
model = Sequential()
model.add(Conv2D(32, kernel_size=(3, 3), activation='relu',
                 input_shape=np.shape(train_input[0])))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Conv2D(32, (3, 3), activation='relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Flatten())
model.add(Dense(1024, activation='tanh'))
model.add(Dropout(0.5))
model.add(Dense(num_classes, activation='softmax'))

model.compile(loss='categorical_crossentropy', optimizer='adam',
               metrics=['accuracy'])

print(model.summary())
```

Dari kode listing pada kode program 10, dapat dijelaskan seperti berikut :

- Melakukan pemodelan Sequential.
- Menambahkan Konvolusi 2D dengan 32 filter konvolusi masing-masing berukuran 3x3 dengan algoritam activation relu dengan data dari train_input mulai dari baris nol.
- Menambahkan Max Pooling dengan matriks 2x2.
- Dilakukan lagi penambahan Konvolusi 2D dengan 32 filter konvolusi masing-masing berukuran 3x3 dengan algoritam activation relu.
- Menambahkan lagi Max Pooling dengan matriks 2x2.
- Mendefinisikan inputan dengan 1024 neuron dan menggunakan algoritma tanh untuk activationnya.
- Dropout terdiri dari pengaturan secara acak tingkat pecahan unit input ke 0 pada setiap pembaruan selama waktu pelatihan, yang membantu mencegah overfitting sebesar 50% .

- Untuk output layer menggunakan data dari variabel num_classes dengan fungsi activationnya softmax.
- Mengonfigurasi proses pembelajaran, yang dilakukan melalui metode compile, sebelum melatih suatu model.
- Menampilkan atau mencetak representasi ringkasan model yang telah dibuat.

Sehingga dari kode program tersebut bila dijalankan, maka menghasilkan seperti pada gambar 7.17.

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

Figure 7.17: In[10]

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

Berikut adalah kode program yang digunakan :

Listing 7.14: Kode Program 11

```
import keras.callbacks
tensorboard = keras.callbacks.TensorBoard(log_dir='./logs/mnist-sty
```

Dari kode listing pada kode program 11, dapat dijelaskan seperti berikut :

- Impor Modul Callbacks dari Librari Keras.
- Variabel callback mendefinisikan Callback ini untuk menulis log untuk TensorBoard, yang memungkinkan Anda untuk memvisualisasikan grafik dinamis dari pelatihan dan metrik pengujian Anda, serta histogram aktivasi untuk berbagai lapisan dalam model Anda.

Sehingga dari kode program tersebut bila dijalankan, maka menghasilkan seperti pada gambar 7.18.

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

Figure 7.18: In[11]

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

Berikut adalah kode program yang digunakan :

Listing 7.15: Kode Program 12

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

Dari kode listing pada kode program 12, dapat dijelaskan seperti berikut :

- Melakukan fit model dengan 32 ukuran subset dari sampel pelatihan Anda
- Epoch sebanyak 10 kali
- Vebrose=2 maksudnya menampilkan nomor dari epoch yang sedang berjalan atau yang sudah dijalankan.
- Validasi plit sebanyak 20% sebagai fraksi data pelatihan untuk digunakan sebagai data validasi.

- Menggunakan TensorBoard sebagai callback untuk diterapkan selama pelatihan dan validasi.
- Variabel score mengembalikan nilai evaluate untuk menampilkan data loss dan data accuracy dari test
- Menampilkan data loss dengan menghitung jumlah kemunculan nol .
- Menampilkan data accuracy dengan menghitung jumlah kemunculan 1.

Sehingga dari kode program tersebut bila dijalankan, maka menghasilkan seperti pada gambar 7.19.

```
Train on 107668 samples, validate on 26918 samples
Epoch 1/10
- 693s - loss: 1.5615 - acc: 0.6233 - val_loss: 0.9945 - val_acc: 0.7260
Epoch 2/10
- 360s - loss: 0.9810 - acc: 0.7283 - val_loss: 0.8854 - val_acc: 0.7521
Epoch 3/10
- 355s - loss: 0.8676 - acc: 0.7516 - val_loss: 0.8852 - val_acc: 0.7550
Epoch 4/10
- 351s - loss: 0.7921 - acc: 0.7670 - val_loss: 0.8286 - val_acc: 0.7717
Epoch 5/10
- 350s - loss: 0.7472 - acc: 0.7770 - val_loss: 0.8424 - val_acc: 0.7687
Epoch 6/10
- 352s - loss: 0.7054 - acc: 0.7854 - val_loss: 0.8333 - val_acc: 0.7687
Epoch 7/10
- 351s - loss: 0.6716 - acc: 0.7929 - val_loss: 0.8297 - val_acc: 0.7679
Epoch 8/10
- 352s - loss: 0.6432 - acc: 0.7987 - val_loss: 0.8535 - val_acc: 0.7706
Epoch 9/10
- 368s - loss: 0.6197 - acc: 0.8047 - val_loss: 0.8778 - val_acc: 0.7635
Epoch 10/10
- 374s - loss: 0.5991 - acc: 0.8094 - val_loss: 0.8634 - val_acc: 0.7684
Test loss: 0.8603214174495762
Test accuracy: 0.767527565614718
```

Figure 7.19: In[12]

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

Berikut adalah kode program yang digunakan :

Listing 7.16: Kode Program 13

```
import time

results = []
for conv2d_count in [1, 2]:
```

```

for dense_size in [128, 256, 512, 1024, 2048]:
    for dropout in [0.0, 0.25, 0.50, 0.75]:
        model = Sequential()
        for i in range(conv2d_count):
            if i == 0:
                model.add(Conv2D(32, kernel_size=(3, 3), activation='relu'))
            else:
                model.add(Conv2D(32, kernel_size=(3, 3), activation='relu'))
                model.add(MaxPooling2D(pool_size=(2, 2)))
        model.add(Flatten())
        model.add(Dense(dense_size, activation='tanh'))
        if dropout > 0.0:
            model.add(Dropout(dropout))
        model.add(Dense(num_classes, activation='softmax'))

model.compile(loss='categorical_crossentropy', optimizer='adam')

log_dir = './logs/conv2d_%d-dense_%d-dropout_%.2f' % (conv2d_count, dense_size, dropout)
tensorboard = keras.callbacks.TensorBoard(log_dir=log_dir)

start = time.time()
model.fit(train_input, train_output, batch_size=32, epochs=10,
          verbose=0, validation_split=0.2, callbacks=[tensorboard])
score = model.evaluate(test_input, test_output, verbose=0)
end = time.time()
elapsed = end - start
print("Conv2D count: %d, Dense size: %d, Dropout: %.2f" % (conv2d_count, dense_size, dropout))
results.append((conv2d_count, dense_size, dropout, score, elapsed))

```

Dari kode listing pada kode program 13, dapat dijelaskan seperti berikut :

- impor modul time dari python anaconda
- Variabel result berisikan array kosong.
- Menggunakan convolution 2D yang dimana akan memiliki 1 atau 2 layer.
- Mendefinisikan dense_size dengan ukuran 128, 256, 512, 1024, 2048
- Mendefinsikan drop_out dengan 0, 25%, 50%, dan 75%
- Melakukan pemodelan Sequential
- Jika ini adalah layer pertama, kita perlu memasukkan bentuk input.
- Kalau tidak kita hanya akan menambahkan layer.
- Kemudian, setelah menambahkan layer konvolusi, kita akan melakukan hal yang sama dengan max pooling.

- Lalu, kita akan meratakan atau flatten dan menambahkan dense size ukuran apa pun yang berasal dari dense_size. Dimana akan selalu menggunakan algoritma tanh
- Jika dropout digunakan, kita akan menambahkan layer dropout. Menyebut dropout ini berarti, katakanlah 50%, bahwa setiap kali ia memperbarui bobot setelah setiap batch, ada peluang 50% untuk setiap bobot yang tidak akan diperbarui
- menempatkan ini di antara dua lapisan padat untuk dihidupkan dari melindunginya dari overfitting.
- Lapisan terakhir akan selalu menjadi jumlah kelas karena itu harus, dan menggunakan softmax. Itu dikompilasi dengan cara yang sama.
- Atur direktori log yang berbeda untuk TensorBoard sehingga dapat membedakan konfigurasi yang berbeda.
- Variabel start akan memanggil modul time atau waktu
- Melakukan fit atau compile
- Melakukan scoring dengan .evaluate yang akan menampilkan data loss dan accuracy dari model
- end merupakan variabel untuk melihat waktu akhir pada saat pemodelan berhasil dilakukan.
- Menampilkan hasil dari run skrip diatas

Sehingga dari kode program tersebut bila dijalankan, maka menghasilkan seperti pada gambar 7.20.

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

Berikut adalah kode program yang digunakan :

Listing 7.17: Kode Program 14

```
model = Sequential()
model.add(Conv2D(32, kernel_size=(3, 3), activation='relu', input_size=(28, 28, 3))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Conv2D(32, (3, 3), activation='relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Flatten())
model.add(Dense(128, activation='tanh'))
model.add(Dropout(0.5))
model.add(Dense(num_classes, activation='softmax'))
```

```
2f, Accuracy: %.2f, Time: %d sec" % (conv2d_count, dense_size, dropout, score[0],
score[1], elapsed))
...:           results.append((conv2d_count, dense_size, dropout, score[0],
score[1], elapsed))
Conv2D count: 1, Dense size: 128, Dropout: 0.00 - Loss: 1.13, Accuracy: 0.74, Time: 1572
sec
Conv2D count: 1, Dense size: 128, Dropout: 0.25 - Loss: 0.91, Accuracy: 0.76, Time: 1632
sec
Conv2D count: 1, Dense size: 128, Dropout: 0.50 - Loss: 0.80, Accuracy: 0.78, Time: 1662
sec
Conv2D count: 1, Dense size: 128, Dropout: 0.75 - Loss: 0.80, Accuracy: 0.78, Time: 1735
sec
Conv2D count: 1, Dense size: 256, Dropout: 0.00 - Loss: 1.28, Accuracy: 0.74, Time: 2153
sec
Conv2D count: 1, Dense size: 256, Dropout: 0.25 - Loss: 1.08, Accuracy: 0.76, Time: 2212
sec
Conv2D count: 1, Dense size: 256, Dropout: 0.50 - Loss: 0.91, Accuracy: 0.78, Time: 2184
sec
Conv2D count: 1, Dense size: 256, Dropout: 0.75 - Loss: 0.78, Accuracy: 0.78, Time: 2184
sec
```

Figure 7.20: In[13]

```
model.compile(loss='categorical_crossentropy', optimizer='adam', m
print(model.summary())
```

Dari kode listing pada kode program 14, dapat dijelaskan seperti berikut :

- Melakukan pemodelan Sequential
- Untuk layer pertama, Menambahkan Convolutio 2D dengan dmensi 32, dan ukuran matriks 3x3 dengan function aktivasi yang digunakan yaitu relu dan menampilkan input_shape
- Dilakukan Max Pooling 2D dengan ukuran matriks 2x2
- Untuk layer kedua, melakukan Convolusi lagi dengan kriteria yang sama tanpa menambahkan input, ini dilakukan untuk mendapatkan data yang terbaik
- Flatten digunakan untuk meratakan inputan
- Menambahkan dense input sebanyak 128 neuron dengan menggunakan function aktivasi tanh.
- Dropout sebanyak 50% untuk menghindari overfitting
- Menambahkan dense pada model untuk output dimana layer ini akan menjadi jumlah dari class yang ada.
- Mengcompile model yang didefinisikan diatas
- Menampilkan ringkasan dari pemodelan yang dilakukan

Sehingga dari kode program tersebut bila dijalankan, maka menghasilkan seperti pada gambar 7.21.

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

Figure 7.21: In[14]

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

Berikut adalah kode program yang digunakan :

Listing 7.18: Kode Program 15

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

Dari kode listing pada kode program 15, dapat dijelaskan seperti berikut :

- Melakukan fit dengan join data train dan test agar dapat dilakukan pelatihan untuk jaringan pada semua data yang dimiliki.

Sehingga dari kode program tersebut bila dijalankan, maka menghasilkan seperti pada gambar 7.22.

```
In [15]: model.fit(np.concatenate((train_input, test_input)),
...:                 np.concatenate((train_output, test_output)),
...:                 batch_size=32, epochs=10, verbose=2)
Epoch 1/10
- 267s - loss: 1.7763 - acc: 0.5881
Epoch 2/10
- 249s - loss: 1.0767 - acc: 0.7061
Epoch 3/10
- 248s - loss: 0.9666 - acc: 0.7297
Epoch 4/10
- 249s - loss: 0.9096 - acc: 0.7411
Epoch 5/10
- 248s - loss: 0.8699 - acc: 0.7518
Epoch 6/10
- 253s - loss: 0.8428 - acc: 0.7551
Epoch 7/10
- 250s - loss: 0.8211 - acc: 0.7608
Epoch 8/10
- 247s - loss: 0.8050 - acc: 0.7639
Epoch 9/10
- 252s - loss: 0.7880 - acc: 0.7679
Epoch 10/10
- 251s - loss: 0.7751 - acc: 0.7697
Out[15]: <keras.callbacks.History at 0x22f13d79a20>
```

Figure 7.22: In[15]

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

Berikut adalah kode program yang digunakan :

Listing 7.19: Kode Program 16

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

Dari kode listing pada kode program 16, dapat dijelaskan seperti berikut :

- Menyimpan atau save model yang telah di latih dengan nama mathsymbols.model

Sehingga dari kode program tersebut bila dijalankan, maka menghasilkan seperti pada gambar 7.23.

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

A screenshot of a Jupyter Notebook interface. The code cell In [16] contains the command `model . save ("mathsymbols . model")`. Below the cell, a file download button is shown for a file named "mathsymbols . model" with a size of 2.443 KB. The file is categorized as a "MODEL File". The date of creation is listed as 13/04/2019 03.45.

Figure 7.23: In[16]

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

Berikut adalah kode program yang digunakan :

Listing 7.20: Kode Program 17

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

Dari kode listing pada kode program 17, dapat dijelaskan seperti berikut :

- Simpan label enkoder (untuk membalikkan one-hot encoder) dengan nama classes.npy

Sehingga dari kode program tersebut bila dijalankan, maka menghasilkan seperti pada gambar 7.24.

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

A screenshot of a Jupyter Notebook interface. The code cell In [17] contains the command `np . save ('classes . npy' , label_encoder . classes_)`. Below the cell, a file download button is shown for a file named "classes . npy" with a size of 28 KB. The file is categorized as a "NPY File". The date of creation is listed as 13/04/2019 03.45.

Figure 7.24: In[17]

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

Berikut adalah kode program yang digunakan :

Listing 7.21: Kode Program 18

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

Dari kode listing pada kode program 18, dapat dijelaskan seperti berikut :

- Impor models dari librari Keras
- Variabel model2 akan memanggil model yang telah disave tadi
- Menampilkan ringkasan dari hasil pemodelan

Sehingga dari kode program tersebut bila dijalankan, maka menghasilkan seperti pada gambar 7.25.

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


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


```

Figure 7.25: In[18]

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

Berikut adalah kode program yang digunakan :

Listing 7.22: Kode Program 19

```
label_encoder2 = LabelEncoder()
label_encoder2.classes_ = np.load('classes.npy')

def predict(img_path):
    newimg = keras.preprocessing.image.img_to_array(pil_image.open(
        newimg / 255.0

    # do the prediction
    prediction = model2.predict(newimg.reshape(1, 32, 32, 3))

    # figure out which output neuron had the highest score, and reverse it
    inverted = label_encoder2.inverse_transform([np.argmax(prediction)])
    print("Prediction: %s, confidence: %.2f" % (inverted[0], np.max(prediction)))
```

Dari kode listing pada kode program 19, dapat dijelaskan seperti berikut :

- Memanggil fungsi LabelEncoder
- Variabel label_encoder akan memanggil class yang disave sebelumnya.
- Function Predict akan mengubah gambar kedalam bentuk array
- Variabel prediction akan melakukan prediksi untuk model2 dengan reshape variabel newimg dengan bentukarray 4D.
- Variabel inverted akan mencari nilai tertinggi output dari hasil prediksi tadi
- Menampilkan hasil dari variabel prediction dan inverted

Sehingga dari kode program tersebut bila dijalankan, maka menghasilkan seperti pada gambar 7.26.

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

Berikut adalah kode program yang digunakan :

Listing 7.23: Kode Program 20

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

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

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

Figure 7.26: In[19]

- Melakukan prediksi dari pelatihan dari gambar v2-00010.png
- Melakukan prediksi dari pelatihan dari gambar v2-00500.png
- Melakukan prediksi dari pelatihan dari gambar v2-00700.png

Sehingga dari kode program tersebut bila dijalankan, maka menghasilkan seperti pada gambar 7.27.

```
In [20]: predict("HASYv2/hasy-data/v2-00010.png")
....:
....: predict("HASYv2/hasy-data/v2-00500.png")
....:
....: predict("HASYv2/hasy-data/v2-00700.png")
Prediction: A, confidence: 0.85
Prediction: \pi, confidence: 0.81
Prediction: \alpha, confidence: 0.90
```

Figure 7.27: In[20]

7.1.3 Penanganan Eror

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

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



Figure 7.28: Eror

3. Solusi pemecahan masalah error tersebut

- Tutup spyder yang sedang dijalankan
- Kemudian buka kembali spyder
- maka eror pada kernel tersebut akan hilang dan kembali stabil agar bisa mengakses konsol tersebut.

Chapter 8

Discussion

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

Chapter 9

Discussion

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

Chapter 10

Discussion

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

Chapter 11

Discussion

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

Chapter 12

Discussion

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

Chapter 13

Discussion

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

Chapter 14

Discussion

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

Appendix A

Form Penilaian Jurnal

gambar A.1 dan A.2 merupakan contoh bagaimana reviewer menilai jurnal kita.

NO	UNSUR	KETERANGAN	MAKS	KETERANGAN
1	Keefektifan Judul Artikel	Maksimal 12 (dua belas) kata dalam Bahasa Indonesia atau 10 (sepuluh) kata dalam Bahasa Inggris	2	a. Tidak lugas dan tidak ringkas (0) b. Kurang lugas dan kurang ringkas (1) c. Ringkas dan lugas (2)
2	Pencantuman Nama Penulis dan Lembaga Penulis		1	a. Tidak lengkap dan tidak konsisten (0) b. Lengkap tetapi tidak konsisten (0,5) c. Lengkap dan konsisten (1)
3	Abstrak	Dalam Bahasa Indonesia dan Bahasa Inggris yang baik, jumlah 150-200 kata. Isi terdiri dari latar belakang, metode, hasil, dan kesimpulan. Isi tertuang dengan kalimat yang jelas.	2	a. Tidak dalam Bahasa Indonesia dan Bahasa Inggris (0) b. Abstrak kurang jelas dan ringkas, atau hanya dalam Bahasa Inggris, atau dalam Bahasa Indonesia saja (1) c. Abstrak yang jelas dan ringkas dalam Bahasa Indonesia dan Bahasa Inggris (2)
4	Kata Kunci	Maksimal 5 kata kunci terpenting dalam paper	1	a. Tidak ada (0) b. Ada tetapi kurang mencerminkan konsep penting dalam artikel (0,5) c. Ada dan mencerminkan konsep penting dalam artikel (1)
5	Sistematika Pembahasan	Terdiri dari pendahuluan, tinjauan pustaka, metode penelitian, hasil dan pembahasan, kesimpulan dan saran, daftar pustaka	1	a. Tidak lengkap (0) b. Lengkap tetapi tidak sesuai sistem (0,5) c. Lengkap dan bersistem (1)
6	Pemanfaatan Instrumen Pendukung	Pemanfaatan Instrumen Pendukung seperti gambar dan tabel	1	a. Takermanfaatkan (0) b. Kurang informatif atau komplementer (0,5) c. Informatif dan komplementer (1)
7	Cara Pengacuan dan Pengutipan		1	a. Tidak baku (0) b. Kurang baku (0,5) c. Baku (1)
8	Penyusunan Daftar Pustaka	Penyusunan Daftar Pustaka	1	a. Tidak baku (0) b. Kurang baku (0,5) c. Baku (1)
9	Peristilahan dan Kebahasaan		2	a. Buruk (0) b. Baik (1) c. Cukup (2)
10	Makna Sumbangan bagi Kemajuan		4	a. Tidak ada (0) b. Kurang (1) c. Sedang (2) d. Cukup (3) e. Tinggi (4)

Figure A.1: Form nilai bagian 1.

11	Dampak Ilmiah		7	a. Tidak ada (0) b. Kurang (1) c. Sedang (3) d. Cukup (5) e. Besar (7)
12	Nisbah Sumber Acuan Primer berbanding Sumber lainnya	Sumber acuan yang langsung merujuk pada bidang ilmiah tertentu, sesuai topik penelitian dan sudah teruji.	3	a. < 40% (1) b. 40-80% (2) c. > 80% (3)
13	Derajat Kemutakhiran Pustaka Acuan	Derajat Kemutakhiran Pustaka Acuan	3	a. < 40% (1) b. 40-80% (2) c. > 80% (3)
14	Analisis dan Sintesis	Analisis dan Sintesis	4	a. Sedang (2) b. Cukup (3) c. Baik (4)
15	Penyimpulan	Sangat jelas relevasinya dengan latar belakang dan pembahasan, dirumuskan dengan singkat	3	a. Kurang (1) b. Cukup (2) c. Baik (3)
16	Unsur Plagiat		0	a. Tidak mengandung plagiat (0) b. Terdapat bagian-bagian yang merupakan plagiat (-5) c. Keseluruhannya merupakan plagiat (- 20)
TOTAL			36	
Catatan : Nilai minimal untuk diterima 25				

Figure A.2: form nilai bagian 2.

Appendix B

FAQ

M : Kalo Intership II atau TA harus buat aplikasi ? D : Ga harus buat aplikasi tapi harus ngoding

M : Pa saya bingung mau ngapain, saya juga bingung mau presentasi apa? D : Makanya baca de, buka jurnal topik ‘ganteng’ nah kamu baca dulu sehari 5 kali ya, 4 hari udah 20 tuh. Bingung itu tanda kurang wawasan alias kurang baca.

M : Pa saya sudah cari jurnal terindeks scopus tapi ga nemu. D : Kamu punya mata de? coba dicolok dulu. Kamu udah lakuin apa aja? tolong di list laporkan ke grup Tingkat Akhir. Tinggal buka google scholar klik dari tahun 2014, cek nama jurnalnya di scimagojr.com beres.

M : Pa saya belum dapat tempat intership, jadi ga tau mau presentasi apa? D : kamu kok ga nyambung, yang dipresentasikan itu yang kamu baca bukan yang akan kamu lakukan.

M : Pa ini jurnal harus yang terindex scopus ga bisa yang lain ? D : Index scopus menandakan artikel tersebut dalam standar semantik yang mudah dipahami dan dibaca serta bukan artikel asal jadi. Jika diluar scopus biasanya lebih sukar untuk dibaca dan dipahami karena tidak adanya proses review yang baik dan benar terhadap artikel.

M : Pa saya tidak mengerti D : Coba lihat standar alasan

M : Pa saya bingung D : Coba lihat standar alasan

M : Pa saya sibuk D : Mbahmu....

M : Pa saya ganteng D : Ndasmu....

M : Pa saya kece D : wes karepmu lah....

Biasanya anda memiliki alasan tertentu jika menghadapi kendala saat proses bimbingan, disini saya akan melakukan standar alasan agar persepsi yang diterima sama dan tidak salah kaprah. Penggunaan kata alasan tersebut antara lain :

1. Tidak Mengerti : anda boleh menggunakan alasan ini jika anda sudah melakukan tahapan membaca dan meresumekan 15 jurnal. Sudah mencoba dan mempraktekkan teorinya dengan mencari di youtube dan google minimal 6 jam sehari selama 3 hari berturut-turut.
2. Bingung : anda boleh mengatakan alasan bingung setelah maksimal dalam berusaha menyelesaikan tugas bimbingan dari dosen(sudah dilakukan semua). Anda belum bisa mengatakan alasan bingung jika anda masih belum menyelesaikan tugas bimbingan dan poin nomor 1 diatas. Setelah anda menyelesaikan tugas bimbingan secara maksimal dan tahap 1 poin diatas, tapi anda masih tetap bingung maka anda boleh memakai alasan ini.

Bibliography

- [1] Abdillah Baraja. Kecerdasan buatan tinjauan historikal. *Speed-Sentra Penelitian Engineering dan Edukasi*, 1(1), 2008.
- [2] Joshua Eckroth. *Python Artificial Intelligence Projects for Beginners: Get up and running with Artificial Intelligence using 8 smart and exciting AI applications*. Packt Publishing Ltd, 2018.
- [3] Stuart J Russell and Peter Norvig. *Artificial intelligence: a modern approach*. Malaysia; Pearson Education Limited,, 2016.
- [4] Xiaojin Zhu and Andrew B Goldberg. Introduction to semi-supervised learning. *Synthesis lectures on artificial intelligence and machine learning*, 3(1):1–130, 2009.