

CERDAS MENGUASAI GIT

CERDAS MENGUASAI GIT

Dalam 24 Jam

Rolly M. Awangga
Informatics Research Center



Kreatif Industri Nusantara

Penulis:

Rolly Maulana Awangga

ISBN : 978-602-53897-0-2

Editor:

M. Yusril Helmi Setyawan

Penyunting:

Syafrial Fachrie Pane

Khaera Tunnisa

Diana Asri Wijayanti

Desain sampul dan Tata letak:

Deza Martha Akbar

Penerbit:

Kreatif Industri Nusantara

Redaksi:

Jl. Ligar Nyawang No. 2

Bandung 40191

Tel. 022 2045-8529

Email : awangga@kreatif.co.id

Distributor:

Informatics Research Center

Jl. Sariasih No. 54

Bandung 40151

Email : irc@poltekpos.ac.id

Cetakan Pertama, 2019

Hak cipta dilindungi undang-undang

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

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

CONTRIBUTORS

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

CONTENTS IN BRIEF

1	Pengenalan Kecerdasan Buatan dan Scikit-Learn	1
2	Membangun Model Prediksi	17
3	Prediksi dengan Random Forest	33
4	Experiment and Result	53
5	Vektorisasi Kata dan Dokumen	65
6	MFCC dan Neural Network	77
7	CNN	91

DAFTAR ISI

Daftar Gambar	xv
Daftar Tabel	xxiii
Foreword	xxvii
Kata Pengantar	xxix
Acknowledgments	xxxi
Acronyms	xxxiii
Glossary	xxxv
List of Symbols	xxxvii
Introduction	xxxix
<i>Rolly Maulana Awangga, S.T., M.T.</i>	
1 Pengenalan Kecerdasan Buatan dan Scikit-Learn	1
1.1 Kecerdasan Buatan	1
1.1.1 Definisi Kecerdasan Buatan	1
1.1.2 Sejarah Kecerdasan Buatan	2
1.1.3 Perkembangan Kecerdasan Buatan	2
	ix

1.2	Scikit-Learn	3
1.2.1	Supervised Learning	3
1.2.2	Regresi	4
1.2.3	Klasifikasi	4
1.2.4	Unsupervised Learning	5
1.2.5	Data Set	5
1.2.6	Training Set	6
1.2.7	Testing Set	6
1.3	Instalasi dan Pemakaian Scikit-Learn	7
1.3.1	Instalasi library scikit dari anaconda, mencoba kompilasi dan uji coba ambil contoh kode dan lihat variabel explorer	7
1.3.2	Loading an example dataset	8
1.3.3	Learning and predicting	10
1.3.4	Model Persistence	11
1.3.5	Conventions	13
2	Membangun Model Prediksi	17
2.1	Teori	17
2.1.1	Binary Classification	17
2.1.2	Supervised Learning, Unsupervised Learning dan Clustering	17
2.1.3	Evaluasi dan Akurasi	20
2.1.4	Cara membuat dan membaca confusion matrix	20
2.1.5	K-fold cross validation	21
2.1.6	Decision Tree	22
2.1.7	Information Gain dan Entropi	22
2.2	Pratikum	23
2.2.1	Scikit-Learn	23
2.2.2	Penanganan Error	30
3	Prediksi dengan Random Forest	33
3.1	TEORI	33
3.1.1	Random Forest	33
3.1.2	Dataset	34
3.1.3	Cara Membaca Dataset Dan Arti Setiap File Dan Isi Field Masing Masing File	34
3.1.4	Cross Validation	36

3.1.5	Arti Score 44% Pada Random Forest, 27% Pada Decission Tree Dan 29% Dari SVM	36
3.1.6	Confusion Matriks	36
3.1.7	Voting Pada Random Forest	38
3.2	Praktek Program	40
3.2.1	Aplikasi Sederhana Menggunakan Pandas	40
3.2.2	Aplikasi Sederhana Menggunakan Numpy	41
3.2.3	Aplikasi Sederhana Menggunakan Matplotlib	42
3.2.4	Menjalankan Program Klasifikasi Random Forest	43
3.2.5	Menjalankan Program Confusion Matrix	47
3.2.6	Menjalankan Program Klasifikasi SVM dan Decission Tree	49
3.2.7	Menjalankan Program Cross Validation	49
3.2.8	Menjalankan Program Komponen Informasi	50
3.3	Penanganan Error	51
3.3.1	Error Index	51
4	Experiment and Result	53
4.1	Teori	53
4.1.1	Klasifikasi Teks	53
4.1.2	Klasifikasi Bunga	54
4.1.3	Pembelajaran Mesin Pada Teks Kata - Kata di Youtube	54
4.1.4	Arti Score 44% Pada Random Forest, 27% Pada Decission Tree Dan 29% Dari SVM	55
4.1.5	Bag of Words	55
4.1.6	TF-IDF	56
4.2	PRAKTIKUM	56
4.2.1	Aplikasi Sederhana Menggunakan Pandas	56
4.2.2	Memecah DataFrame Menjadi 2 Dataframe	57
4.2.3	Vektorisasi Dan Klasifikasi Dari Data Youtube Eminem Dengan Decision Tree	58
4.2.4	Vektorisasi Dan Klasifikasi Dari Data Youtube Eminem Dengan SVM	59
4.2.5	Vektorisasi Dan Klasifikasi Dari Data Youtube Eminem Dengan Decision Tree 2	59
4.2.6	Plotting Confusion Matrix	60
4.2.7	Menjalankan Program Cross Validation	61
4.2.8	Program Pengamatan Komponen Informasi	62

4.3	Penanganan Error	62
4.3.1	Error Index	62
5	Vektorisasi Kata dan Dokumen	65
5.1	Teori	65
5.1.1	Vektorisasi	65
5.1.2	Vektor Dataset Google	66
5.1.3	Konsep Vektorisasi Untuk Kata	66
5.1.4	Konsep Vektorisasi Untuk Dokumen	66
5.1.5	Mean Dan Standar Deviasi	66
5.1.6	Skip-gram	67
5.2	PRAKTEK PROGRAM	67
5.2.1	Mencoba Dataset	67
5.2.2	Extract Words dan PermuteSentences	70
5.2.3	Fungsi Librari gensim TaggedDocument dan Doc2Vec	71
5.2.4	Menambahkan data Training Dari File Dengan Doc2Vec	71
5.2.5	Mengapa Harus Dilakukan Pengocokan Dan Pembersihan Data	73
5.2.6	Mengapa Model Harus Di Save Dan Temporari Training Harus Dihapus	74
5.2.7	Infer Code	75
5.2.8	Cosine Similarity	75
5.2.9	Score Dari Cross Validation	76
5.2.10	Penanganan Error	76
6	MFCC dan Neural Network	77
6.1	Teori	77
6.2	Praktek Program	80
6.2.1	GTZAN Genre Collection dan data dari freesound	80
6.2.2	Fungsi Display MFCC	82
6.2.3	Fungsi Extract Features Song	82
6.2.4	Fungsi Generate Features And Labels	83
6.2.5	Penggunaan Fungsi Generate Features And Labels Sangat Lama Ketika Meload Dataset Genre	84
6.2.6	Pemisahan Data Training Dan Data Set Sebesar 80%	85
6.2.7	Fungsi Sequential	86
6.2.8	Fungsi Compile	87
6.2.9	Fungsi Fit	88

6.2.10	Fungsi Evaluate	88
6.2.11	Fungsi Predict	89
6.3	Penanganan Error	90
6.3.1	Module Error	90
7	CNN	91
7.1	Teori	91
7.1.1	Teks Tokenizer	91
7.1.2	konsep dasar K Fold Cross Validation pada dataset komentar Youtube	91
7.1.3	kode program for train, test in splits	92
7.1.4	Jelaskan apa maksudnya kode program <i>train_content = d['CONTENT'].iloc[train_idx]</i> dan <i>test_content = d['CONTENT'].iloc[test_idx]</i> . dilengkapi dengan ilustrasi atau gambar	92
7.1.5	Soal No. 5 Jelaskan apa maksud dari fungsi <i>tokenizer = Tokenizer(num_words=2000)</i> dan <i>tokenizer.fit_on_texts(train_content)</i> , dilengkapi dengan ilustrasi atau gambar	92
7.1.6	Jelaskan apa maksud dari fungsi <i>d_train_inputs = tokenizer.texts_to_matrix(train_content, mode='tfidf')</i> dan <i>d_test_inputs = tokenizer.texts_to_matrix(test_content, mode='tfidf')</i> , dilengkapi dengan ilustrasi kode dan atau gambar	93
7.1.7	Praktek	96
7.1.8	Penanganan Error	111
	Daftar Pustaka	113
	Index	115

DAFTAR GAMBAR

1.1	Versi Anaconda Yang Digunakan	7
1.2	Versi Python Yang Digunakan	7
1.3	Instalasi Scikit Dari Anaconda	7
1.4	Contoh Skrip	8
1.5	Hasil Yang Muncul Di CMD	8
1.6	Gambar Yang Muncul Dari Matplotlib	9
1.7	Penjelasan	9
1.8	Penjelasan 2	9
1.9	Penjelasan 3	9
1.10	Penjelasan 4	10
1.11	Membuka Python Shell	10
1.12	Menggunakan Estimator Sklearn	10
1.13	Mendefinisikan Classifier	10

1.14	Memanggil Classifier Tanpa Baris Terakhir	11
1.15	Memprediksi Nilai Baru	11
1.16	Hasil Pengujian Classifier	11
1.17	Hasil Pengujian Classifier	12
1.18	Pickle Pada Python	12
1.19	Pengujian Classifier Pickle	12
1.20	Penggunaan Joblib	12
1.21	Deklarasi Numpy	13
1.22	Contoh Type Casting	13
1.23	Menggunakan FitTransform	13
1.24	Regresi Yang Dilempar	14
1.25	Refitting dan Memperbaharui Parameter	14
1.26	MultiClass Classifier	15
1.27	MultiClass Classifier biner 2D	15
1.28	MultiLabel Classifier	15
2.1	Binary Classification	18
2.2	Supervised Learning	19
2.3	Unsupervised Learning	19
2.4	Cluster	20
2.5	Evaluasi dan Akurasi	20
2.6	K-fold cross validation	21
2.7	Decision Tree	22
2.8	Information gain	23
2.9	Entropi	23
2.10	Loading Dataset	24
2.11	Generate Binary Label	24
2.12	One-hot Encoding	25
2.13	Shuffle Rows	26

2.14	Fit Decision Tree	26
2.15	Fit Decision Tree	27
2.16	Fit Decision Tree	27
2.17	Score	28
2.18	Cross Val Score	28
2.19	Max Depth	29
2.20	Depth in Range	29
2.21	Matplotlib	30
2.22	Error Graphviz	30
2.23	Folder Graphviz	31
2.24	Menambahkan Graphviz kePATH	31
2.25	Evaluasi Error	31
2.26	Error File Not Exist	32
2.27	Kolom Direktori	32
2.28	Memasuki Direktori Dataset	32
2.29	Evaluasi Error	32
3.1	Random Forest Spyder	34
3.2	Random Forest Graphic	34
3.3	Dataset Pandas	34
3.4	Dataset Pandas	35
3.5	Confusion Matrix	38
3.6	Voting Random Forest	40
3.7	Aplikasi Sederhana Menggunakan Pandas	41
3.8	Aplikasi Sederhana Menggunakan Numpy	42
3.9	Aplikasi Sederhana Menggunakan Matplotlib	43
3.10	Program Random Forest Tasya	44
3.11	Program Random Forest Tasya	44
3.12	Program Random Forest Tasya	44

3.13	Program Random Forest Tasya	44
3.14	Program Random Forest Tasya	45
3.15	Program Random Forest Tasya	45
3.16	Program Random Forest Tasya	45
3.17	Program Random Forest Tasya	45
3.18	Program Random Forest Tasya	45
3.19	Program Random Forest Tasya	45
3.20	Program Random Forest Tasya	46
3.21	Program Random Forest Tasya	46
3.22	Program Random Forest Tasya	46
3.23	Program Random Forest Tasya	46
3.24	Program Random Forest Tasya	47
3.25	Program Random Forest Tasya	47
3.26	Program Random Forest Tasya	47
3.27	Program Random Forest Tasya	47
3.28	Program Confusion Matrix Tasya	47
3.29	Program Confusion Matrix Tasya	48
3.30	Program Confusion Matrix Tasya	48
3.31	Program Confusion Matrix Tasya	48
3.32	Program Confusion Matrix Tasya	49
3.33	Program Decission Tree Tasya	49
3.34	Program SVM Tasya	49
3.35	Program Cross Validation Tasya	50
3.36	Program Cross Validation Tasya	50
3.37	Program Cross Validation Tasya	50
3.38	Program Komponen Informasi Tasya	50
3.39	Program Komponen Informasi Tasya	51
3.40	Error Index	51

3.41	File Codingan	51
3.42	Menghapus Spasi	52
3.43	Error Teratasi	52
4.1	Klasifikasi Teks Tasya	54
4.2	Klasifikasi Bunga Berwana Ungu Tasya	54
4.3	Klasifikasi Comment Spam Di Youtube Tasya	55
4.4	Bag of Words Tasya	55
4.5	Contoh TF-IDF Tasya	56
4.6	Dataset Original Tasya	57
4.7	Dataset Dummy Tasya	58
4.8	Split DataFrame Tasya	58
4.9	Dataset Youtube Eminem Tasya	58
4.10	Dataset Youtube Eminem Tasya	59
4.11	Dataset Youtube Eminem SVM Tasya	59
4.12	Dataset Youtube Eminem Tasya	59
4.13	Confusion Matrix Tasya	61
4.14	Cross Validation Tasya	61
4.15	Hasil Cross Validation Tasya	61
4.16	Program Komponen Informas Tasyai	62
4.17	Error Key Tasya	62
4.18	Error Key Tasya	63
4.19	Error Key Tasya	63
4.20	Error Key Tasya	63
5.1	Contoh Mean dan Standar Deviasi	67
5.2	Contoh Mean dan Standar Deviasi	67
5.3	Contoh Skipgram	68
5.4	Vektor Love Tasya	68
5.5	Vektor Faith Tasya	68

5.6	Vektor Fall Tasya	68
5.7	Vektor Sick Tasya	69
5.8	Vektor Clear Tasya	69
5.9	Vektor Shine Tasya	69
5.10	Vektor Bag Tasya	69
5.11	Vektor Car Tasya	69
5.12	Vektor Wash Tasya	69
5.13	Vektor Motor Tasya	70
5.14	Similariti Tasya	70
5.15	Extract Words Tasya	71
5.16	PermuteSentencesi Tasya	71
5.17	TaggedDocument Tasya	71
5.18	Data Training Imdb Tasya	72
5.19	Data Training Polarity Tasya	72
5.20	Data Training Tomatoes	73
5.21	Pengcokan Dan Pembersihan Data Tasya	74
5.22	Save Model Tasya	74
5.23	Save Model Tasya	74
5.24	Save Model HAsil Tasya	74
5.25	Save Model HasilTasya	74
5.26	Infer Code Tasya	75
5.27	Infer Code Tasya	76
5.28	Score Cross Validation Tasya	76
5.29	Error Memory Tasya	76
6.1	Contoh Pembobotan Neural Network Tasya	78
6.2	Cara Membaca Hasil Plot MFCC Tasya	79
6.3	One Hot Encoding Tasya	79
6.4	Numpy Unique Tasya	80

6.5	To Categorical Tasya	80
6.6	Sequential Tasya	80
6.7	Meload Data Genre Collection Tasya	82
6.8	Display MFCC Tasya	82
6.9	Hasil Display MFCC Tasya	83
6.10	Extract Features Tasya	83
6.11	Fungsi Generate Features And Labels Tasya	84
6.12	Hasil Fungsi Generate Features And Labels Tasya	85
6.13	Pemisahan Data Training dan Data Set Tasya	86
6.14	Pemisahan Data Training dan Data Set Tasya	86
6.15	Pemisahan Data Training dan Data Set Tasya	87
6.16	Fungsi Compile Tasya	87
6.17	Fungsi Fit Tasya	88
6.18	Fungsi Evaluate Tasya	89
6.19	Fungsi Evaluate Tasya	89
6.20	Fungsi Predict Tasya	89
6.21	Module Error Tasya	90
6.22	Penyelesaian Module Error Tasya	90
7.1	Ilustrasi KFold Cross Tasya	92
7.2	Ilustrasi Text To Matrix Tasya	93
7.3	Ilustrasi np Absolute Tasya	93
7.4	Ilustrasi One Hot Encoding Tasya	94
7.5	Ilustrasi Neural Network Pemodelan Tasya	94
7.6	Algoritma Konvolusi Tasya	96
7.7	Algoritma Konvolusi Tasya	96
7.8	Algoritma Konvolusi Tasya	96
7.9	Algoritma Konvolusi Tasya	97
7.10	Algoritma Konvolusi Tasya	97

7.11	Algoritma Konvulasi Tasya	98
7.12	Algoritma Konvulasi Tasya	98
7.13	Kode Program Blok In 1 Tasya	98
7.14	Kode Program Blok In 2 Tasya	99
7.15	Kode Program Blok In 3 Tasya	100
7.16	Kode Program Blok In 4 Tasya	100
7.17	Kode Program Blok In 5 Tasya	101
7.18	Kode Program Blok In 6 Tasya	101
7.19	Kode Program Blok In 7 Tasya	101
7.20	Kode Program Blok In 8 Tasya	102
7.21	Kode Program Blok In 9 Tasya	103
7.22	Kode Program Blok In 10 Tasya	104
7.23	Kode Program Blok In 11 Tasya	104
7.24	Kode Program Blok In 12 Tasya	105
7.25	Kode Program Blok In 13 Tasya	107
7.26	Kode Program Blok In 14 Tasya	108
7.27	Kode Program Blok In 15 Tasya	108
7.28	Kode Program Blok In 16 Tasya	109
7.29	Kode Program Blok In 17 Tasya	109
7.30	Kode Program Blok In 18 Tasya	109
7.31	Kode Program Blok In 19 Tasya	110
7.32	Kode Program Blok In 20 Tasya	111
7.33	Error Tasya	111
7.34	Penanganan Error Kernel Tasya	112

DAFTAR TABEL

Listings

3.1	Code Program Sederhana Pandas	40
3.2	Code Program Sederhana Numpy	41
3.3	Code Program Sederhana Matplotlib	42
6.1	Kode Load Data Untuk MFCC	81
6.2	Code Fungsi Display MFCC	82
6.3	Panggil Generate Labels	84
6.4	Code Pemisahan Data Training Dan Testing	85
6.5	Code Fungsi Sequential	86
6.6	Code Fungsi Compile	87
6.7	Code Fungsi Fit	88
6.8	Code Fungsi Evaluate	88
6.9	Code Fungsi Predict	89
7.1	K Fold Cross Validation	91
7.2	Membuat model Neural Network	94
7.3	Compile model	95
	src/Chapter7/1164086/in1.py	96
	src/Chapter7/1164086/in2.py	97
	src/Chapter7/1164086/in3.py	99

src/Chapter7/1164086/in4.py	100
src/Chapter7/1164086/in5.py	100
src/Chapter7/1164086/in6.py	101
src/Chapter7/1164086/in7.py	101
src/Chapter7/1164086/in8.py	102
src/Chapter7/1164086/in9.py	102
src/Chapter7/1164086/in10.py	103
src/Chapter7/1164086/in11.py	104
src/Chapter7/1164086/in12.py	104
src/Chapter7/1164086/in13.py	105
src/Chapter7/1164086/in14.py	107
src/Chapter7/1164086/in15.py	108
src/Chapter7/1164086/in16.py	108
src/Chapter7/1164086/in17.py	109
src/Chapter7/1164086/in18.py	109
src/Chapter7/1164086/in19.py	110
src/Chapter7/1164086/in19.py	110

FOREWORD

Sepatah kata dari Kaprodi, Kabag Kemahasiswaan dan Mahasiswa

KATA PENGANTAR

Buku ini diciptakan bagi yang awam dengan git sekalipun.

R. M. AWANGGA

*Bandung, Jawa Barat
Februari, 2019*

ACKNOWLEDGMENTS

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

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

R. M. A.

ACRONYMS

AI	Artificial Intelligence
ETL	Extract Transform Load
NLP	Natural Language Processing

GLOSSARY

cybernetics	Adalah sistem yang berinteraksi langsung dengan diri sendiri yang memahami dan menentukan proses tujuan.
Heuristik	Adalah sebuah metode yang mengembangkan efisiensi dalam proses pencarian.
Supervised	Adalah sebuah tugas pengumpulan data untuk menyimpulkan fungsi dari data pelatihan berlabel.
Unsupervised	Adalah Tidak adanya memiliki data latih, sehingga dari data yang ada kita mengelompokkan data tersebut menjadi 2 ataupun 3 bagian.

SYMBOLS

- A Amplitude
- $\&$ Propositional logic symbol
- a Filter Coefficient

- \mathcal{B} Number of Beats

INTRODUCTION

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

Informatics Research Center
Bandung, Jawa Barat, Indonesia

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

Rumusnya cuman sebagai contoh aja biar keren[1].

$$ABCDEF\alpha\beta\Gamma\Delta\sum_{def}^{abc} \tag{I.1}$$

BAB 1

Pengenalan Kecerdasan Buatan dan Scikit-Learn

1.1 Kecerdasan Buatan

1.1.1 Definisi Kecerdasan Buatan

Kecerdasan Buatan biasa disebut dengan istilah AI (Artificial Intelligence). AI sendiri merupakan suatu cabang dalam bisnis sains komputer sains dimana mengkaji tentang bagaimana cara untuk melengkapi sebuah komputer dengan kemampuan atau kepintaran layaknya atau mirip dengan yang dimiliki manusia. Sebagai contoh, sebagaimana komputer dapat berkomunikasi dengan pengguna baik menggunakan kata, suara maupun lain sebagainya. Dengan kemampuan ini, diharapkan komputer mampu mengambil keputusan sendiri untuk berbagai kasus yang ditemuinya kemudian itulah yang disebut dengan kecerdasan buatan. Kecerdasan buatan adalah kemampuan komputer digital atau robot yang dikendalikan komputer untuk melakukan tugas yang umumnya dikaitkan dengan sesuatu yang cerdas. Istilah ini sering diterapkan pada proyek pengembangan sistem yang diberkahi dengan karakteristik proses intelektual manusia, seperti kemampuan untuk berpikir, menemukan makna, menggeneralisasi, atau belajar dari pengalaman masa lalu.

Kecerdasan Buatan adalah salah satu bidang studi yang berhubungan dengan pemanfaatan mesin untuk memecahkan persoalan yang rumit dengan cara lebih manusiawi dan lebih bisa di pahami oleh manusia. Kecerdasan buatan makin canggih dengan kemampuan komputer dalam memperbarui pengetahuannya dengan banyaknya testing dan perkembangan target analisa. Untuk kecerdasan buatan ada banyak contoh dan jenisnya. Salah satu contoh yang paling terkenal dari Artificial Intelligence ialah Google Assistant. Google Assistant digunakan untuk kemudahan user dalam menemukan berbagai hal maupun penyetingan langsung terhadap smartphone yang digunakan dan masih banyak lagi.

1.1.2 Sejarah Kecerdasan Buatan

Artificial intelligence merupakan inovasi baru di bidang ilmu pengetahuan. Mulai terbentuk sejak adanya komputer modern dan kira-kira terjadi sekitaran tahun 1940 dan 1950. Ilmu pengetahuan komputer ini khusus ditujukan dalam perancangan otomatisasi tingkah laku cerdas dalam sistem kecerdasan komputer. Pada awal 50-an, studi tentang “mesin berpikir” memiliki berbagai nama seperti cybernetics, teori automata, dan pemrosesan informasi. Pada tahun 1956, para ilmuwan jenius seperti Alan Turing, Norbert Wiener, Claude Shannon dan Warren McCulloch telah bekerja secara independen di bidang cybernetics, matematika, algoritma dan teori jaringan. Namun, seprang ilmuwan komputer dan kognitif John McCarthy adalah orang yang dating dengan ide untuk bergabung dengan upaya penelitian terpisah ini kedalam satu bidang yang akan mempelajari topic baru untuk imajinasi manusia yaitu kecerdasan buatan. Dia adalah orang yang menciptakan istilah tersebut dan kemudian mendirikan laboratorium Kecerdasan Buatan di MIT dan Stanford.

Pada tahun 1956, McCarthy yang sama mendirikan Konferensi Dartmouth di Hanover, New Hampshire. Peneliti terkemuka dalam teori kompleksitas, simulasi bahasa, hubungan antara keacakan dan pemikiran kreatif, jaringan saraf diundang. Tujuan dari bidang penelitian yang baru dibuat adalah untuk mengembangkan mesin yang dapat mensimulasikan setiap aspek kecerdasan. Itulah sebabnya Konferensi Dartmouth 1956 dianggap sebagai kelahiran Kecerdasan Buatan. Sejak saat itu, Kecerdasan Buatan telah hidup melalui decade kemuliaan dan cemoohan, yang dikenal luas sebagai musim panas dan musim dingin AI. Musim panasnya ditandai dengan optimism dan dana besar, sedangkan musim dinginnya dihadapkan dengan pemotongan dana, ketidakpercayaan dan pesimisme.

1.1.3 Perkembangan Kecerdasan Buatan

Teknologi Artificial Intelligence semakin ramai dibahas dalam berbagai diskusi teknologi di seluruh dunia. Menurut kebanyakan orang, pekerjaan seperti kasir, operator telepon, pengemudi truk, dan lainnya sangat berpeluang besar untuk tergantikan oleh Artificial Intelligence. Mengapa terjadi hal demikian? dikarenakan memang bahwa AI lebih unggul dalam hal kinerja, fitur dan lain sebagainya. Namun, dalam beberapa aspek memang pekerja manusia masih unggul dibandingkan AI itu sendiri. Para

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

AI Summer 1 (1956-1973) KONferensi Dartmouth diikuti oleh 17 tahun kemajuan luar biasa. Proyek penelitian yang dilakukan di MIT, universitas di Edinburgh, Stanford dan Carnegie Mellon menerima dana besar-besaran, yang akhirnya membuahkan hasil. Selama tahun-tahun itulah komputer pemrograman mulai melakukan masalah aljabar, membuktikan teorema geometris, memahami dan menggunakan sintaks dan tata bahasa Inggris. Terlepas dari ditinggalkannya koneksionisme dan terjemahan mesin yang gagal, yang menunda penelitian Natural Language Processing (NLP) selama bertahun-tahun, banyak prestasi dari masa lalu yang membuat sejarah. Berikut ini beberapa diantaranya : Pelopor pembelajaran mesin, Ray Solomonoff meletakkan dasar-dasar teori matematika AI, memperkenalkan metode Bayesian universal untuk inferensi dan prediksi induktif Thomas Evans menciptakan program ANALOGI heuristik, yang memungkinkan komputer memecahkan masalah geometri-analogi Unimation, perusahaan robotika pertama di dunia, menciptakan robot industri Unimate, yang bekerja pada jalur perakitan mobil General Motors. Joseph Weizenbaum membangun ELIZA-program interaktif yang dapat membawa percakapan dalam bahasa Inggris tentang topik apapun. Ross Quillian menunjukkan jaring semantik, sedangkan Jaime Carbonell (Sr.) mengembangkan Cendickia-program interaktif untuk instruksi yang dibantu komputer berdasarkan jaring semantik. Edward Feigenbaum dan Julian Feldman menerbitkan *Computeks and Thought*, kumpulan artikel pertama tentang AI.

1.2 Scikit-Learn

1.2.1 Supervised Learning

Supervised Learning adalah tugas pengumpulan data untuk menyimpulkan fungsi dari data pelatihan berlabel. Data pelatihan terdiri dari serangkaian contoh pelatihan. Dalam supervised learning, setiap contoh adalah pasangan yang terdiri dari objek input (biasanya vektor) dan nilai output yang diinginkan (juga disebut sinyal pengawasan super). Algoritma pembelajaran yang diawasi menganalisis data pelatihan dan menghasilkan fungsi yang disimpulkan, yang dapat digunakan untuk memetakan contoh-contoh baru. Skenario optimal akan memungkinkan algoritma menentukan label kelas dengan benar untuk instance yang tidak terlihat. Ini membutuhkan algoritma pembelajaran untuk menggeneralisasi dari data pelatihan untuk situasi yang tidak terlihat dengan cara yang "masuk akal". Supervised Learning adalah pen-

dekatan dimana sudah terdapat data yang dilatih selain itu juga terdapat variable yang ditargetkan sehingga tujuan dari pendekatan ini yaitu mengelompokkan suatu data ke data yang sudah ada. Supervised Learning menyediakan algoritma pembelajaran dengan jumlah yang diketahui untuk mendukung penilaian dimasa depan. Chatbots, mobil self-driving, program pengenalan wajah, sistem pakar dan robot adalah beberapa sistem yang dapat menggunakan pembelajaran yang diawasi atau tidak diawasi. Supervised Learning sebagian besar terkait dengan AI berbasis pengambilan tetapi mereka juga mungkin mampu menggunakan model pembelajaran generatif. Data pelatihan untuk pembelajaran yang diawasi mencakup serangkaian contoh dengan subjek input berpasangan dan output yang diinginkan (yang juga disebut sebagai sinyal pengawasan).

Dalam pembelajaran yang diawasi untuk pemrosesan gambar, misalnya sistem AI mungkin dilengkapi dengan gambar berlabel kendaraan dalam kategori seperti mobil dan truk. Setelah jumlah pengamatan yang cukup, sistem harus dapat membedakan antara dan mengkategorikan gambar yang tidak berlabel, dimana waktu pelatihan dapat dikatakan lengkap. Model Supervised Learning memiliki beberapa keunggulan dibandingkan pendekatan tanpa pengawasan, tetapi mereka juga memiliki keterbatasan. Sistem lebih cenderung membuat penilaian bahwa manusia dapat berhubungan, misalnya karena manusia telah memberikan dasar untuk keputusan. Namun, dalam kasus metode berbasis pengambilan, Supervised Learning mengalami kesulitan dalam menangani informasi baru. Jika suatu sistem dengan kategori untuk mobil dan truk disajikan dengan sepeda, misalnya ia harus salah dikelompokkan dalam satu kategori atau yang lain. Namun, jika sistem AI bersifat generatif, ia mungkin tidak tahu apa sepeda itu tetapi akan dapat mengenalinya sebagai milik kategori yang terpisah.

1.2.2 Regresi

Regresi adalah metode analisis statistik yang digunakan untuk melihat pengaruh antara dua ataupun lebih variabel. Regresi adalah membahas masalah ketika variabel output adalah nilai riil atau berkelanjutan, seperti "gaji" atau "berat". Banyak model yang berbeda dapat digunakan, yang paling sederhana adalah regresi linier. Ia mencoba untuk menyesuaikan data dengan hyper-plane terbaik yang melewati poin.

1.2.3 Klasifikasi

Klasifikasi adalah pembagian sesuatu menurut kelas-kelas (class). Menurut Ilmu Pengetahuan, Klasifikasi merupakan proses pengelompokkan benda berdasarkan ciri-ciri persamaan dan juga perbedaan. Dalam masalah klasifikasi, kami mencoba memprediksi sejumlah nilai terpisah. Label (y) umumnya datang dalam bentuk kategorikal dan mewakili sejumlah kelas. Dalam pembelajaran mesin dan statistik, klasifikasi adalah pendekatan pembelajaran yang diawasi di mana program komputer belajar dari input data yang diberikan kepadanya dan kemudian menggunakan pembelajaran ini untuk mengklasifikasikan pengamatan baru. Kumpulan data ini mungkin hanya bersifat dua kelas (seperti mengidentifikasi apakah orang tersebut berjenis ke-

lamin laki-laki atau perempuan atau bahwa surat itu spam atau bukan-spam) atau mungkin juga multi-kelas. Beberapa contoh masalah klasifikasi adalah: pengenalan ucapan, pengenalan tulisan tangan, identifikasi metrik, klasifikasi dokumen dll.

1.2.4 Unsupervised Learning

Unsupervised Learning berbeda dengan Supervised Learning. Perbedaannya ialah unsupervised learning tidak memiliki data latih, sehingga dari data yang ada kita mengelompokkan data tersebut menjadi 2 ataupun 3 bagian dan seterusnya. Unsupervised Learning adalah pelatihan algoritma kecerdasan buatan (AI) menggunakan informasi yang tidak diklasifikasikan atau diberi label dan memungkinkan algoritma untuk bertindak atas informasi tersebut tanpa bimbingan. Dalam Unsupervised Learning, sistem AI dapat mengelompokkan informasi yang tidak disortir berdasarkan persamaan dan perbedaan meskipun tidak ada kategori yang disediakan.

Dalam Unsupervised Learning, sistem AI disajikan dengan data yang tidak berlabel, tidak ter kategorisasi dan algoritma sistem bekerja pada data tanpa pelatihan sebelumnya. Outputnya tergantung pada algoritma kode. Menundukkan suatu sistem pada Unsupervised Learning adalah salah satu cara untuk menguji AI. Algoritma Unsupervised Learning dapat melakukan tugas pemrosesan yang lebih kompleks daripada sistem pembelajaran yang diawasi. Namun, pembelajaran tanpa pengawasan bisa lebih tidak terduga daripada model alternatif. Sementara Unsupervised Learning mungkin, misalnya, mencari tahu sendiri cara memilah kucing dari anjing, mungkin juga menambahkan kategori yang tidak terduga dan tidak diinginkan untuk menangani breed yang tidak biasa, membuat kekacauan bukannya keteraturan

1.2.5 Data Set

Dataset adalah objek yang merepresentasikan data dan juga relasi yang ada di memory. Strukturnya mirip dengan data di database, namun bedanya dataset berisi koleksi dari data table dan data relation. Mendapatkan data yang tepat berarti mengumpulkan atau mengidentifikasi data yang berkorelasi dengan hasil yang ingin Anda prediksi; yaitu data yang berisi sinyal tentang peristiwa yang Anda pedulikan. Data harus diselaraskan dengan masalah yang Anda coba selesaikan. Gambar kucing tidak terlalu berguna ketika Anda sedang membangun sistem identifikasi wajah. Memverifikasi bahwa data selaras dengan masalah yang ingin Anda selesaikan harus dilakukan oleh ilmuwan data. Jika Anda tidak memiliki data yang tepat, maka upaya Anda untuk membangun solusi AI harus kembali ke tahap pengumpulan data. Format ujung kanan untuk pembelajaran dalam umumnya adalah tensor, atau array multi-dimensi. Jadi jalur pipa data yang dibangun untuk pembelajaran mendalam umumnya akan mengkonversi semua data - baik itu gambar, video, suara, teks atau deret waktu menjadi vektor dan tensor yang dapat diterapkan operasi aljabar linier. Data itu seringkali perlu dinormalisasi, distandarisasi dan dibersihkan untuk meningkatkan kegunaannya, dan itu semua adalah langkah dalam ETL pembelajaran mesin. DeepLearning4j menawarkan alat ETV DataVec untuk melakukan tugas-tugas pemrosesan data tersebut.

Pembelajaran yang dalam, dan pembelajaran mesin yang lebih umum, membutuhkan pelatihan yang baik agar bekerja dengan baik. Mengumpulkan dan membangun set pelatihan badan yang cukup besar dari data yang diketahui membutuhkan waktu dan pengetahuan khusus domain tentang di mana dan bagaimana mengumpulkan informasi yang relevan. Perangkat pelatihan bertindak sebagai tolok ukur terhadap mana jaring pembelajaran dalam dilatih. Itulah yang mereka pelajari untuk direkonstruksi sebelum mereka melepaskan data yang belum pernah mereka lihat sebelumnya. Pada tahap ini, manusia yang berpengetahuan luas perlu menemukan data mentah yang tepat dan mengubahnya menjadi representasi numerik yang dapat dipahami oleh algoritma pembelajaran mendalam, tensor. Membangun set pelatihan, dalam arti tertentu, pra-pra pelatihan. Set pelatihan yang membutuhkan banyak waktu atau keahlian dapat berfungsi sebagai keunggulan dalam dunia ilmu data dan pemecahan masalah. Sifat keahlian sebagian besar dalam memberi tahu algoritma Anda apa yang penting bagi Anda dengan memilih apa yang masuk ke dalam set pelatihan. Ini melibatkan menceritakan sebuah kisah melalui data awal yang Anda pilih yang akan memandu jaring pembelajaran mendalam Anda saat mereka mengekstraksi fitur-fitur penting, baik di set pelatihan maupun dalam data mentah yang telah mereka ciptakan untuk dipelajari. Untuk membuat set pelatihan yang bermanfaat, Anda harus memahami masalah yang Anda selesaikan; yaitu apa yang Anda inginkan agar jaring pembelajaran mendalam Anda memperhatikan, di mana hasil yang ingin Anda prediksi.

1.2.6 Training Set

Training Set adalah set digunakan oleh algoritma klasifikasi. Dapat dicontohkan dengan : decision tree, bayesian, neural network dll. Semuanya dapat digunakan untuk membentuk sebuah model classifier. Menjalankan pelatihan yang diatur melalui jaringan saraf mengajarkan pada net cara menimbang berbagai fitur, menyesuaikan koefisien berdasarkan kemungkinan mereka meminimalkan kesalahan dalam hasil Anda. Koefisien-koefisien tersebut, juga dikenal sebagai parameter, akan terkandung dalam tensor dan bersama-sama mereka disebut model, karena mereka mengkodekan model data yang mereka latih. Mereka adalah takeaways paling penting yang akan Anda dapatkan dari pelatihan jaringan saraf.

1.2.7 Testing Set

Testing Set adalah set yang digunakan untuk mengukur sejauh mana classifier berhasil melakukan klasifikasi dengan benar. Ini berfungsi sebagai meterai persetujuan, dan Anda tidak menggunakannya sampai akhir. Setelah Anda melatih dan mengoptimalkan data Anda, Anda menguji jaringan saraf Anda terhadap pengambilan sampel acak akhir ini. Hasil yang dihasilkannya harus memvalidasi bahwa jaring Anda secara akurat mengenali gambar, atau mengenalnya setidaknya x dari jumlah tersebut. Jika Anda tidak mendapatkan prediksi yang akurat, kembalilah ke set pelatihan, lihat hyperparameter yang Anda gunakan untuk menyetel jaringan, serta kualitas data Anda dan lihat teknik pra-pemrosesan Anda.

```
C:\Windows\system32\cmd.exe
Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation. All rights reserved.

C:\Users\Pier.Pier-PC>conda --version
conda 4.5.12

C:\Users\Pier.Pier-PC>
```

Gambar 1.1 Versi Anaconda Yang Digunakan

```
C:\Users\Pier.Pier-PC>python --version
Python 2.7.15

C:\Users\Pier.Pier-PC>
```

Gambar 1.2 Versi Python Yang Digunakan

1.3 Instalasi dan Pemakaian Scikit-Learn

1.3.1 Instalasi library scikit dari anaconda, mencoba kompilasi dan uji coba ambil contoh kode dan lihat variabel explorer

1. Pastikan bahwa sudah terinstal Anaconda pada PC anda, caranya buka CMD lalu ketikan `conda --version` jika hasilnya seperti
2. Pastikan juga Kebutuhan Scikit seperti Numpy, Scipy dan Python telah terinstal. untuk mengeceknya buka CMD dan ketikan seperti gambar berikut.
3. Pada CMD ketikan `conda install scikit-learn` kemudian tunggu sampai instalasi selesai.

```
C:\Users\Pier.Pier-PC>conda install scikit-learn
Solving environment: done

## Package Plan ##

  environment location: E:\Anaconda2

added / updated specs:
- scikit-learn

The following packages will be downloaded:

package                                     | build                | size
-----|-----|-----
scikit-learn-0.20.2                       | py27hf381715_0      | 5.1 MB
ca-certificates-2019.1.23                 | py27_0               | 158 KB
conda-4.6.7                               | py27_0               | 1.7 MB
-----|-----|-----
Total:                                     |                      | 7.0 MB

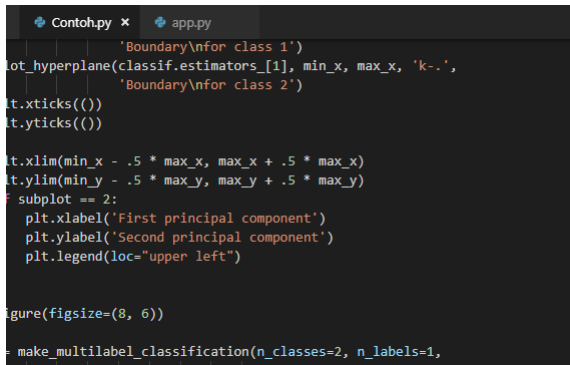
The following packages will be UPDATED:

ca-certificates: 2018.03.07-0 --> 2019.1.23-0
conda:          4.5.12-py27_0  --> 4.6.7-py27_0
scikit-learn:   0.20.1-py27hf381715_0 --> 0.20.2-py27hf381715_0

Proceed [Y/n]? y

Downloading and Extracting Packages
scikit-learn-0.20.2 : 5.1 MB | ##### | 100%
ca-certificates-2019 : 158 KB | ##### | 100%
conda-4.6.7 : 1.7 MB | ##### | 100%
Preparing transaction: done
Verifying transaction: done
Executing transaction: done
```

Gambar 1.3 Instalasi Scikit Dari Anaconda



```

Contoh.py x app.py
'Boundary\nfor class 1')
plot_hyperplane(classif.estimators_[1], min_x, max_x, 'k--',
'Boundary\nfor class 2')
plt.xticks(())
plt.yticks(())

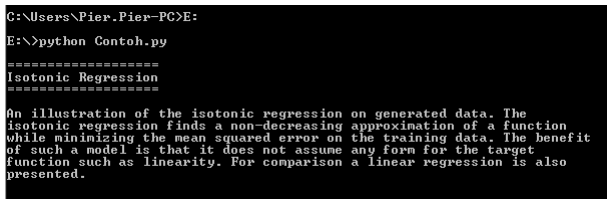
plt.xlim(min_x - .5 * max_x, max_x + .5 * max_x)
plt.ylim(min_y - .5 * max_y, max_y + .5 * max_y)
subplot == 2:
    plt.xlabel('First principal component')
    plt.ylabel('Second principal component')
    plt.legend(loc="upper left")

figure(figsize=(8, 6))

make_multilabel_classification(n_classes=2, n_labels=1,

```

Gambar 1.4 Contoh Skrip



```

C:\Users\Pier.Pier-PC>E:
E:\>python Contoh.py
Isotonic Regression
An illustration of the isotonic regression on generated data. The
isotonic regression finds a non-decreasing approximation of a function
while minimizing the mean squared error on the training data. The benefit
of such a model is that it does not assume any form for the target
function such as linearity. For comparison a linear regression is also
presented.

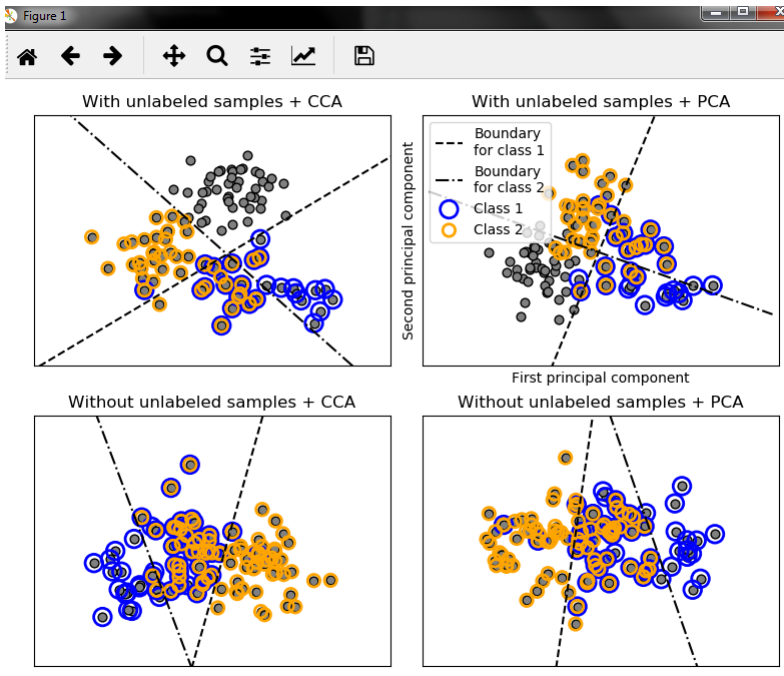
```

Gambar 1.5 Hasil Yang Muncul Di CMD

4. Setelah itu, kita akan mencoba salah satu contoh dasar penggunaan scikit pada website sebelumnya. Dan disini menggunakan contoh Multilabel classification.
5. Salin skrip contoh tersebut ke Text Editor Visual Code atau yang anda miliki. File ini kemudian di save dengan nama 'contoh.py'
6. Setelah tersimpan, jalankan di CMD dengan mengetikan 'python contoh.py' maka akan muncul hasil seperti dibawah ini.

1.3.2 Loading an example dataset

1. Mengimpor dataset, iris dan digit sebagai contoh data.
2. Misalnya, dalam kasus dataset digit, digits.data memberikan akses ke fitur yang dapat digunakan untuk mengklasifikasikan sampel digit.
3. Digit.target memberikan kebenaran dasar untuk dataset digit, yaitu angka yang sesuai dengan setiap gambar digit yang dipelajari.
4. Menggambarkan bagaimana mulai dari masalah awal seseorang dapat membentuk data untuk konsumsi di scikit-belajar.



Gambar 1.6 Gambar Yang Muncul Dari Matplotlib

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

Gambar 1.7 Penjelasan

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

Gambar 1.8 Penjelasan 2

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

Gambar 1.9 Penjelasan 3

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

Gambar 1.10 Penjelasan 4

```
C:\Users\Pier.Pier-PC>python
Python 2.7.15 |Anaconda, Inc.| (default, Dec 10 2018, 21:57:18) [MSC v.1500 64 b
it (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
```

Gambar 1.11 Membuka Python Shell

```
C:\Users\Pier.Pier-PC>python
Python 2.7.15 |Anaconda, Inc.| (default, Dec 10 2018, 21:57:18) [MSC v.1500 64 b
it (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>> from sklearn import svm
>>>
```

Gambar 1.12 Menggunakan Estimator Sklearn

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

Gambar 1.13 Mendefinisikan Classifier

1.3.3 Learning and predicting

1. Pada website sebelumnya, cari Learning And Predicting dan ikuti langkah-langkahnya.
2. Buka Python Shell, atau dengan membukan Command Prompt di PC dan mengetikan python. maka akan masuk ke Python Shellnya pada gambar1.11.
3. Pada python shell ketikan (from sklearn import svm) yang dimana artinya akan memanggil dan menggunakan estimator dari kelas sklearn.svm.SVC pada gambar1.12.
4. Kemudian, kita definisikan clf sebagai classifier, disini gamma didefinisikan secara manual pada gambar1.13.
5. Estimator clf (for classifier) pertama kali dipasang pada model. Ini dilakukan dengan melewati training set ke metode fit. Untuk training set, akan menggunakan semua gambar dari set data yang ada, kecuali untuk gambar terakhir, yang dicadangan untuk prediksi. Pada skrip dibawah memilih training set den-

```

10.2 0.2 0.2 13.2 10.2 0.2 0.12
>>> clf.fit(digits.data[:-1], digits.target[:-1])
SVC(C=100.0, cache_size=200, class_weight=None, coef0=0.0,
    decision_function_shape='ovr', degree=3, gamma=0.001, kernel='rbf',
    max_iter=1, probability=False, random_state=None, shrinking=True,
    tol=0.001, verbose=False)

```

Gambar 1.14 Memanggil Classifier Tanpa Baris Terakhir

```

    tol=0.001, verbose=False)
>>> clf.predict(digits.data[-1:])
array([8])

```

Gambar 1.15 Memprediksi Nilai Baru

```

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

```

Gambar 1.16 Hasil Pengujian Classifier

gan sintaks Python [: -1], yang menghasilkan array baru yang berisi semua kecuali item terakhir dari `digits.data` terlihat pada gambar 1.14.

6. Pada penggalan skrip dibawah, ini menunjukkan prediksi nilai baru menggunakan gambar terakhir dari `digits.data`. Dengan prediksi akan menentukan gambar dari set pelatihan yang paling cocok dengan gambar terakhir terlihat pada gambar 1.15.

1.3.4 Model Persistence

1. Pada Python Shell ketikan `'from sklearn import svm'` artinya akan mengimport sebuah Support Vector Machine(SVM) yang merupakan algoritma classification yang akan diambil dari Scikit-Learn.
2. Kemudian, lanjutkan dengan `'from sklearn import datasets'` yang artinya akan mengambil package datasets dari Scikit-Learn.
3. ketikan, `clf = svm.SVC(gamma='scale')` berfungsi untuk mendeklarasikan suatu value yang bernama `clf` yang berisi `gamma`. Parameter `gamma` menentukan seberapa jauh pengaruh dari satu contoh training.
4. Ketikan, `X, y = iris.data, iris.target`, artinya `X` sebagai data iris, dan `y` merupakan larik target.
5. Ketikan, `clf.fit(X, y)` berfungsi untuk melakukan pengujian classifier. hasilnya seperti pada gambar 1.16.

```
clf=0.001, verbose=False
>>> import pickle
>>> s = pickle.dumps(clf)
```

Gambar 1.17 Hasil Pengujian Classifier

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

Gambar 1.18 Pickle Pada Python

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

Gambar 1.19 Pengujian Classifier Pickle

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

Gambar 1.20 Penggunaan Joblib

6. Dari gambar 1.17 dapat dijelaskan bahwa akan mengimport Pickle dari Python. Pickle digunakan untuk serialisasi dan de-serialisasi struktur objek Python. Objek apa pun dengan Python dapat di-Pickle sehingga dapat disimpan di disk. kemudian menyimpan data objek ke file CLF sebelumnya dengan menggunakan function `pickle.dumps(clf)`.
7. Setelah mengetikkan fungsi fungsi diatas, selanjutnya ketikkan '`clf2 = pickle.loads(s)`' yang artinya `pickle.loads` digunakan untuk memuat data pickle dari string byte. "S" dalam loads mengacu pada fakta bahwa dalam Python 2, data dimuat dari string, seperti pada gambar 1.18
8. Pada gambar 1.19 dilakukan pengujian nilai baru dengan menggunakan '`clf2.predict(X[0:1])`' dengan target asumsinya (0,1) hasilnya berbentuk array.
9. Dalam kasus khusus scikit-learn, mungkin lebih menarik untuk menggunakan joblib (dump dan load) untuk menggantikan Pickle, yang lebih efisien pada data besar tetapi hanya bisa di Pickle ke disk dan tidak ke string. untuk menggunakan Joblib pertama ketikkan '`from joblib import dump, load`' yang artinya akan Merekonstruksi objek Python dari file yang sudah ada.

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

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

Gambar 1.21 Deklarasi Numpy

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

Gambar 1.22 Contoh Type Casting

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

Gambar 1.23 Menggunakan FitTransform

1.3.5 Conventions

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

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

2. Pada gambar 1.22 dapat dijelaskan bahwa :

`rng = np.random.RandomState(0)`, digunakan untuk menginisialisasikan random number generator. `X = rng.rand(10, 2000)` artinya akan merandom value antara 10 sampai 2000. `X = np.array(X, dtype='float32')` Array numpy terdiri dari buffer memori "mentah" yang diartikan sebagai array melalui 'views'. Anda dapat menganggap semua array numpy sebagai tampilan. Mendeklarasikan X sebagai float32.

3. Dalam contoh ini, X adalah float32, yang dilemparkan ke float64 oleh `fittransform (X)`.
4. Target regresi dilemparkan ke float64 dan target klasifikasi dipertahankan.

`list(clf.predict(irisdata[:3]))`, akan memprediksi 3 data dari iris. `clf.fit irisdata, iristargetnames[iristarget]` menguji classifier dengan ada targetnya yaitu irisnya sendiri. `list(clf.predict(irisdata[:3]))`, setelah diuji maka akan muncul datanya seperti dibawah ini Di sini, prediksi pertama () mengembalikan array inte-

```

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

```

Gambar 1.24 Regresi Yang Dilempar

```

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

```

Gambar 1.25 Refitting dan Memperbaharui Parameter

ger, karena `iris.target` (array integer) yang digunakan sesuai. Prediksi kedua () mengembalikan array string, karena `iris.target_names` cocok.

5. Refitting dan Memperbaharui Parameter

`y = rngbinomial(1, 0.5, 100)`, random value dengan angka binomial atau suku dua untuk `y` `clf.setparams(kernel='linear')` `fit(X, y)` mengubah kernel default menjadi linear `clf.setparams(kernel='rbf', gamma='scale')` `fit(X, y)` Di sini, kernel default rbf pertama kali diubah menjadi linear melalui `SVC.setparams()` setelah estimator dibuat, dan diubah kembali ke rbf untuk mereparasi estimator dan membuat prediksi kedua.

6. MultiClass VS MultiLabel Classifier

`from sklearn.multiclass import OneVsRestClassifier`, adalah ketika kita ingin melakukan klasifikasi multiclass atau multilabel dan baik untuk menggunakan `OneVsRestClassifier` per kelas. Untuk setiap classifier, kelas tersebut dipasang terhadap semua kelas lainnya. (Ini cukup jelas dan itu berarti bahwa masalah klasifikasi multiclass / multilabel dipecah menjadi beberapa masalah klasifikasi biner). `from sklearn.preprocessing import LabelBinarizer`, adalah kelas utilitas untuk membantu membuat matriks indikator label dari daftar label multi-kelas.

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

Gambar 1.26 MultiClass Classifier

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

Gambar 1.27 MultiClass Classifier biner 2D

```
>>> from sklearn.preprocessing import MultiLabelBinarizer
>>> y = [[0, 1], [0, 2], [1, 3], [0, 2, 3], [2, 4]]
>>> y = MultiLabelBinarizer().fit_transform(y)
>>> clf.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]])
>>>
```

Gambar 1.28 MultiLabel Classifier

Dalam gambar dibawah, classifier cocok pada array 1d label multiclass dan oleh karena itu metode predict () memberikan prediksi multiclass yang sesuai.

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

BAB 2

MEMBANGUN MODEL PREDIKSI

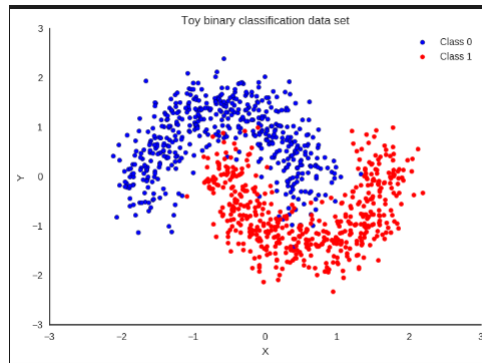
2.1 Teori

2.1.1 Binary Classification

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

2.1.2 Supervised Learning, Unsupervised Learning dan Clustering

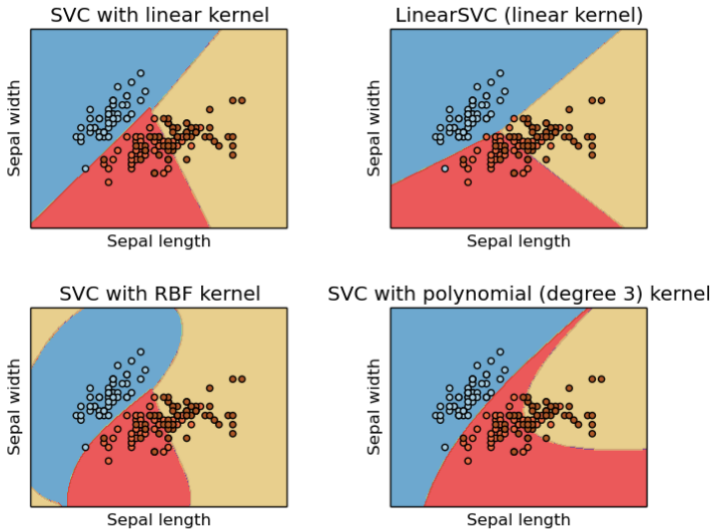
1. Supervised learning adalah tugas pembelajaran mesin untuk mempelajari suatu fungsi yang memetakan input ke output berdasarkan contoh pasangan input-output. Ini menyimpulkan fungsi dari data pelatihan berlabel yang terdiri dari serangkaian contoh pelatihan. Dalam pembelajaran yang diawasi, setiap contoh



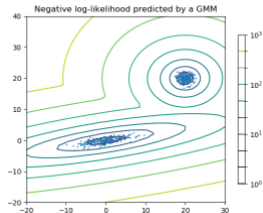
Gambar 2.1 Binary Classification

adalah pasangan yang terdiri dari objek input (biasanya vektor) dan nilai output yang diinginkan (juga disebut sinyal pengawas). Algoritma pembelajaran yang diawasi menganalisis data pelatihan dan menghasilkan fungsi yang disimpulkan, yang dapat digunakan untuk memetakan contoh-contoh baru. Skenario optimal akan memungkinkan algoritma menentukan label kelas dengan benar untuk instance yang tidak terlihat. Ini membutuhkan algoritma pembelajaran untuk menggeneralisasi dari data pelatihan untuk situasi yang tidak terlihat dengan cara yang "masuk akal" (lihat bias induktif). Tugas paralel dalam psikologi manusia dan hewan sering disebut sebagai pembelajaran konsep. Contoh dibawah yaitu Supervised Learning dengan SVC.

2. Unsupervised learning adalah istilah yang digunakan untuk pembelajaran bahasa Ibrani, yang terkait dengan pembelajaran tanpa guru, juga dikenal sebagai organisasi mandiri dan metode pemodelan kepadatan probabilitas input. Analisis cluster sebagai cabang pembelajaran mesin yang mengelompokkan data yang belum diberi label, diklasifikasikan atau dikategorikan. Alih-alih menanggapi umpan balik, analisis klaster mengidentifikasi kesamaan dalam data dan bereaksi berdasarkan ada tidaknya kesamaan di setiap potongan data baru. Berikut merupakan contoh Unsupervised Learning dengan Gaussian mixture models.
3. Cluster analysis or clustering adalah tugas pengelompokan sekumpulan objek sedemikian rupa sehingga objek dalam kelompok yang sama (disebut kluster) lebih mirip (dalam beberapa hal) satu sama lain daripada pada kelompok lain (kluster). Ini adalah tugas utama penambangan data eksplorasi, dan teknik umum untuk analisis data statistik, yang digunakan di banyak bidang, termasuk pembelajaran mesin, pengenalan pola, analisis gambar, pengambilan informasi, bioinformatika, kompresi data, dan grafik komputer. Analisis Cluster sendiri bukan merupakan salah satu algoritma spesifik, tetapi tugas umum yang harus dipecahkan. Ini dapat dicapai dengan berbagai algoritma yang berbeda secara signifikan dalam pemahaman mereka tentang apa yang merupakan sebuah clus-

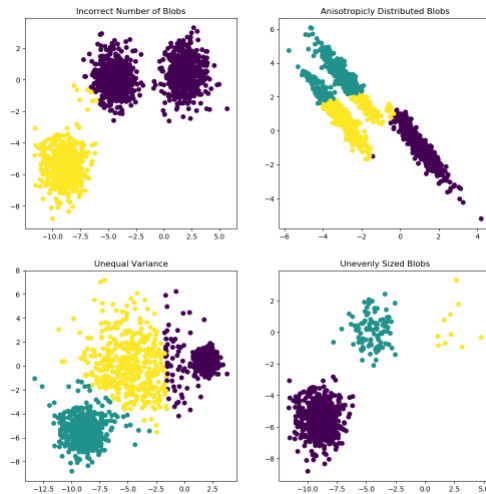


Gambar 2.2 Supervised Learning



Gambar 2.3 Unsupervised Learning

ter dan bagaimana cara menemukannya secara efisien. Gagasan populer mengenai cluster termasuk kelompok dengan jarak kecil antara anggota cluster, area padat ruang data, interval atau distribusi statistik tertentu. Clustering karena itu dapat dirumuskan sebagai masalah optimasi multi-objektif. Algoritma pengelompokan dan pengaturan parameter yang sesuai (termasuk parameter seperti fungsi jarak yang akan digunakan, ambang kepadatan atau jumlah cluster yang diharapkan) tergantung pada set data individual dan penggunaan hasil yang dimaksudkan. Analisis kluster bukan merupakan tugas otomatis, tetapi proses berulang penemuan pengetahuan atau optimasi multi-objektif interaktif yang melibatkan percobaan dan kegagalan. Seringkali diperlukan untuk memodifikasi praproses data dan parameter model hingga hasilnya mencapai properti yang diinginkan.



Gambar 2.4 Cluster

2.1.3 Evaluasi dan Akurasi

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

```
>>> from sklearn import svm, datasets
>>> from sklearn.model_selection import cross_val_score
>>> iris = datasets.load_iris()
>>> X, y = iris.data, iris.target
>>> clf = svm.SVC(gamma='scale', random_state=0)
>>> cross_val_score(clf, X, y, scoring='recall_macro',
...                 cv=5)
array([0.96..., 0.96..., 0.96..., 0.93..., 1.        ])
>>> model = svm.SVC()
>>> cross_val_score(model, X, y, cv=5, scoring='wrong_choice')
```

Gambar 2.5 Evaluasi dan Akurasi

2.1.4 Cara membuat dan membaca confusion matrix

1. Cara membuat dan membaca confusion matrix :
 - 1) Tentukan pokok permasalahan dan atributnya, misal gaji dan listik.
 - 2) Buat pohon keputusan
 - 3) Lalu data testingnya

- 4) Lalu mencari nilai a, b, c, dan d. Semisal a = 5, b = 1, c = 1, dan d = 3.
- 5) Selanjutnya mencari nilai recall, precision, accuracy, serta dan error rate.

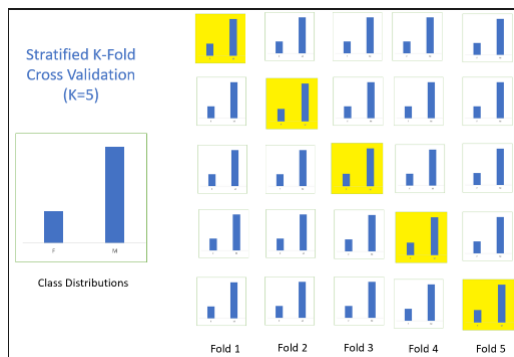
2. Berikut adalah contoh dari confusion matrix :

- Recall = $3/(1+3) = 0,75$
- Precision = $3/(1+3) = 0,75$
- Accuracy = $(5+3)/(5+1+1+3) = 0,8$
- Error Rate = $(1+1)/(5+1+1+3) = 0,2$

2.1.5 K-fold cross validation

1. Cara kerja K-fold cross validation :

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

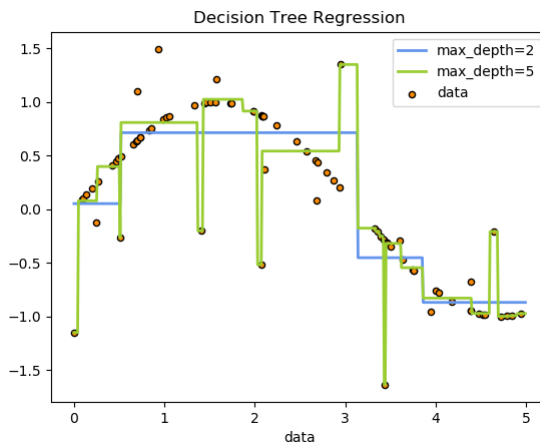


Gambar 2.6 K-fold cross validation

2.1.6 Decision Tree

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

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



Gambar 2.7 Decision Tree

2.1.7 Information Gain dan Entropi

1. Information gain didasarkan pada penurunan entropi setelah dataset dibagi pada atribut. Membangun decision tree adalah semua tentang menemukan atribut yang mengembalikan perolehan informasi tertinggi (mis., Cabang yang paling homogen).
2. Entropi adalah ukuran keacakan dalam informasi yang sedang diproses. Semakin tinggi entropi, semakin sulit untuk menarik kesimpulan dari informasi itu. Membalik koin adalah contoh tindakan yang memberikan informasi yang acak. Untuk koin yang tidak memiliki afinitas untuk kepala atau ekor, hasil dari sejumlah lemparan sulit diprediksi. Mengapa? Karena tidak ada hubungan antara membalik dan hasilnya. Inilah inti dari entropi.

```

from scipy.stats import entropy
import numpy as np

def information_gain(X, y):

    def _entropy(labels):
        counts = np.bincount(labels)
        return entropy(counts, base=None)

    def _ig(x, y):
        # indices where x is set/not set
        x_set = np.nonzero(x)[1]
        x_not_set = np.delete(np.arange(x.shape[1]), x_set)

        h_x_set = _entropy(y[x_set])
        h_x_not_set = _entropy(y[x_not_set])

        return entropy_full - (((len(x_set) / f_size) * h_x_set)
                               + ((len(x_not_set) / f_size) * h_x_not_set))

    entropy_full = _entropy(y)

    f_size = float(X.shape[0])

    scores = np.array([_ig(x, y) for x in X.T])
    return scores

```

Gambar 2.8 Information gain

```

import matplotlib.pyplot as plt
import numpy as np

from skimage import data
from skimage.util import img_as_ubyte
from skimage.filters.rank import entropy
from skimage.morphology import disk

# First example: object detection.

noise_mask = np.full((128, 128), 28, dtype=np.uint8)
noise_mask[32:-32, 32:-32] = 30

noise = (noise_mask * np.random.random(noise_mask.shape) - 0.5 *
         noise_mask).astype(np.uint8)
img = noise + 128

entr_img = entropy(img, disk(10))

fig, (ax0, ax1, ax2) = plt.subplots(nrows=1, ncols=3, figsize=(10, 4))

ax0.imshow(noise_mask, cmap='gray')
ax0.set_xlabel("Noise mask")
ax1.imshow(img, cmap='gray')
ax1.set_xlabel("Noisy image")
ax2.imshow(entr_img, cmap='viridis')
ax2.set_xlabel("Local entropy")

fig.tight_layout()

```

Gambar 2.9 Entropi

2.2 Pratikum

2.2.1 Scikit-Learn

1. # load dataset (student mat pakenya)


```

import pandas as pd
durian = pd.read_csv('student-mat.csv', sep=';')

```



```
len(d)
```

Codingan diatas digunakan untuk mengimport atau memanggil module pandas sebagai pd. Kemudian mendefinisikan variabel "durian" yang akan memanggil dataset yang didapatkan dari data csv. Jika skrip dijalankan di Spyder, hasilnya seperti berikut

```
In [25]: import pandas as pd
...: durian = pd.read_csv('student-mat.csv', sep=';')
...: len(durian)
Out[25]: 395
```

Gambar 2.10 Loading Dataset

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

ada bagian ini mendeklarasikan pass/fail nya data berdasarkan G1+G2+G3. Dengan ketentuan nilai pass nya yaitu sama dengan 30. kemudian pada variabel durian dideklarasikan jika baris dengan G1+G2+G3 ditambahkan, dan hasilnya sama dengan 35 maka axisnya 1. ketika dijalankan hasilnya seperti berikut

```
In [28]: durian['pass'] = durian.apply(lambda row: 1 if (row['G1']+row['G2']+row['G3'])
>= 35 else 0, axis=1)
...: durian = durian.drop(['G1', 'G2', 'G3'], axis=1)
...: durian.head()
Out[28]:
```

	school	sex	age	address	famsize	...	Dalc	Walc	health	absences	pass
0	GP	F	18	U	GT3	...	1	1	3	6	0
1	GP	F	17	U	GT3	...	1	1	3	4	0
2	GP	F	15	U	LE3	...	2	3	3	10	0
3	GP	F	15	U	GT3	...	1	1	5	2	1
4	GP	F	16	U	GT3	...	1	2	5	4	0

Gambar 2.11 Generate Binary Label

3. # use one-hot encoding on categorical columns
durian = pd.get_dummies(durian, columns=['sex', 'school', 'a
'famsize',
'Pstatus', 'Mjob', 'Fjob',
'reason', 'guardian', 'school',
'famsup', 'paid', 'activities',
'nursery', 'higher', 'intern
'romantic'])
durian.head()

One-hot encoding adalah proses di mana variabel kategorikal dikonversi menjadi bentuk yang dapat disediakan untuk algoritma ML untuk melakukan pekerjaan yang lebih baik dalam prediksi. Karena saya memuat data menggunakan

panda, disini menggunakan fungsi panda `pd.get_dummies` untuk jenis kelamin , sekolah, alamat dll. Metode `head` ini digunakan untuk mengembalikan baris n atas 5 secara default dari frame atau seri data. hasilnya seperti berikut

```
In [29]: durian = pd.get_dummies(durian, columns=['sex', 'school', 'address', 'famsize',
...:      'Pstatus', 'Mjob', 'Fjob',
...:      'reason', 'guardian', 'schoolsup', 'famsup',
...:      'paid', 'activities',
...:      'nursery', 'higher', 'internet', 'romantic'])
...: durian.head()
Out[29]:
```

	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]

Gambar 2.12 One-hot Encoding

4. # shuffle rows

```
durian = durian.sample(frac=1)
# split training and testing data
durian_train = d[:500]
durian_test = d[500:]
```

```
durian_train_att = durian_train.drop(['pass'], axis=1)
durian_train_pass = durian_train['pass']
```

```
durian_test_att = durian_test.drop(['pass'], axis=1)
durian_test_pass = durian_test['pass']
```

```
durian_att = durian.drop(['pass'], axis=1)
durian_pass = d['pass']
```

number of passing students in whole dataset:

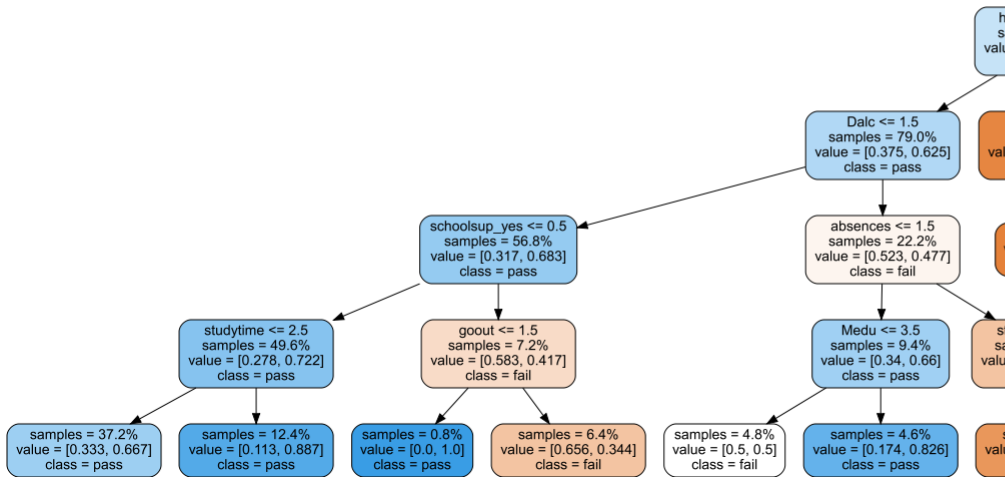
```
import numpy as np
print("Passing: %d out of %d (%.2f%%)" % (np.sum(d_pass), len(d_pass),
100*float(np.sum(d_pass)) / len(d_pass)))
```

Sammple digunakan untuk mengembalikan sampel acak item dari objek. Pada bagian tersebut, terdapat train dan test yaing digunakan untuk untuk membagi train, test dan kemudian membagi lagi train ke validasi dan test.

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

5. # fit a decision tree

```
from sklearn import tree
mangga = tree.DecisionTreeClassifier(criterion="entropy", ma
mangga = mangga.fit(durian_train_att, durian_train_pass)
```

Gambar 2.15 Fit Decision Tree

```
filled=True, rounded=True)
```

TREEEXPORTGRAPHVIZ merupakan fungsi yang menghasilkan representasi Graphviz dari decision tree, yang kemudian ditulis ke outfile. Disini akan menyimpan classifiernya, akan meng ekspor file student performance jika salah akan mengembalikan nilai fail.

```
In [34]: tree.export_graphviz(mangga, out_file="student-performance.dot", label="all",
impurity=False, proportion=True,
...,
feature_names=list(durian_train_att), class_names=["fail",
"pass"],
...,
filled=True, rounded=True)
```

Gambar 2.16 Fit Decision Tree

8. `mangga.score(durian_test_att, durian_test_pass)`

Score juga disebut prediksi, dan merupakan proses menghasilkan nilai berdasarkan model pembelajaran mesin yang terlatih, diberi beberapa data input baru. Nilai atau skor yang dibuat dapat mewakili prediksi nilai masa depan, tetapi mereka juga mungkin mewakili kategori atau hasil yang mungkin. Jadi disini Mangga akan memprediksi nilai dari durian test att dan test pass. Hasilnya seperti dibawah ini

9. `from sklearn.model_selection import cross_val_score`
`angka = cross_val_score(mangga, durian_att, durian_pass, cv`

```
In [6]: mangga.score(durian_test_att, durian_test_pass)
Out[6]: 0.7334360554699538
```

Gambar 2.17 Score

```
# show average score and +/- two standard deviations away
#(covering 95% of scores)
```

```
print("Accuracy: %0.2f (+/- %0.2f)" % (nangka.mean(), nangka
```

Skrip ini akan mengevaluasi score dengan validasi silang. Dimana variabel nangka berisikan crossvalscore yang merupakan fungsi pembantu pada estimator dan dataset. Kemudian akan menampilkan score rata rata dan kurang lebih dua standar deviasi yang mencakup 95 persen score. dengan menggunakan perintah print hasil yang didapatkan sebagai berikut

```
In [12]: from sklearn.model_selection import cross_val_score
...: nangka = cross_val_score(mangga, durian_att, durian_pass,
cv=5)
...: # show average score and +/- two standard deviations away
(covering 95% of scores)
...: print("Accuracy: %0.2f (+/- %0.2f)" % (nangka.mean(),
nangka.std() * 2))
Accuracy: 0.70 (+/- 0.08)
```

Gambar 2.18 Cross Val Score

```
10. for max_depth in range(1, 20):
    mangga = tree.DecisionTreeClassifier(criterion="entropy")
    max_depth=max_depth)
    nangka = cross_val_score(mangga, durian_att, durian_pass)
    print("Max depth: %d, Accuracy: %0.2f (+/- %0.2f)" %
(max_depth, nangka.mean(), nangka.std() * 2)
)
```

Pada skrip ini menunjukkan seberapa dalam tree itu. Semakin dalam tree, semakin banyak perpecahan yang dimilikinya dan menangkap lebih banyak informasi tentang data. variabel mangga akan mendefinisikan tree nya yang kemudian variabel nangka akan mengevaluasi score dengan validasi silang. disini mendefinisikan decision tree dengan kedalaman mulai dari 1 hingga 20 dan merencanakan pelatihan dan menguji skor auc. Jika di run hasilnya seperti berikut

```
11. depth_acc = np.empty((19,3), float)
i = 0
for max_depth in range(1, 20):
    mangga = tree.DecisionTreeClassifier(criterion="entropy")
    max_depth=max_depth)
    nangka = cross_val_score(t, d_att, d_pass, cv=5)
    depth_acc[i,0] = max_depth
    depth_acc[i,1] = nangka.mean()
```

```

In [89]: for max_depth in range(1, 20):
...:     mangga = tree.DecisionTreeClassifier(criterion="entropy",
max_depth=max_depth)
...:     scores = cross_val_score(mangga, durian_att, durian_pass, cv=5)
...:     print("Max depth: %d, Accuracy: %0.2f (+/- %0.2f)" % (max_depth,
scores.mean(), scores.std() * 2))
Max depth: 1, Accuracy: 0.58 (+/- 0.01)
Max depth: 2, Accuracy: 0.56 (+/- 0.11)
Max depth: 3, Accuracy: 0.54 (+/- 0.10)
Max depth: 4, Accuracy: 0.57 (+/- 0.10)
Max depth: 5, Accuracy: 0.55 (+/- 0.08)
Max depth: 6, Accuracy: 0.55 (+/- 0.05)
Max depth: 7, Accuracy: 0.58 (+/- 0.08)
Max depth: 8, Accuracy: 0.54 (+/- 0.05)
Max depth: 9, Accuracy: 0.56 (+/- 0.05)
Max depth: 10, Accuracy: 0.55 (+/- 0.09)
Max depth: 11, Accuracy: 0.56 (+/- 0.07)
Max depth: 12, Accuracy: 0.55 (+/- 0.10)
Max depth: 13, Accuracy: 0.55 (+/- 0.09)
Max depth: 14, Accuracy: 0.55 (+/- 0.05)
Max depth: 15, Accuracy: 0.55 (+/- 0.07)
Max depth: 16, Accuracy: 0.55 (+/- 0.07)
Max depth: 17, Accuracy: 0.54 (+/- 0.11)
Max depth: 18, Accuracy: 0.55 (+/- 0.05)
Max depth: 19, Accuracy: 0.53 (+/- 0.07)

```

Gambar 2.19 Max Depth

```

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

```

depth_acc

Depth acc akan membuat array kosong dengan mengembalikan array baru dengan bentuk dan tipe yang diberikan, tanpa menginisialisasi entri. Dengan 19 sebagai bentuk array kosong, 3 sebagai output data-type dan float urutan kolom-utama (gaya Fortran) dalam memori. variabel mangga yang akan melakukan split score dan angka akan mengvalidasi score secara silang. dan pada akhirnya angka std yaitu menghitung standar deviasi dari data yang diberikan (elemen array) di sepanjang sumbu yang ditentukan (jika ada), hasilnya sebagai berikut

```

...: for max_depth in range(1, 20):
...:     mangga = tree.DecisionTreeClassifier(criterion="entropy",
max_depth=max_depth)
...:     scores = cross_val_score(mangga, durian_att, durian_pass, cv=5)
...:     depth_acc[i,0] = max_depth
...:     depth_acc[i,1] = scores.mean()
...:     depth_acc[i,2] = scores.std() * 2
...:     i += 1
...:
...:
...: depth_acc
Out[90]:
array([[1.00000000e+00, 5.79751704e-01, 6.30768599e-03],
[2.00000000e+00, 5.62381532e-01, 1.05526097e-01],
[3.00000000e+00, 5.39466082e-01, 9.97942033e-02],
[4.00000000e+00, 5.67281727e-01, 1.03645308e-01],
[5.00000000e+00, 5.46902791e-01, 8.51209423e-02],
[6.00000000e+00, 5.51870834e-01, 5.25904737e-02],
[7.00000000e+00, 5.56902791e-01, 5.89805597e-02],
[8.00000000e+00, 5.39051444e-01, 6.59884566e-02],
[9.00000000e+00, 5.49018987e-01, 9.31993778e-02],
[1.00000000e+01, 5.69369523e-01, 7.14387282e-02],
[1.10000000e+01, 5.59371146e-01, 6.92276696e-02],
[1.20000000e+01, 5.46616358e-01, 7.98064960e-02],
[1.30000000e+01, 5.59339500e-01, 8.47496130e-02],
[1.40000000e+01, 5.59337877e-01, 5.37281193e-02],
[1.50000000e+01, 5.62030185e-01, 8.37479468e-02],
[1.60000000e+01, 5.54241318e-01, 8.45354250e-02],
[1.70000000e+01, 5.34149627e-01, 7.60417853e-02],
[1.80000000e+01, 5.36583901e-01, 3.95368822e-02],
[1.90000000e+01, 5.61008530e-01, 5.71807354e-02]])

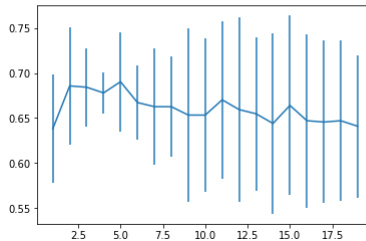
```

Gambar 2.20 Depth in Range

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

Mengimpor librari dari matplotlib yaitu pyplot sebagai plt fig dan ax menggunakan subplots untuk membuat gambar dan satu set subplot. axerrorbar akan membuat error bar kemudian grafik akan ditampilkan menggunakan show. Grafiknya seperti berikut

```
In [45]: import matplotlib.pyplot as plt
...: fig, ax = plt.subplots()
...: ax.errorbar(depth_acc[:,0], depth_acc[:,1], yerr=depth_acc[:,2])
...: plt.show()
```



Gambar 2.21 Matplotlib

2.2.2 Penanganan Error

2.2.2.1 Error Graphviz

1. Berikut ini merupakan error yang didapatkan saat menjalankan graphviz pada Spyder

```
File "E:\Anaconda2\lib\site-packages\graphviz\backend.py", line 150, in run
    raise ExecutableNotFound(cmd)

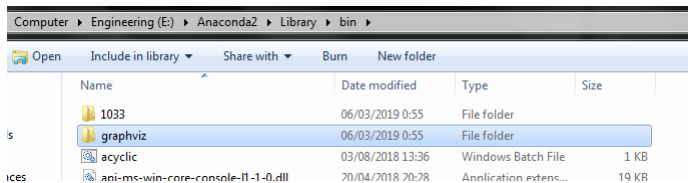
ExecutableNotFound: failed to execute ['dot', '-Tsvg'], make sure the Graphviz
executables are on your systems' PATH

Out[9]: <graphviz.files.Source at 0xd7df2b0>
```

Gambar 2.22 Error Graphviz

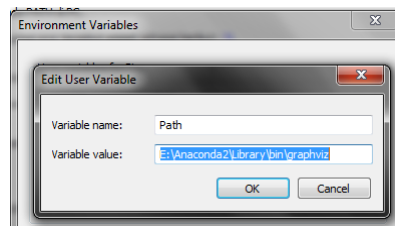
2. Pada gambar diatas kode erornya adalah ExecutableNotFound failed to execute dot Tsvg. Error ini terjadi karena tidak terdaptarnya environment variable dari Graphviz pada PATH di PC.
3. Solusi yang bisa dilakukan untuk mengatasi error tersebut adalah sebagai berikut :

- Buka Folder dimana Anaconda2 terinstall. Disini Anacondanya terinstal di folder E
- Kemudian, pada Anaconda2 buka Library/Bin/graphviz



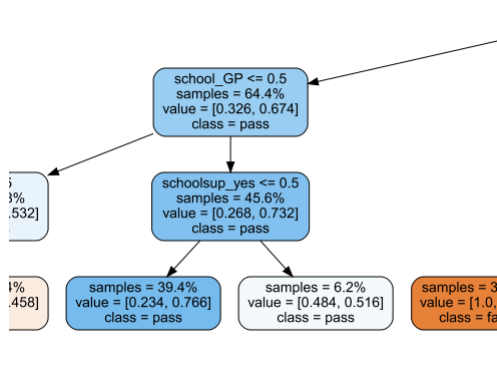
Gambar 2.23 Folder Graphviz

- Salin alamat tersebut, kemudian buka Environment Variable dan Edit.
- Pada bagian PATH pilih edit, dan salin alamat tersebut seperti berikut :



Gambar 2.24 Menambahkan Graphviz kePATH

- Klik ok, kemudian restart Spyder dan jalankan kembali Skrip, maka hasilnya akan seperti berikut dan error berhasil diselesaikan



Gambar 2.25 Evaluasi Error

2.2.2.2 Error File Not Exist

1. Berikut ini merupakan eror yang didapatkan saat menjalankan file csv sebagai dataset

```

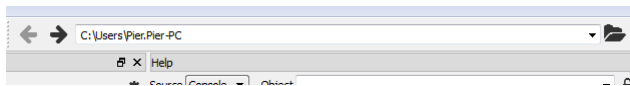
import pandas as pd
durian = pd.read_csv('student-mat.csv', sep=';')
IOError: File student-mat.csv does not exist

```

Gambar 2.26 Error File Not Exist

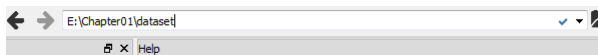
2. Pada gambar diatas kode erornya adalah IOError File student csv does not exist. Error ini terjadi karena file yang dituju tidak berada didalam file yang salam dengan skrip ataupun filenya belum didefinisikan.
3. Solusi yang bisa dilakukan untuk mengatasi eror tersebut adalah sebagai berikut :

- Buka Spyder, kemudian pada pojok kanan atas ada kolom sebagai berikut



Gambar 2.27 Kolom Direktori

- Pada kolom tersebut buka folder dimana file csv atau datasetnya tersimpan. Pada tutorial ini alamat foldernya sebagai berikut



Gambar 2.28 Memasuki Direktori Dataset

- Kemudian jalankan lagi skrip tadi, akan berhasil seperti dibawah ini dan eror terselesaikan

```

In [17]: import pandas as pd
...: durian = pd.read_csv('student-mat.csv', sep=';')
...: len(durian)
Out[17]: 395

```

Gambar 2.29 Evaluasi Error

BAB 3

PREDIKSI DENGAN RANDOM FOREST

3.1 TEORI

3.1.1 Random Forest

3.1.1.1 Pengertian Random Forest adalah konstruk data yang diterapkan pada machine learning yang mengembangkan sejumlah besar pohon keputusan acak yang menganalisis sekumpulan variabel. Jenis algoritma ini membantu meningkatkan cara teknologi menganalisis data yang kompleks. Juga merupakan algoritma machine learning yang fleksibel, mudah digunakan, bahkan tanpa penyetelan hyperparameter, dengan hasil yang baik. Ini juga merupakan salah satu algoritma yang paling banyak digunakan, karena kesederhanaan dan faktanya dapat digunakan untuk tugas klasifikasi dan regresi. Dibawah ini merupakan salah satu ilustrasi penggunaan Random Forest yang saya lakukan untuk memprediksi apakah uang kertas bank otentik atau tidak berdasarkan pada empat atribut. Ini merupakan hasil dari ilustrasi Random Forest pada Spyder

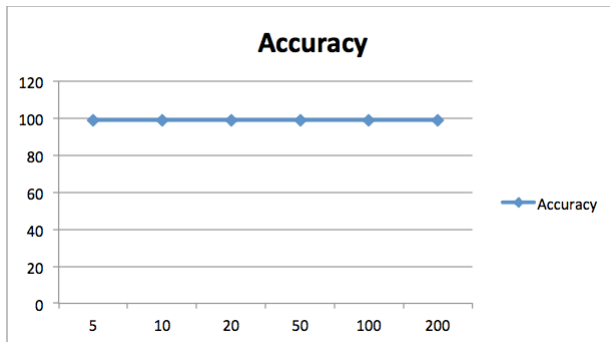
Setelah di plotting hasilnya seperti berikut

```
In [14]: runfile('E:/Chapter01/dataset/tugas3.py',
wdir='E:/Chapter01/dataset')
[[155
  2]]
[ 1 117]]
```

	precision	recall	f1-score	support
0	0.99	0.99	0.99	157
1	0.98	0.99	0.99	118
micro avg	0.99	0.99	0.99	275
macro avg	0.99	0.99	0.99	275
weighted avg	0.99	0.99	0.99	275

0.9890909090909091

Gambar 3.1 Random Forest Spyder



Gambar 3.2 Random Forest Graphic

3.1.2 Dataset

3.1.2.1 Pengertian Dataset Dataset adalah kumpulan data. Paling umum satu data set sesuai dengan isi tabel database tunggal, atau matriks data statistik tunggal, di mana setiap kolom tabel mewakili variabel tertentu, dan setiap baris sesuai dengan anggota tertentu dari dataset yang dipertanyakan.

3.1.3 Cara Membaca Dataset Dan Arti Setiap File Dan Isi Field Masing Masing File

1. Gunakan library Pandas pada python untuk dapat membaca dataset dengan format text file.
2. Setelah itu, buat variabel baru "dataset" yang berisikan perintah untuk membaca file csv. seperti berikut

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

Gambar 3.3 Dataset Pandas

Pada gambar diatas dapat dijelaskan bahwa :

- Memanggil Librari Panda untuk membaca dataset
- Membuat variabel "Dataset" yang berisikan pdreadcsv untuk membaca dataset. Pada contoh ini menggunakan txt tapi tetap bisa membaca datasetnya, mengapa? Karena pada saat dijalankan librari panda secara otomatis akan mengubah data dalam bentuk text file ke format csv.

3. Setelah di run akan muncul hasil seperti berikut :

dataset - DataFrame							
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

Gambar 3.4 Dataset Pandas

Pertama tama gambar diatas merupakan dataset yang digunakan untuk evaluasi mobil setelah dibuat untuk mengecek dan menguji induksi konstruktif dan metode penemuan struktur. Datasetnya dapat didapatkan dari laman <https://archive.ics.uci.edu/>. Penjelasan dari isi field diatas adalah sebagai berikut :

- Atribut Index merupakan atribut otomatis untuk penomoran data yang 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 kedalam 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 .

3.1.4 Cross Validation

Cross Validation adalah prosedur resampling yang digunakan untuk mengevaluasi model machine learning pada sampel data yang terbatas. Prosedur ini memiliki parameter tunggal yang disebut k yang mengacu pada jumlah grup tempat sampel data yang akan dibagi. Karena itu, prosedur ini sering disebut k-fold cross-validation. Proses penentuan apakah hasil numerik yang mengukur hubungan yang dihipotesiskan antar variabel, dapat diterima sebagai deskripsi data, dikenal sebagai Validationi. Umumnya, estimasi kesalahan untuk model dibuat setelah training, lebih dikenal sebagai evaluasi residu. Dalam proses ini, estimasi numerik dari perbedaan respons yang diprediksi dan yang asli dilakukan, juga disebut kesalahan training. Namun, ini hanya memberi kita gambaran tentang seberapa baik model kita pada data yang digunakan untuk melatihnya. Sekarang mungkin bahwa model tersebut kurang cocok atau overfitting data. Jadi, masalah dengan teknik evaluasi ini adalah bahwa itu tidak memberikan indikasi seberapa baik pelajar akan menggeneralisasi ke set data independen / tidak terlihat. Model ini dikenal sebagai Cross Validation.

3.1.5 Arti Score 44% Pada Random Forest, 27% Pada Decision Tree Dan 29% Dari SVM

Itu merupakan presentase keakurasian prediksi yang dilakukan pada saat testing menggunakan label pada dataset yang digunakan. Score merupakan mendefinisikan aturan evaluasi model. Maka pada saat dijalankan akan muncul persentase tersebut yang menunjukkan keakurasian atau keberhasilan dari prediksi yang dilakukan. Jika menggunakan Random Forest maka hasilnya 40% , jika menggunakan Decision Tree hasil prediksinya yaitu 27% dan pada SVM 29% .

3.1.6 Confusion Matriks

3.1.6.1 Confusion Matriks Dan Contohnya Perhitungan Confusion Matriks dapat dilakukan sebagai berikut. Disini saya menggunakan data yang dibuat sendiri untuk menampilkan data aktual dan prediksi.

- Import librari Pandas, Matplotlib, dan Numpy.
- Buat variabel `y_actu` yang berisikan data aktual.
- Buat variabel `y_pred` berisikan data yang akan dijadikan sebagai prediksi.
- Buat variabel `df_confusion` yang berisikan crosstab untuk membangun tabel tabulasi silang yang dapat menunjukkan frekuensi kemunculan kelompok data tertentu.
- Pada variabel `df_confusion` definisikan lagi nama baris yaitu Actual dan kolomnya Predicted
- Kemudian definisikan suatu fungsi yang diberi nama `plot_confusion_matrix` yang berisikan pendefinisian confusion matrix dan juga akan di plotting. untuk code lengkapnya sebagai berikut

```
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd

y_actu = pd.Series([2, 0, 2, 2, 0, 1, 1, 2, 2, 0, 1, 2], name='Actual')
y_pred = pd.Series([0, 0, 2, 1, 0, 2, 1, 0, 2, 0, 2, 2], name='Predicted')
df_confusion = pd.crosstab(y_actu, y_pred)

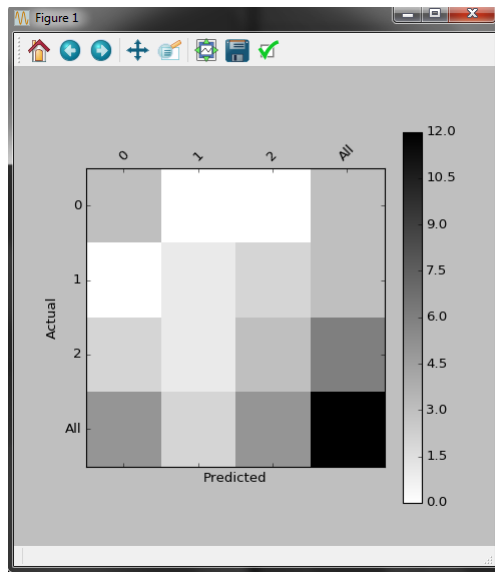
df_confusion = pd.crosstab(y_actu, y_pred, rownames=['Actual'], colnames=['Predicted'])

def plot_confusion_matrix(df_confusion, title='Confusion matrix', cmap=plt.cm.Blues):
    plt.imshow(df_confusion, cmap=cmap) # imshow
    plt.title(title)
    plt.colorbar()
    tick_marks = np.arange(len(df_confusion.columns))
    plt.xticks(tick_marks, df_confusion.columns, rotation=45)
    plt.yticks(tick_marks, df_confusion.index)
    plt.tight_layout()
    plt.ylabel(df_confusion.index.name)
    plt.xlabel(df_confusion.columns.name)

plot_confusion_matrix(df_confusion)

plt.show()
```

Hasilnya akan seperti berikut :



Gambar 3.5 Confusion Matrix

3.1.7 Voting Pada Random Forest

3.1.7.1 Pengertian Voting yaitu suara untuk setiap target yang diprediksi pada saat melakukan Random Forest. Pertimbangan target prediksi dengan voting tertinggi sebagai prediksi akhir dari algoritma random forest.

3.1.7.2 Contoh

- Untuk menggunakan Voting pada Random Forest dapat dilihat code berikut. Disini saya mengilustrasikan voting untuk berbagai macam algoritma terutama Random Forest.

```
import numpy as np
import matplotlib.pyplot as plt

from sklearn.linear_model import LogisticRegression
from sklearn.naive_bayes import GaussianNB
from sklearn.ensemble import RandomForestClassifier
from sklearn.ensemble import VotingClassifier

clf1 = LogisticRegression(solver='lbfgs', max_iter=1000, ran
clf2 = RandomForestClassifier(n_estimators=100, random_state
clf3 = GaussianNB()
X = np.array([[-1.0, -1.0], [-1.2, -1.4], [-3.4, -2.2], [1.1
y = np.array([1, 1, 2, 2])
```

```

ecclf = VotingClassifier(estimators=[('lr', clf1), ('rf', clf2)],
                        voting='soft',
                        weights=[1, 1, 5])

# predict class probabilities for all classifiers
probas = [c.fit(X, y).predict_proba(X) for c in (clf1, clf2)]

# get class probabilities for the first sample in the dataset
class1_1 = [pr[0, 0] for pr in probas]
class2_1 = [pr[0, 1] for pr in probas]

# plotting

N = 4 # number of groups
ind = np.arange(N) # group positions
width = 0.35 # bar width

fig, ax = plt.subplots()

# bars for classifier 1-3
p1 = ax.bar(ind, np.hstack(([class1_1[:-1], [0]])), width,
            color='green', edgecolor='k')
p2 = ax.bar(ind + width, np.hstack(([class2_1[:-1], [0]])),
            color='lightgreen', edgecolor='k')

# bars for VotingClassifier
p3 = ax.bar(ind, [0, 0, 0, class1_1[-1]], width,
            color='blue', edgecolor='k')
p4 = ax.bar(ind + width, [0, 0, 0, class2_1[-1]], width,
            color='steelblue', edgecolor='k')

# plot annotations
plt.axvline(2.8, color='k', linestyle='dashed')
ax.set_xticks(ind + width)
ax.set_xticklabels(['LogisticRegression\nweight 1',
                    'GaussianNB\nweight 1',
                    'RandomForestClassifier\nweight 5',
                    'VotingClassifier\n(average probabilities)'],
                  rotation=40,
                  ha='right')

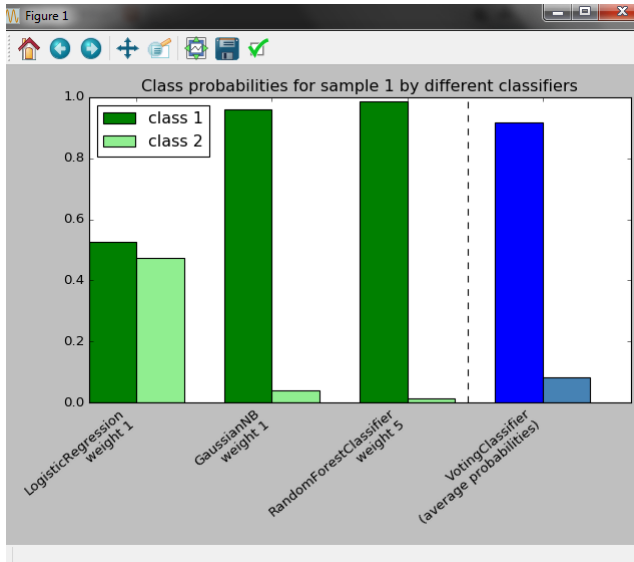
plt.ylim([0, 1])
plt.title('Class probabilities for sample 1 by different classifiers')
plt.legend([p1[0], p2[0]], ['class 1', 'class 2'], loc='upper right')

```



```
plt.tight_layout()
plt.show()
```

- Hasilnya sebagai berikut



Gambar 3.6 Voting Random Forest

3.2 Praktek Program

3.2.1 Aplikasi Sederhana Menggunakan Pandas

Disini saya akan membuat program sederhana menggunakan Pandas yaitu untuk memilih baris dari DataFrame yang diberikan berdasarkan value di salah satu kolom.

```
1 import pandas as pd
2 d = {'kol1': [1, 4, 3, 4, 5], 'kol2': [4, 5, 6, 7, 8], 'kol3': [7, 8, 9, 0, 1]}
3 df = pd.DataFrame(data=d)
4 print("Original DataFrame")
5 print(df)
6 print('Baris Untuk kolom2 dengan value 7')
7 print(df.loc[df['kol2'] == 7 ] )
```

Listing 3.1 Code Program Sederhana Pandas

Dari code diatas dapat dijelaskan perbarisnya sebagai berikut :

- Baris pertama, yaitu import pandas yang artinya kita akan mengimport librari Pandas dari python dengan inisiasi pd.

- Variabel `d` didefinisikan data data untuk kolom 1, kolom2, dan kolom3
- Variabel `df` akan mengubah data pada variabel `d` disejajarkan menjadi baris dan kolom dengan menggunakan `pd dataframe`.
- Baris selanjutnya yaitu akan mencetak atau menampilkan tulisan Original `dataFrame` pada jendela konsol.
- Print `df` artinya akan mencetak atau menampilkan `DataFrame` dari data yang telah dibuat tadi.
- Baris selanjutnya yaitu akan mencetak atau menampilkan tulisan Baris Untuk kolom2 dengan value 7 pada jendela konsol.
- Baris terakhir akan menampilkan data yang telah disortir berdasarkan perintah. Dimana hanya akan menampilkan data yang terdapat value 7 pada kolom 2.

Hasilnya sebagai berikut :

```
Original DataFrame
  kol1 kol2 kol3
0     1     4     7
1     4     5     8
2     3     6     9
3     4     7     0
4     5     8     1
Baris Untuk kolom2 dengan value 7
  kol1 kol2 kol3
3     4     7     0
```

Gambar 3.7 Aplikasi Sederhana Menggunakan Pandas

3.2.2 Aplikasi Sederhana Menggunakan Numpy

Program yang akan dibuat yaitu menentukan atau menemukan nilai yang sama dari dua array . dapat dilihat dalam listing 3.2.

```
1 import numpy as np
2 array1 = np.array([0, 10, 20, 40, 60])
3 print("Array1: ", array1)
4 array2 = np.array([10, 30, 40])
5 print("Array2: ", array2)
6 print("Data Yang Sama Dari Kedua Array Adalah:")
7 print(np.intersect1d(array1, array2))
```

Listing 3.2 Code Program Sederhana Numpy

Dari code diatas dapat dijelaskan perbarisnya sebagai berikut :

- Baris pertama, yaitu `import numpy` yang artinya kita akan mengimport library Numpy dari python dengan inisiasi `np`.
- Variabel `array1` berisikan `np array` yang dimana akan membuat sebuah Array berisikan value yang telah disebutkan.

- Akan mencetak tulisan "Array1" dan menampilkan data dari variabel array1.
- Variabel array2 berisikan np array yang dimana akan membuat sebuah Array berisikan value yang telah disebutkan.
- Akan mencetak tulisan "Array2" dan menampilkan data dari variabel array2.
- Baris selanjutnya akan mencetak dan menampilkan tulisan "Data Yang Sama Dari Kedua Array Adalah:" pada jendela konsol.
- Dan yang terakhir np intersect1d akan menampilkan irisan dari array1 dan array2

Hasilnya sebagai berikut :

```
( 'Array1: ', array([ 0, 10, 20, 40, 60]))
( 'Array2: ', array([10, 30, 40]))
Data Yang Sama Dari Kedua Array Adalah:
[10 40]
```

Gambar 3.8 Aplikasi Sederhana Menggunakan Numpy

3.2.3 Aplikasi Sederhana Menggunakan Matplotlib

Program yang akan dibuat yaitu membuat dua baris atau lebih dengan lebar dan warna yang berbeda. Code lengkap pada listing 3.3

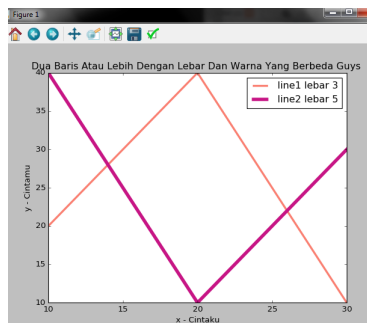
```
1 import matplotlib.pyplot as plt
2 # line 1 points
3 x1 = [10,20,30]
4 y1 = [20,40,10]
5 # line 2 points
6 x2 = [10,20,30]
7 y2 = [40,10,30]
8 # Set the x axis label of the current axis.
9 plt.xlabel('x - Cintaku')
10 # Set the y axis label of the current axis.
11 plt.ylabel('y - Cintamu')
12 # Set a title
13 plt.title('Dua Baris Atau Lebih Dengan Lebar Dan Warna Yang Berbeda Guys ')
14 # Display the figure.
15 plt.plot(x1,y1, color='salmon', linewidth = 3, label = 'line1 lebar 3')
16 plt.plot(x2,y2, color='mediumvioletred', linewidth = 5, label = 'line2 lebar 5')
17 # show a legend on the plot
18 plt.legend()
19 plt.show()
```

Listing 3.3 Code Program Sederhana Matplotlib

Dari code diatas dapat dijelaskan sebagai berikut :

- Pertama tama yaitu akan meng import librari Pyplot dari Matplotlib sebagai plt.
- Variabel x1 dan y1 akan berisikan value untuk titik atau point dari garis 1 nya.
- Begitu juga dengan variabel x2 dan y2 akan berisikan value untuk titik atau point dari garis 2 nya.
- Plt.xlabel akan mengatur label sumbu x dari axis saat ini dengan nama x Cintaku.
- Plt.ylabel akan mengatur label sumbu y dari axis saat ini dengan nama x Cintamu.
- Plt title akan mendefinisikan title atau judul dari grafik ini.
- plt plot akan menampilkan figurenya. Untuk line 1 diberi warna salmon dengan lebar garisnya 3 cm diberi label "line1 lebar 3". Dan untuk line 2 diberi warna mediumvioletred dengan lebar garisnya 5 cm diberi label "line2 lebar 5".
- plt legend untuk menampilkan legend
- Plt show digunakan untuk menampilkan grafik pada saat skrip dijalankan.

Hasilnya sebagai berikut :



Gambar 3.9 Aplikasi Sederhana Menggunakan Matplotlib

3.2.4 Menjalankan Program Klasifikasi Random Forest

Berikut adalah output dari percobaan Random Forest yang telah dilakukan

- Jika dilihat dari outputnya, code berikut berfungsi untuk membaca data yang berupa dataset 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.

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

Gambar 3.10 Program Random Forest Tasya

```
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
```

Gambar 3.11 Program Random Forest Tasya

- Output ini mengembalikan baris n teratas (5 secara default) dari dataframe imgatt.
- Output ini menampilkan jumlah baris dan kolom dari dataframe imgatt.

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

Gambar 3.12 Program Random Forest Tasya

- Dari outputnya dapat dilihat bahwa variabel imgatt2 menggunakan function pivot untuk mengubah kolom jadi baris, dan baris jadi kolom dari dataframe sebelumnya.

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

Gambar 3.13 Program Random Forest Tasya

- Sama seperti output sebelumnya, imgatt2 head itu berfungsi untuk mengembalikan nilai atau value teratas dari dataframe imgatt2.
- Output ini menampilkan jumlah baris dan kolom dari dataframe imgatt2
- Dan melakukan pivot yang mana imgid menjadi index yang artinya unik.
- Output diatas akan meload jawabannya yang berisi apakah burung itu termasuk dalam spesies yang mana. Dua kolomnya adalah imgid dan label.
- Output dari percobaan sebelumnya, menunjukkan 11788 baris dan 1 kolom. Dimana kolom itu adalah jenis spesies burungnya.

```
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]
```

Gambar 3.14 Program Random Forest Tasya

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

Gambar 3.15 Program Random Forest Tasya

```
In [7]: imglabels =
pd.read_csv("C:/080_2011/image_class_labels.txt",
...:        sep=" ", header=None, names=['imgid',
...:        'label'])
...:
...: imglabels = imglabels.set_index('imgid')
...:
...: # description from dataset README:
...: #
...: # The ground truth class labels (bird species labels) for
...: # each image are contained
...: # in the file image_class_labels.txt, with each line
...: # corresponding to one image:
...: #
...: # <image_id> <class_id>
...: #
...: # where <image_id> and <class_id> correspond to the IDs in
...: # images.txt and classes.txt,
...: # respectively.
```

Gambar 3.16 Program Random Forest Tasya

```
In [8]: imglabels.head()
Out[8]:
      label
imgid
1         1
2         1
3         1
4         1
5         1
```

Gambar 3.17 Program Random Forest Tasya

```
In [9]: imglabels.shape
Out[9]: (11788, 1)
```

Gambar 3.18 Program Random Forest Tasya

- Melakukan join antara imgatt2 dengan imglabels karena isinya sama. Sehingga kita akan mendapatkan data ciri dan data jawabannya atau labelnya sehingga bisa dikategorikan supervised learning.

```
In [10]: df = imgatt2.join(imglabels)
...: df = df.sample(frac=1)
```

Gambar 3.19 Program Random Forest Tasya

- Output diatas akan drop label yang didepan, dan menggunakan label yang paling belakang yang baru di join.

```
In [11]: df_att = df.iloc[:, :312]
...: df_label = df.iloc[:, 312:]
```

Gambar 3.20 Program Random Forest Tasya

- Output berikut mengecek isinya. Ini mengecek 5 data teratas dari df att

```
In [12]: df_att.head()
Out[12]:
```

	1	2	3	4	5	6	7	...	306	307	308	309
imgid	310	311	312					...				
10298	0	0	0	0	0	0	1	...	0	0	0	0
0	0	1										
673	0	0	0	0	0	0	0	...	0	0	0	0
0	0	1										
5793	0	0	0	0	0	0	0	...	0	0	0	0
0	0	1										
8683	0	0	0	0	0	0	1	...	0	0	0	0
1	0	0										
7830	0	0	0	0	0	0	1	...	0	0	0	0
0	0	1										

[5 rows x 312 columns]

Gambar 3.21 Program Random Forest Tasya

- Output berikut mengecek isinya. Ini mengecek 5 data teratas dari df label

```
In [13]: df_label.head()
Out[13]:
```

	label
imgid	
10298	175
673	13
5793	99
8683	148
7830	134

Gambar 3.22 Program Random Forest Tasya

- Output diatas membagi menjadi dua bagian, 8000 row pertama sebagai data training sisanya sebagai data testing.

```
In [14]: df_train_att = df_att[:8000]
...: df_train_label = df_label[:8000]
...: df_test_att = df_att[8000:]
...: df_test_label = df_label[8000:]
...:
...: df_train_label = df_train_label['label']
...: df_test_label = df_test_label['label']
```

Gambar 3.23 Program Random Forest Tasya

- Memanggil kelas RandomForestClassifier. max features diartikan sebagai berapa banyak kolom pada setiap tree disini kolom pada setiap tree adalah 50.

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

Gambar 3.24 Program Random Forest Tasya

- Output ini melakukan fit untuk membangun random forest yang sudah ditentukan dengan maksimum fitur sebanyak 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)
```

Gambar 3.25 Program Random Forest Tasya

- Menampilkan hasil prediksi dari random forest sebelumnya.

```
In [17]: print(clf.predict(df_train_att.head()))
[175  13  99 148 134]
```

Gambar 3.26 Program Random Forest Tasya

- Menampilkan besaran akurasi dari prediksi diatas atau Score perolehan dari klasifikasi.

```
In [18]: clf.score(df_test_att, df_test_label)
Out[18]: 0.44931362196409713
```

Gambar 3.27 Program Random Forest Tasya

3.2.5 Menjalankan Program Confusion Matrix

Berikut adalah output dari percobaan Confusion Matrix yang telah dilakukan

- Memetakan Random Forest ke dalam Confusion Matrix

```
In [19]: from sklearn.metrics import confusion_matrix
...: pred_labels = clf.predict(df_test_att)
...: cm = confusion_matrix(df_test_label, pred_labels)
```

Gambar 3.28 Program Confusion Matrix Tasya

- Melihat hasil dari gambar diatas


```
In [20]: cm
Out[20]:
array([[ 5,  2,  2, ...,  0,  0,  0],
       [ 0, 18,  0, ...,  0,  0,  0],
       [ 2,  1,  9, ...,  0,  0,  0],
       ...,
       [ 0,  0,  0, ...,  3,  0,  0],
       [ 0,  0,  0, ...,  0, 12,  0],
       [ 0,  0,  0, ...,  0,  0, 10]], dtype=int64)
```

Gambar 3.29 Program Confusion Matrix Tasya

```
In [21]: 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)
```

Gambar 3.30 Program Confusion Matrix Tasya

- Plotting Confusion Matrix dengan Matplotlib
- Set plot sumbu sesuai dengan nama datanya dan membaca file classes.txt

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

Gambar 3.31 Program Confusion Matrix Tasya

- Plot hasil perubahan label

```
In [23]: import numpy as np
...: np.set_printoptions(precision=2)
...: plt.figure(figsize=(60,60), dpi=300)
...: plot_confusion_matrix(cm, classes=birds, normalize=True)
...: plt.show()
Normalized confusion matrix
[[0.25 0.1  0.1  ... 0.  0.  0. ]
 [0.  0.72 0.  ... 0.  0.  0. ]
 [0.11 0.06 0.5  ... 0.  0.  0. ]
 ...
 [0.  0.  0.  ... 0.14 0.  0. ]
 [0.  0.  0.  ... 0.  0.52 0. ]
 [0.  0.  0.  ... 0.  0.  0.77]]
```

Gambar 3.32 Program Confusion Matrix Tasya

3.2.6 Menjalankan Program Klasifikasi SVM dan Decission Tree

Berikut adalah output dari percobaan Klasifikasi SVM dan Decission Tree yang telah dilakukan

- Mencoba klasifikasi dengan decission tree dengan dataset yang sama dan akan muncul akurasi prediksinya.

```
In [24]: from sklearn import tree
...: clf = tree.DecisionTreeClassifier()
...: clf.fit(df_train_att, df_train_label)
...: clf.score(df_test_att, df_test_label)
Out[24]: 0.2769271383315734
```

Gambar 3.33 Program Decission Tree Tasya

- Mencoba klasifikasi dengan SVM dengan dataset yang sama dan akan muncul akurasi prediksinya.

```
In [25]: from sklearn import svm
...: clfsvm = svm.SVC()
...: clfsvm.fit(df_train_att, df_train_label)
...: clfsvm.score(df_test_att, df_test_label)
E:\Anaconda2\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[25]: 0.2911826821541711
```

Gambar 3.34 Program SVM Tasya

3.2.7 Menjalankan Program Cross Validation

Berikut adalah output dari percobaan Cross Validation yang telah dilakukan

- Hasil Cross Validation untuk Random Forest
- Hasil Cross Validation untuk Decission Tree
- Hasil Cross Validation untuk SVM

```
In [26]: from sklearn.model_selection import cross_val_score
...: scores = cross_val_score(clf, df_train_att, df_train_label,
cv=5)
...: # show average score and +/- two standard deviations away
(covering 95% of scores)
...: print("Accuracy: %0.2f (+/- %0.2f)" % (scores.mean(),
scores.std() * 2))
Out[26]: 0.2911826821541711
```

Gambar 3.35 Program Cross Validation Tasya

```
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))
Accuracy: 0.45 (+/- 0.01)
Accuracy: 0.27 (+/- 0.02)
```

Gambar 3.36 Program Cross Validation Tasya

```
In [29]: scoressvm = cross_val_score(clfsvm, df_train_att,
df_train_label, cv=5)
...: print("Accuracy: %0.2f (+/- %0.2f)" % (scoressvm.mean(),
scoressvm.std() * 2))
Accuracy: 0.27 (+/- 0.02)
```

Gambar 3.37 Program Cross Validation Tasya

3.2.8 Menjalankan Program Komponen Informasi

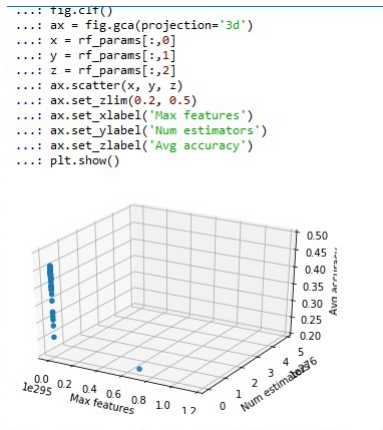
Berikut adalah output dari percobaan Komponen Informasi yang telah dilakukan

- Dari output ini dapat mengetahui berapa banyak tree yang dibuat, berapa banyak atribut yang dipakai dan informasi lainnya

```
In [30]: max_features_opts = range(5, 50, 5)
...: n_estimators_opts = range(10, 200, 20)
...: rf_params =
np.empty((len(max_features_opts)*len(n_estimators_opts),4), float)
...: i = 0
...: for max_features in max_features_opts:
...:     for n_estimators in n_estimators_opts:
...:         clf =
RandomForestClassifier(max_features=max_features,
n_estimators=n_estimators)
...:         scores = cross_val_score(clf, df_train_att,
df_train_label, cv=5)
...:         rf_params[i,0] = max_features
...:         rf_params[i,1] = n_estimators
...:         rf_params[i,2] = scores.mean()
...:         rf_params[i,3] = scores.std() * 2
...:         i += 1
...:         print("Max features: %d, num estimators: %d,
accuracy: %0.2f (+/- %0.2f)" %
(max_features,
n_estimators, scores.mean(), scores.std() * 2))
Max features: 5, num estimators: 10, accuracy: 0.26 (+/- 0.01)
Max features: 5, num estimators: 30, accuracy: 0.36 (+/- 0.02)
Max features: 5, num estimators: 50, accuracy: 0.39 (+/- 0.00)
Max features: 5, num estimators: 70, accuracy: 0.41 (+/- 0.01)
Max features: 5, num estimators: 90, accuracy: 0.42 (+/- 0.01)
Max features: 5, num estimators: 110, accuracy: 0.43 (+/- 0.02)
Max features: 5, num estimators: 130, accuracy: 0.44 (+/- 0.01)
Max features: 5, num estimators: 150, accuracy: 0.45 (+/- 0.01)
```

Gambar 3.38 Program Komponen Informasi Tasya

- Output berikut merupakan hasil dari plotting komponen informasi agar dapat dibaca



Gambar 3.39 Program Komponen Informasi Tasya

3.3 Penanganan Error

3.3.1 Error Index

1. Berikut ini merupakan error yang didapatkan saat menjalankan program diatas

```

In [34]: from sklearn.ensemble import RandomForestClassifier
...:     clf =
RandomForestClassifier(max_features=50, random_state=0,
n_estimators=100)
File "<ipython-input-34-4269507e9565>", line 2
    clf = RandomForestClassifier(max_features=50, random_state=0,
n_estimators=100)
    ^
IndentationError: unexpected indent

```

Gambar 3.40 Error Index

2. Pada gambar diatas kode erornya adalah IndentationError unexpected indent. Error ini terjadi karena adanya inkosisten pemberian indent di kode program.
3. Solusi yang bisa dilakukan untuk mengatasi error tersebut adalah sebagai berikut :

- Buka file code program dan lihat pada bagian erornya

```

from sklearn.ensemble import RandomForestClassifier
clf = RandomForestClassifier(max_features=50, random_state=0, n_estimators=100)

```

Gambar 3.41 File Codingan

- Dapat dilihat pada baris kedua terdapat spasi dibagian depan, hilangkan atau hapus spasi tersebut seperti berikut

```
from sklearn.ensemble import RandomForestClassifier
clf = RandomForestClassifier(max_features=50, random_state=0, n_estimators=100)
```

Gambar 3.42 Menghapus Spasi

- Save, kemudian ketika dijalankan eror akan teratasi

```
In [35]: from sklearn.ensemble import RandomForestClassifier
...: clf = RandomForestClassifier(max_features=50,
random_state=0, n_estimators=100)
```

Gambar 3.43 Eror Teratasi

BAB 4

EXPERIMENT AND RESULT

4.1 Teori

4.1.1 Klasifikasi Teks

4.1.1.1 Pengertian Klasifikasi Teks Merupakan salah satu tugas terpenting dalam Pemrosesan Bahasa Alami (Natural Language Processing). Ini adalah proses mengklasifikasikan string teks atau dokumen ke dalam kategori yang berbeda, tergantung pada konten string. Klasifikasi teks memiliki berbagai aplikasi, seperti mendeteksi sentimen pengguna dari tweet, mengklasifikasikan email sebagai spam atau ham, mengklasifikasikan posting blog ke dalam kategori yang berbeda, penandaan otomatis permintaan pelanggan, dan sebagainya. Berikut adalah contoh dari Klasifikasi Teks. Contohnya, misal kita ingin mencari kata dog, table, on, the . kemudian jika kata yang dimaksud sesuai maka akan menampilkan bilangan biner 1 dan jika salah 0. Seperti dibawah ini :

the dog is on the table

0	0	1	1	0	1	1	1
are	cat	dog	is	now	on	table	the

Gambar 4.1 Klasifikasi Teks Tasya

4.1.2 Klasifikasi Bunga

Jelaskan mengapa klasifikasi bunga tidak bisa menggunakan machine learning, sertakan ilustrasi sendiri. Dikarenakan tidak semua bunga memiliki ciri - ciri yang sama. Atau dalam kata lain terdapat data noise dalam klasifikasi bunga sehingga tidak bisa menggunakan machine learning. Contohnya Anggrek memiliki warna ungu, dengan jumlah kelopak 5. Kemudian ada bunga ungu dengan jumlah kelopak yang sama namun ternyata bukan anggrek dan kategorinya banyak sekali. Bahkan ada bunga yang tidak jelas apakah warnanya sesuai atau tidak, sehingga bisa menyebabkan data noise.



Gambar 4.2 Klasifikasi Bunga Berwana Ungu Tasya

4.1.3 Pembelajaran Mesin Pada Teks Kata - Kata di Youtube

Menggunakan teknik bag-of-words pada klasifikasi berbasis text dan kata untuk mengklasifikasikan komentar yang ada di internet sebagai spam atau bukan. Misalkan pada kolom komentar dapat di cek seberapa sering suatu kata muncul dalam kalimat. Setiap kata dapat dijadikan baris dan kolomnya ini 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. Contohnya seperti berikut :

	COMMENT_ID	DATE	CONTENT	CLASS
346	z13th1q4yzihf1bl023qzpjuejterjdj	2014-11-14T13:27:52	How can this have 2 billion views when there's...	0
346	z13fn1wfpb5e51xe04chdxakpazgchyaxzo0k	2014-11-14T13:28:08	I don't now why I'm watching this in 2014	0
347	z130zd5b3titudkoe04ccbeohojkuzppvbg	2015-05-23T13:04:32	subscribe to me for call of duty vids and give...	1
348	z12he50arvkv15u04ctawgkzbfjgccc4	2015-06-05T14:14:48	hi guys please my android photo editor downloa...	1
349	z13vhvu54u3ewpp5h04ccb4zuoardmjljk0k	2015-06-05T18:05:16	The first billion viewed this because they tho...	0

Gambar 4.3 Klasifikasi Comment Spam Di Youtube Tasya

4.1.4 Arti Score 44% Pada Random Forest, 27% Pada Decission Tree Dan 29% Dari SVM

Itu merupakan presentase keakurasian prediksi yang dilakukan pada saat testing menggunakan label pada dataset yang digunakan. Score merupakan mendefinisikan aturan evaluasi model. Maka pada saat dijalankan akan muncul persentase tersebut yang menunjukkan keakurasian atau keberhasilan dari prediksi yang dilakukan. Jika menggunakan Random Forest maka hasilnya 40% , jika menggunakan Decission Tree hasil prediksinya yaitu 27% dan pada SVM 29% .

4.1.5 Bag of Words

4.1.5.1 Pengertian Bag of Words Merupakan representasi teks yang menggambarkan kemunculan kata-kata dalam dokumen. ePngelompokan kata kata kedalam perhitungan, berapakai sebuah kata muncul dalam satu kalimat. Disebut "tas" kata-kata, karena informasi tentang susunan atau struktur kata dalam dokumen dibuang. Model ini hanya berkaitan dengan apakah kata-kata yang diketahui muncul dalam dokumen, bukan di mana dalam dokumen.

Contohnya disini akan melihat kemunculan kata dari kalimat :

1. I Love Dogs
2. I hate dogs and knitting
3. Knitting is my hobby and passion.

	I	love	dogs	hate	and	knitting	is	my	hobby	passion
Doc 1	1	1	1							
Doc 2	1		1	1	1	1				
Doc 3					1	1	1	2	1	1

Gambar 4.4 Bag of Words Tasya

4.1.6 TF-IDF

TF-IDF memberi kita frekuensi kata dalam setiap dokumen dalam korpus 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. Dalam ilustrasi disini saya akan mengganti contoh Bag of Words menjadi bentuk TF-IDF.

	I	love	dogs	hate	and	knitting	is	my	hobby	passion
Doc 1	0.18	0.48	0.18							
Doc 2	0.18		0.18	0.48	0.18	0.18				
Doc 3					0.18	0.18	0.48	0.95	0.48	0.48

Gambar 4.5 Contoh TF-IDF Tasya

4.2 PRAKTIKUM

4.2.1 Aplikasi Sederhana Menggunakan Pandas

Disini saya akan menggunakan Dataset dari <https://www.kaggle.com/spscientist/students-performance-in-exams> dan akan mengambil Data Dummy sebanyak 500 records dan membuat Dataframe baru.

```
import pandas as pd
mhs = pd.read_csv('StudentsPerformanceExam.csv', sep=';')
df = pd.DataFrame(mhs, columns = ['gender', 'race/ethnicity', '

dummy = pd.get_dummies (df['test preparation course'])
dummy.head()

df = df.join(dummy)
```

Maksud dari kodingan diatas yaitu :

1. Baris pertama impor librari pandas dengan inisiasi pd
2. Definisikan variabel mhs untuk membaca file csv dengan pandas
3. variabel df akan menggunakan function pd.DataFrame untuk membuat dataframe di pandas dari file CSV yang tadi.
4. Mendefinisikan variabel dummy untuk mengubah data categorical menjadi integer. dimana merupakan data sebelum di Dummy.

Index	gender	race/ethnicity	ntal level of educi	lunch	t preparation cou	math score	reading score	writing score
0	female	group C	some college	standard	completed	69	90	88
1	female	group B	master's degree	standard	none	90	95	93
2	male	group A	associate's degree	free/reduced	none	47	57	44
3	male	group C	some college	standard	none	76	78	75
4	female	group B	associate's degree	standard	none	71	83	78
5	female	group B	some college	standard	completed	88	95	92
6	male	group B	some college	free/reduced	none	40	43	39
7	male	group D	high school	free/reduced	completed	64	64	67
8	female	group B	high school	free/reduced	none	38	60	50
9	male	group C	associate's degree	standard	none	58	54	52
10	male	group D	associate's degree	standard	none	40	52	43
11	female	group B	high school	standard	none	65	81	73
12	male	group A	some college	standard	completed	78	72	70
13	female	group A	master's degree	standard	none	50	53	58
14	female	group C	some high school	standard	none	69	75	78
15	male	group C	high school	standard	none	88	89	86
16	female	group B	some high school	free/reduced	none	18	32	28
17	male	group C	master's degree	free/reduced	completed	46	42	46
18	female	group C	associate's degree	free/reduced	none	54	58	61
19	male	group D	high school	standard	none	66	69	63
20	female	group B	some college	free/reduced	completed	65	75	70
21	male	group D	some college	standard	none	44	54	53
			some high					

Gambar 4.6 Dataset Original Tasya

5. Atribut atau kolom yang ingin di Dummy yaitu test preparation course. Dalam test preparation course terdapat dua value yaitu Completed dan none. Yang jika di dummy maka valuenya akan berubah menjadi 1 dan 0 seperti berikut :
6. kemudian df akan melakukan join dengan dataframe dummy.

4.2.2 Memecah DataFrame Menjadi 2 Dataframe

Dari dataframe tersebut dipecah menjadi dua dataframe yaitu 450 row pertama dan 50 row sisanya

```
mhs_train= mhs[:450]
mhs_test= mhs[451:]
```

1. mhs train akan mendefinisikan dataframe untuk train dengan 450 data pertama
2. mhs test mendefinisikan dataframe untuk test untuk data setelah 451. Hasilnya seperti berikut :

dummy - DataFrame			
Index	completed	none	
0	1	0	
1	0	1	
2	0	1	
3	0	1	
4	0	1	
5	1	0	
6	0	1	
7	1	0	
8	0	1	
9	0	1	
10	0	1	
11	0	1	
12	1	0	
13	0	1	

Gambar 4.7 Dataset Dummy Tasya

mhs_test	DataFrame	(50, 8)
mhs_tra...	DataFrame	(450, 8)

Gambar 4.8 Split DataFrame Tasya

4.2.3 Vektorisasi Dan Klasifikasi Dari Data Youtube Eminem Dengan Decision Tree

1. Ini hasil dari impor dataset

```
In [43]: import pandas as pd
...: d = pd.read_csv("Youtube04-Eminem.csv")

In [44]: from sklearn.feature_extraction.text import CountVectorizer
...: vectorizer = CountVectorizer()
```

Gambar 4.9 Dataset Youtube Eminem Tasya

2. Ini hasil Setelah di Klasifikasikan dengan Decision Tree
3. Dalam in 52 impor Tree dari Sklearn. Dan mendefinisikan variabel clf untuk memanggil Decision Tree Classifier dan melakukan fit atau pengujian
4. Dalam In 53 menggunakan prediksi untuk clf dengan function predict untuk memprediksi test. Dan hasilnya muncul dalam bentuk array.
5. clf score memunculkan akurasi prediksi yang dilakukan terhadap clf.

```
In [52]: from sklearn import tree
...: clf = tree.DecisionTreeClassifier(criterion="entropy", max_depth=5)
...: clf = clf.fit(d_train_att, d_train_label)

In [53]: clf.predict(d_test_att)
Out[53]:
array([0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 1, 1, 0, 0, 1, 0, 1, 0, 1, 1, 0, 0, 0, 0,
       0, 1, 0, 0, 1, 0, 1, 1, 1, 1, 1, 1, 0, 1, 0, 1, 0, 1, 1, 1, 0, 0, 0, 1,
       1, 1, 0, 1, 0, 1, 1, 1, 0, 0, 0, 1, 0, 0, 1, 0, 1, 0, 0, 0, 0, 1, 0,
       0, 0, 1, 0, 0, 0, 1, 1, 0, 0, 0, 0, 1, 0, 1, 0, 1, 0, 1, 0, 0, 0, 1,
       0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 1, 1, 1, 0, 0, 0, 0, 1, 1, 0, 1, 1,
       0, 1, 0, 0, 0, 1, 0, 0, 0, 1, 1, 0, 1, 0, 0, 0, 1, 0, 1, 1, 1, 0, 1,
       0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1], dtype=int64)

In [54]: clf.score(d_test_att, d_test_label)
Out[54]: 0.9324324324324325
```

Gambar 4.10 Dataset Youtube Eminem Tasya

4.2.4 Vektorisasi Dan Klasifikasi Dari Data Youtube Eminem Dengan SVM

```
In [42]: from sklearn import svm
...: clfsvm = svm.SVC()
...: clfsvm.fit(d_train_att, d_train_label)
...: clfsvm.score(d_test_att, d_test_label)
E:\Anaconda2\lib\site-packages\sklearn\svm\base.py:
will change from 'auto' to 'scale' in version 0.22
gamma explicitly to 'auto' or 'scale' to avoid this
"avoid this warning.", FutureWarning)
Out[42]: 0.6959459459459459
```

Gambar 4.11 Dataset Youtube Eminem SVM Tasya

Dari Gambar diatas dapat dijelaskan bahwa :

1. Impor SVM dari sklearn
2. Melakukan fit dari d train att dan d train label atau disebut dengan pengujian
3. Mendefinisikan variabel clf untuk melakukan prediksi dataset Youtube Eminem dengan SVM. Dan akan muncul hasil prediksinya

4.2.5 Vektorisasi Dan Klasifikasi Dari Data Youtube Eminem Dengan Decision Tree 2

```
In [59]: from sklearn import tree
...: clf = tree.DecisionTreeClassifier(criterion="entropy", max_depth=5)
...: clf = clf.fit(d_train_att, d_train_label)
...: clf.predict(d_test_att)
...: clf.score(d_test_att, d_test_label)
Out[59]: 0.9324324324324325
```

Gambar 4.12 Dataset Youtube Eminem Tasya

Maksud dari codingan diatas yaitu, mengkasifikasikan Dataset Youtube Eminem dengan Decision Tree dengan melakukan prediksi menggunakan function test pada d test att, dan memberikan akurasi prediksi menggunakan prediksi score.

4.2.6 Plotting Confusion Matrix

Berikut adalah skript dari plotting confusion matrix dari contoh yang ada pada bagian teori

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

    print(cm)

    plt.imshow(cm, interpolation='nearest', cmap=cmap)
    plt.title(title)
    #plt.colorbar()
    tick_marks = np.arange(len(classes))
    plt.xticks(tick_marks, classes, rotation=90)
    plt.yticks(tick_marks, classes)

    fmt = '.2f' if normalize else 'd'
    thresh = cm.max() / 2.
    #for i, j in itertools.product(range(cm.shape[0]), range(cm.shape[1])):
    #    plt.text(j, i, format(cm[i, j], fmt),
    #             horizontalalignment="center",
    #             color="white" if cm[i, j] > thresh else "black")

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


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

```
plt.show()
```

Hasilnya adalah sebagai berikut : Dari gambar dapat dijelaskan bahwa data array

```
In [18]: import numpy as np
...: np.set_printoptions(precision=2)
...: plt.figure(figsize=(60,60), dpi=300)
...: plot_confusion_matrix(cm, classes=clf, normalize=True)
...: plt.savefig()
Normalized confusion matrix
[[0.97 0.03]
 [0.06 0.94]]
```

Gambar 4.13 Confusion Matrix Tasya

merupakan data asli dan data prediksi yang dilakukan dengan Random Forest. Dengan melakukan normalisasi data confusion matrix.

4.2.7 Menjalankan Program Cross Validation

```
In [23]: from sklearn.model_selection import cross_val_score
...: scores = cross_val_score(clf,d_train_att,d_train_label,cv=5)
...:
...: skorrata2=scores.mean()
...: skoresd=scores.std()
```

Gambar 4.14 Cross Validation Tasya

Gambar diatas akan dijelaskan seperti berikut :

1. Dari sklearn mengimpor Cross Validation
2. Variabel scores akan melakukan cross validation pada variabel clf, d train att , dan d train label
3. Variabel skorrata2 akan menghitung nilai rata rata dari variabel scores tadi menggunakan function mean
4. skoresd Menghitung standar deviasi dari data yang diberikan. Hasilnya seperti berikut :

```
In [52]: from sklearn import tree
...: clf = tree.DecisionTreeClassifier(criterion="entropy", max_depth=5)
...: clf = clf.fit(d_train_att, d_train_label)

In [53]: clf.predict(d_test_att)
Out[53]:
array([0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 1, 1, 0, 1, 0, 1, 0, 1, 1, 1, 0, 0, 0,
       0, 1, 0, 0, 1, 0, 1, 1, 1, 1, 1, 1, 0, 1, 0, 1, 1, 1, 0, 0, 0,
       1, 1, 0, 1, 0, 1, 1, 1, 0, 0, 0, 1, 0, 1, 0, 0, 1, 1, 0, 0, 0, 1,
       0, 0, 1, 0, 0, 0, 1, 1, 0, 0, 0, 1, 0, 1, 0, 1, 0, 0, 1, 0,
       0, 0, 1, 1, 0, 0, 0, 0, 0, 1, 1, 1, 1, 0, 0, 0, 0, 1, 1, 0, 1, 1,
       0, 1, 0, 0, 1, 0, 0, 1, 1, 0, 1, 0, 1, 1, 1, 0, 1, 0, 1, 1,
       0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 1, 1]) dtype=int64)

In [54]: clf.score(d_test_att,d_test_label)
Out[54]: 0.9324324324324325
```

Gambar 4.15 Hasil Cross Validation Tasya

4.2.8 Program Pengamatan Komponen Informasi

```
In [19]: max_features_opts = range(5, 50, 5)
...: n_estimators_opts = range(10, 150, 20)
...: rf_params =
np.empty((len(max_features_opts)*len(n_estimators_opts),4), float)
...: i = 0
...: for max_features in max_features_opts:
...:     for n_estimators in n_estimators_opts:
...:         clf =
RandomForestClassifier(max_features=max_features,
n_estimators=n_estimators)
...:         scores = cross_val_score(clf, d_train_att,
d_train_label, cv=5)
...:         rf_params[i,0] = max_features
...:         rf_params[i,1] = n_estimators
...:         rf_params[i,2] = scores.mean()
...:         rf_params[i,3] = scores.std() * 2
...:         i += 1
```

Gambar 4.16 Program Komponen Informas Tasyai

Dari gambar diatas dapat dijelaskan bahwa :

1. Max featuresnya dari range 5 sampai 50
2. n estimators dengan range 10 sampai 150
3. Variabel rf params berisikan function np empty dimana akan membuat array baru berisikan tipe yang didefinisikan dengan random value
4. Mendefinisika i dimulai dari angka 0 dimana max features dan n estimators menggunakan klasifikasi randomforestclassifier menggunakan data prediksi
5. Mendefinisikan rfparams untuk max features , n estimators, nilai rata dan std

4.3 Penanganan Error

4.3.1 Error Index

1. Berikut ini merupakan eror yang didapatkan saat menjalankan program diatas

```
KeyError: 'student'
```

Gambar 4.17 Error Key Tasya

2. Pada gambar diatas kode erornya adalah KeyError. Error ini terjadi karena key-word yang dimasukan tidak ada.
3. Solusi yang bisa dilakukan untuk mengatasi eror tersebut adalah sebagai berikut :

```
###
dummy = pd.get_dummies (df['student'])
dummy.head()
```

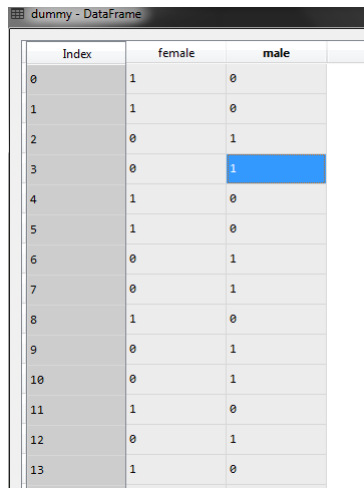
Gambar 4.18 Error Key Tasya

- Pada gambar diatas Dataset StudentPerformanceExam tidak terdapat atribut student, maka dari itu kita harus merubahnya dengan atribut yang terdapat di dataset tersebut. Mari kita gunakan atribut gender. Ubah skrip menjadi seperti berikut

```
dummy = pd.get_dummies (df['gender'])
dummy.head()
```

Gambar 4.19 Error Key Tasya

-
- Maka ketika di run akan muncul data dummy nya seperti berikut



Index	female	male
0	1	0
1	1	0
2	0	1
3	0	1
4	1	0
5	1	0
6	0	1
7	0	1
8	1	0
9	0	1
10	0	1
11	1	0
12	0	1
13	1	0

Gambar 4.20 Error Key Tasya

VEKTORISASI KATA DAN DOKUMEN

5.1 Teori

5.1.1 Vektorisasi

Karena ketika menggunakan algoritma machine learning tidak bisa secara langsung menggunakan teks melainkan teks tersebut harus diubah menjadi angka. Kita membutuhkan cara untuk merepresentasikan data teks untuk algoritma pembelajaran mesin, vektorisasi membantu mengubah teks biasa kedalam bentuk vektor yang dapat dimengerti oleh komputer atau machine learning. Kita mungkin ingin melakukan klasifikasi dokumen, sehingga setiap dokumen adalah "input" dan label kelas adalah "output" untuk algoritma prediksinya. Algoritma mengambil vektor angka sebagai input, oleh karena itu kita perlu mengkonversi dokumen menjadi vektor angka dengan panjang tetap atau sama.

Untuk ilustrasinya misal, saya memiliki kamus berisikan kata-kata MonkeyLearn, is, not, great, dan saya ingin membuat vektor teks "MonkeyLearn is great", saya akan memiliki vektor berikut: (1, 1, 0, 0, 1,).

5.1.2 Vektor Dataset Google

Karena di dalam satu dataset berisikan setidaknya 3 Milyar kata dan kalimat. Yang dimana dimensi berisikan kata - kata unik dari data tersebut. Maka dari itu dimensi pada dataset Google bisa mencapai 300.

Untuk ilustrasinya, misalkan kita memiliki sebuah buku dengan tebal 1000 , dimana bukunya dibagi menjadi dua Chapter. Kemudian kita akan menggabungkan kata dari setiap Chapter tersebut. Maka akan didapatkan irisan yang akan berjumlah lebih dari 200. dikarenakan banyak kata yang berbeda beda.

5.1.3 Konsep Vektorisasi Untuk Kata

Konsepnya yaitu kata atau teks akan dihapuskan noisy datanya atau dihapus data yang tidak terpakai, seperti tag html jika ada, titik, koma, dll. Kemudian tokenization artinya kita akan mengelompokkan kalimat menjadi token atau membagi kata kata menjadi potongan kecil. Baru setelah itu dilakukan normalisasi untuk mengubah datanya menjadi angka.

Ilustrasinya, misalnya ada beberapa kalimat seperti berikut :

- There used to be Iron Age.

Kemudian didapatkan token seperti berikut “There”, ”was”, ”to”, ”be”, ”used”, ”Stone”, ”Bron

Maka ketika di cek pada kalimat diatas hasilnya seperti berikut

- There used to be iron age = [1,0,1,1,1,0,0,1,0,0,1,0,0,0,0]

5.1.4 Konsep Vektorisasi Untuk Dokumen

Hampir mirip dengan konsep kata, namun untuk di dokumen biasanya konsepnya digunakan untuk mencari kesamaan atau memprediksi seberapa sering muncul kata dalam 2 kalimat atau 2 paragraf.

Ilustrasinya, misalkan dalam sebuah artikel kita ingin mencari seberapa banyak kata ”dimana” muncul. Maka dengan Doc2Vec dapat diprediksi hasilnya.

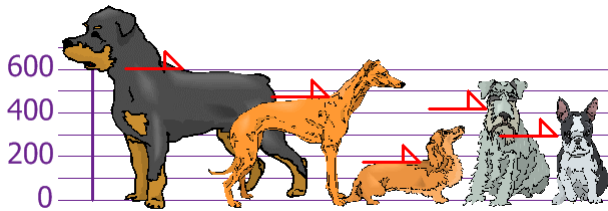
5.1.5 Mean Dan Standar Deviasi

5.1.5.1 Pengertian Mean adalah nilai rata-rata dari beberapa buah data. Nilai mean dapat ditentukan dengan membagi jumlah data dengan banyaknya data.

Deviasi standar adalah ukuran ringkasan perbedaan setiap pengamatan dari rata-rata. Deviasi standar mengukur penyebaran data tentang nilai rata-rata. Ini berguna dalam membandingkan set data yang mungkin memiliki mean yang sama tetapi rentang yang berbeda.

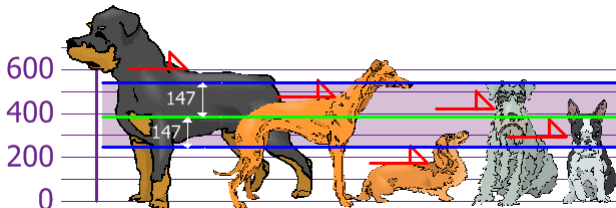
5.1.5.2 Contoh

1. Misalkan kita sudah menghitung tinggi anjing.



Gambar 5.1 Contoh Mean dan Standar Deviasi

2. Tingginya dilihat dari bahu : 600mm, 470mm, 170mm, 430mm and 300mm.
3. Kita hitung mean atau rata ratanya, dengan menjumlahkan seluruh data dan membaginya dengan jumlah n nya hasilnya yaitu 394.
4. Kemudian kita ingin melihat berapa perbedaan tinggi dari anjing - anjing tersebut menggunakan variance.
5. Baru Gunakan standar deviasi didapatkan hasil 147 mm . Dengan Deviasi Standar kita bisa tahu mana anjing dengan tinggi normal dan anjing yang kekurangan tinggi.



Gambar 5.2 Contoh Mean dan Standar Deviasi

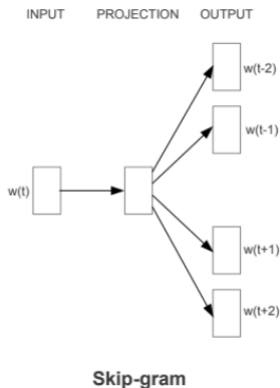
5.1.6 Skip-gram

Arsitektur model Skip-gram biasanya mencoba untuk memprediksi kata konteks sumber (kata-kata sekitarnya) diberi kata target (kata tengah). Contohnya seperti berikut

5.2 PRAKTEK PROGRAM

5.2.1 Mencoba Dataset

5.2.1.1 Vektor



Gambar 5.3 Contoh Skipgram

- Pada gambar diatas dapat dilihat bahwa vektor memiliki array sebanyak 300 dimensi. Untuk identitas sektor satu adalah 0.10

```
In [14]: gmodel ['love']
Out[14]:
array([ 0.10302734, -0.15234375,  0.02587891,  0.16503906,
        -0.16503906,
         0.06689453,  0.29296875, -0.26367188, -0.140625 ,
         0.20117100])
```

Gambar 5.4 Vektor Love Tasya

- Pada gambar diatas untuk vektor faith dapat dilihat memiliki nilai 0.26 , untuk similaritasnya cukup mendekati vektor love dimana faith dapat dikategorikan dalam satu kategori dengan love.

```
In [15]: gmodel ['faith']
Out[15]:
array([ 0.26367188, -0.04150391,  0.1953125 ,  0.13476562,
        -0.14648438,
         0.06689453,  0.29296875, -0.26367188, -0.140625 ,
         0.20117100])
```

Gambar 5.5 Vektor Faith Tasya

- Vektor fall hanya memiliki nilai minus yaitu -0.04 , dimana mesin memahami bahwa fall tidak terdapat dalam satu kategori yang sama dengan love dan faith

```
In [16]: gmodel ['fall']
Out[16]:
array([-0.04272461,  0.10742188, -0.09277344,  0.16894531, -0.1328125 ,
         0.06689453,  0.29296875, -0.26367188, -0.140625 ,
         0.20117100])
```

Gambar 5.6 Vektor Fall Tasya

- Vektor sick memiliki nilai identitas 1.82 dimana tidak mendekati love, faith maupun fall.

```
In [17]: gmodel['sick']
Out[17]:
array([ 1.82617188e-01,  1.49414062e-01, -4.05273438e-02,
 1.64062500e-01,
-2.59765625e-01,  3.22265625e-01,  1.73828125e-01,
```

Gambar 5.7 Vektor Sick Tasya

- Vektor clear memiliki nilai identitas -2,44 dan tidak mendekati nilai dari vektor fall sehingga tidak dapat dijadikan dalam satu kategori

```
In [18]: gmodel['clear']
Out[18]:
array([-2.44140625e-04, -1.02058701e-01, -1.49414062e-01,
-4.24094688e-02,
```

Gambar 5.8 Vektor Clear Tasya

- Untuk vektor shine -0.12 tidak mendekati vektor manapun.

```
In [19]: gmodel['shine']
Out[19]:
array([-0.12082344,  0.25976562, -0.15917969, -0.27734375,
 0.38273438,
 0.09960938,  0.39257812, -0.22949219, -0.18359575,  0.3671875
```

Gambar 5.9 Vektor Shine Tasya

- Vektor bag memiliki i=nilai identitas -0.03 yang mendekati dengan vektor fall. SEhingga mesin memahami bahwa mungkin saja kedua vektor tersebut berada dalam satu kategori.

```
In [20]: gmodel['bag']
Out[20]:
array([-0.0315625,  0.15234375, -0.12402344,  0.13378906,
-0.11328125,
-0.0133667, -0.16113281,  0.14640438, -0.06835938,  0.140025
-0.06085859, -0.3046875,  0.20996094, -0.04345703, -0.2189375
```

Gambar 5.10 Vektor Bag Tasya

- Vektor car nilainya 0.13 mendekati vektor love dan faith sehingga mungkin dapat dikategorikan dalam satu kategori.

```
In [21]: gmodel['car']
Out[21]:
array([ 0.13085938,  0.08842205,  0.03344727, -0.05883789,
 0.04083081,
-0.14257812,  0.04931641, -0.16894531,  0.20898438,
 0.11962891,
```

Gambar 5.11 Vektor Car Tasya

- Vektor wash memiliki nilai 9.46 jauh dari vektor vektor lainnya.

```
In [22]: gmodel['wash']
Out[22]:
array([ 9.46044932e-03,  1.41601562e-01, -5.46875000e-02,
 1.34795625e-01,
-2.38281250e-01,  3.24218750e-01, -0.44726562e-02,
-1.29883812e-01,
```

Gambar 5.12 Vektor Wash Tasya

- Vektor motor memiliki nilai identitas 5.73 yang bisa mendekati vektor wash. Dapat dikatakan bahwa motor dapat dicuci jika diarti dalam satu kategori yang sama.

```
In [23]: model['motor']
Out[23]:
array([ 5.73738469e-02,  1.58998625e-01, -4.61425781e-02,
        -1.33023908e-01,
        -2.59765625e-01, -1.77734375e-01,  3.68652344e-02,
```

Gambar 5.13 Vektor Motor Tasya

5.2.1.2 Similariti

1. Lihat gambar berikut yang merupakan hasil prediksi similariti

```
In [49]: model.similarity('love', 'faith')
Out[49]: 0.5705479345872814
In [20]: model.similarity('love', 'fall')
Out[20]: 0.114258711174504
In [21]: model.similarity('love', 'sick')
Out[21]: 0.2665636147352396
In [22]: model.similarity('love', 'clear')
Out[22]: 0.8058444763112513
In [23]: model.similarity('love', 'shine')
Out[23]: 0.2006529663580627
In [24]: model.similarity('love', 'bag')
Out[24]: 0.8751609888992802
In [25]: model.similarity('love', 'car')
Out[25]: 0.804178989147326
In [26]: model.similarity('love', 'wash')
Out[26]: 0.1138335683306497
In [27]: model.similarity('love', 'motor')
Out[27]: 0.88877487864184623
In [28]: model.similarity('love', 'cycle')
Out[28]: 0.85664588917538113
```

Gambar 5.14 Similariti Tasya

Dapat disimpulkan bahwa

- Untuk Love dan faith hasilnya adalah 37
- Untuk Love dan fall hasilnya adalah 11
- Untuk Love dan sick hasilnya adalah 26
- Untuk Love dan clear hasilnya adalah 6
- Untuk Love dan shine hasilnya adalah 20
- Artinya love dan faith memang dalam kategoruiyang sama misalnya dalam kategori percintaan. MESin sudah mengetahui bahwa keduanya dapat dikategorikan sebagai percintaan.

5.2.2 Extract Words dan PermuteSentences

5.2.2.1 Extract Words ExtractWords merupakan function untuk menambahkan, menghilangkan atau menghapuskan, hal hal yang tidak penting atau tidakperlu di dalam teks. Dalam contoh dibawah ini. menggunakan function extract words untuk menghapus komen dengan python style , mencari data yang diinginkan, dan memberikan spasi pada teks.

```
In [33]: import re
...: def extract_words(cantik):
...:     cantik = cantik.lower()
...:     cantik = re.sub(r'&.*$', '', cantik) #hapus python style
...:
...:     cantik = re.findall(r"'\w+'", cantik)
...:     cantik = re.split(r' ', ', ', cantik)
...:     return cantik.split()
```

Gambar 5.15 Extract Words Tasya

5.2.2.2 *PermuteSentences* *PermuteSentences* merupakan class yang digunakan untuk melakukan pengocokan secara acak pada data yang ada. Digunakan cara ini agar tidak terjadi kelebihan memori pada saat dijalankan. Contoh dibawah yaitu fungsi akan memanggil `length`. Yang kemudian mendefinisikan variabel `req` untuk `length` dan melakukan random choice yaitu pengocokan acak untuk kata car.

```
import random
class PermuteSentences(object):
    def __init__(self, length):
        self.length = length
    def __iter__(self):
        req = list(self.length)
        random.choice('car')
```

Gambar 5.16 *PermuteSentences* Tasya

5.2.3 Fungsi Librari *gensim* *TaggedDocument* dan *Doc2Vec*

Doc2vec adalah algoritma unsupervised untuk menghasilkan vektor untuk kalimat / paragraf / dokumen. Dan *TaggedDocument* merupakan function dari *Doc2Vec* untuk menampilkan tag kata atau kalimat yang diinginkan dari sebuah dokumen.

- Impor modul *TaggedDocument* dan modul *Doc2Vec*
- Hasilnya seperti berikut

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

Gambar 5.17 *TaggedDocument* Tasya

5.2.4 Menambahkan data Training Dari File Dengan *Doc2Vec*

Yang harus dilakukan yaitu :

1. Import librari `re` atau `request` dari python dan import modul `os` untuk memungkinkan kita menggunakan operasi sistem sesuai yang dibutuhkan.
2. Mendefinisikan variabel `unsup sentences` yang berisikan variabel kosong.
3. Panggil data training dan data testing dari file yang telah disediakan
4. Kemudian coba data tersebut

Untuk lebih jelasnya, berikut hasil percobaan dari praktek yang dilakukan

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

- Untuk skrip diatas mengambil contoh dataset dari Imdb yang kemudian mengambil data training dan data testing nya.
- File tersebut diambil dari folder aclImdb
- File yang dibuka berbentuk teks atau berekstensi txt
- Kemudian file yang dibuka tadi diasiasikan sebagai f
- variabel sent akan membaca f
- Variabel words akan memanggil dan mengirimkan fungsi extract words dari skrip sebelumnya
- KEmudian list dari unsup sentences akan diupdate dengan menambahkan objek baru dan diberi tag. berikut hasilnya

```
In [5]: 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]))
```

Gambar 5.18 Data Training Imdb Tasya

- Untuk contoh dibawah pun sama seperti ynag diatas, bedanya hanya untuk contoh kedua ini dilakukan proses token dan datasetnya dari review polarity.

```
In [6]: for dirname in ["txt_sentoken/pos", "txt_sentoken/neg"]:
...:     for fname in sorted(os.listdir(dirname)):
...:         if fname[-4:] == '.txt':
...:             with open(dirname+"/"+fname, encoding='UTF-8') as f:
...:                 for i, kirim in enumerate(f):
...:                     words = extract_words(kirim)
...:                     unsup_sentences.append(TaggedDocument(words, ["%s/%s-%d" %
...: (dirname, fname, i)]))
```

Gambar 5.19 Data Training Polarity Tasya

- Untuk contoh dibawah pun sama seperti yang diatas, bedanya hanya untuk contoh kedua ini dilakukan proses token dan datasetnya dari review rotten tomatoes.

```
In [11]: with open("stanfordSentimentTreebank/original_rt_snipperts.txt",
encoding='UTF-8') as f:
...:     for i, line in enumerate(f):
...:         words = extract_words(sent)
...:         unsup_sentences.append(TaggedDocument(words, ["rt-%d" % i]))

In [12]: Traceback (most recent call last):
```

Gambar 5.20 Data Training Tomatoes

5.2.5 Mengapa Harus Dilakukan Pengocokan Dan Pembersihan Data

Pengocokan dilakukan untuk mendapatkan hasil yang lebih akurat pada saat melakukan score, karena pengocokan mempengaruhi performa positif negatifnya dari scoring.

Kemudian Pembersihan data dilakukan untuk membersihkan tag spasi ataupun data noisy yang tidak diperlukan dalam dokumen.

- Import dulu librari re atau request
- Kemudian Disini akan mendefinisikan function extract words untuk menghapus tag html, hapus tanda petik dan lainnya
- Setelah itu Data akan dirandomatau dilakukan pengocokan dengan mengimport librari modul dengan mendefinisikan class PermuteSentences. BERikut skrip lengkapnya

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

import random
class PermuteSentences(object):
    def __init__(self, sents):
        self.sents=sents

    def __iter__(self):
        shuffled = list(self.sents)
        random.shuffle(shuffled)
        for sent in shuffled:
            yield sent
```

- Berikut merupakan hasil dari pengocokan dan pembersihan data

```
In [4]: import re
...: def extract_words(sent):
...:     sent = sent.lower()
...:     sent = re.sub(r'<[^>]+>', '', sent) #hapus tag html
...:     sent = re.sub(r'(\w)\s(\w)', ' ', sent) #hapus petik satu
...:     sent = re.sub(r'\s+', ' ', sent) #hapus tanda baca
...:     sent = re.sub(r'\s+', ' ', sent) #hapus spasi yang berurutan
...:     return sent.split()
...:
...:
...: import random
...: class PermuteSentences(object):
...:     def __init__(self, sents):
...:         self.sents=sents
...:
...:     def __iter__(self):
...:         shuffled = list(self.sents)
...:         random.shuffle(shuffled)
...:         for sent in shuffled:
...:             yield sent
```

Gambar 5.21 Pengocokan Dan Pembersihan Data Tasya

5.2.6 Mengapa Model Harus Di Save Dan Temporari Training Harus Dihapus

- Sebelum model di save atau disipman lakukan pengocokan terlebih dahulu seperti berikut.

```
In [54]: mute=PermuteSentences(unsup_sentences)
```

Gambar 5.22 Save Model Tasya

- Kemudian setelah itu gunakan perintah Doc2Vec load dan memasukan nama file setelah disimpan.

```
In [55]: model = Doc2Vec(mute, dm=0,hs=1,vector_size=52)
```

Gambar 5.23 Save Model Tasya

- Ketika model telah di save, barulah hapus temporari dengan skrip berikut. Maka hasil dari kedua praktek diatas adalah sebagai berikut

```
In [56]: model.delete_temporary_training_data(keep_inference=True)
```

Gambar 5.24 Save Model Hasil Tasya

```
In [57]: model.save('haci.d2v')
```

Gambar 5.25 Save Model HasilTasya

Hal yang dapat disimpulkan yaitu : Model disave untuk memudahkan kita dalam meng-edit file, kita tidak perlu mengetik ulang semua skrip dan tinggal membuka file tersebut jika ingin menguji ulang modelnya. Temporari training merupakan data training yang sebelumnya kita gunakan untuk mencoba skripnya, namun karena kita telah membuat modelnya dan untuk menghemat memori dilakukan penghapusan temporari training agar tidak terjadi lag ataupun hal lainnya.

5.2.7 Infer Code

Infer vektor digunakan untuk mengkalkulasikan berapa vektor dari kata yang diberikan. Atau dengan kata lain mengubah kata yang diberikan menjadi bentuk vektor. Dari model yang telah dibuat

- Masukan skrip berikut yang artinya akan memanggil infer vector untuk mengkompres kalimat berikut menjadi bentuk vektor

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

- Hasilnya seperti berikut

```
In [59]: model.infer_vector(extract_words("ahelah cape"))
Out[59]:
array([[ 0.00220989,  0.00535457,  0.00817098, -0.0019864 , -0.00143027,
        -0.00163056,  0.00238419, -0.00621389, -0.0055491 ,  0.0092905 ,
         0.00689598,  0.0016727 , -0.00941525,  0.00544497, -0.00227619,
        -0.00714622, -0.00829009, -0.00784442, -0.00561465, -0.00818796,
        -0.00663066,  0.00073265, -0.00662555,  0.00189114, -0.00272863,
        -0.00585739,  0.00160669, -0.00118292, -0.00352464,  0.00602394,
         0.00848661,  0.00831539,  0.00395764,  0.00161309,  0.0066474 ,
        -0.00597237,  0.00840245, -0.00063893, -0.00362349, -0.0005215 ,
        -0.0076603 , -0.00738003,  0.00465537, -0.00909361, -0.0062611 ,
         0.00567978, -0.00528364,  0.00788782, -0.00389069,  0.00556741,
        -0.00092625, -0.00557183], dtype=float32)
```

Gambar 5.26 Infer Code Tasya

5.2.8 Cosine Similarity

Cosine similarity digunakan untuk melihat kesamaan atau kemiripan dari suatu kalimat/paragraf yang diinginkan. Apakah kalimat tersebut dapat dikategorikan dalam satu kategori atau tidak. Disini saya akan membandingkan beberapa kalimat seperti berikut :

- Masukan perintah berikut dimana sistem akan mengimpor cosine similarity dan kemudian gunakan function infer vector untuk membandingkan dua kalimat berikut.
- Hasilnya seperti berikut, data dilihat bahwa kata

```
from sklearn.metrics.pairwise import cosine_similarity
cosine_similarity(
    [model.infer_vector(extract_words("she going to scho
    [model.infer_vector(extract_words("Services sucks."))
```

```
In [61]: from sklearn.metrics.pairwise import cosine_similarity
...: cosine_similarity(
...:     [model.infer_vector(extract_words("capedeh"))],
...:     [model.infer_vector(extract_words("capedeh"))])
Out[61]: array([[0.9999999]], dtype=float32)
```

Gambar 5.27 Infer Code Tasya

5.2.9 Score Dari Cross Validation

- Pertama kita akan mengimpor KNN RF dan numpy

```
from sklearn.neighbors import KNeighborsClassifier
from sklearn.ensemble import RandomForestClassifier
from sklearn.model_selection import cross_val_score
import numpy as np
```

- Kemudian lakukan cek score dengan perintah berikut

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

- Hasilnya seperti berikut

```
In [67]: scores = cross_val_score(clf, sentvecs, sentiments, cv=5)
...: np.mean(scores), np.std(scores)
Out[67]: (0.4953333333333333, 0.013800161029656305)
```

Gambar 5.28 Score Cross Validation Tasya

5.2.10 Penanganan Error

5.2.10.1 Error Memory

1. Berikut ini merupakan error yang didapatkan saat menjalankan program diatas

MemoryError:

Gambar 5.29 Error Memory Tasya

2. Error diatas merupakan memori eror dimana memori dari ram hampir terpakai semua sehingga program tidak dapat dijalankan. Cara mengatasinya yaitu
 - Membatasi penggunaan memori
 - Menambah kapasitas ram
 - Jika terjadi blank screen atau lag, force shutdown. Dan jangan membuka aplikasi yang memakan banyak memori lainnya.

MFCC DAN NEURAL NETWORK

6.1 Teori

1. Kenapa fi

le suara harus di lakukan MFCC. dilengkapi dengan ilustrasi atau gambar.

Nilai-nilai MFCC meniru pendengaran manusia dan mereka biasanya digunakan dalam aplikasi pengenalan suara serta genre musik deteksi. Nilai-nilai MFCC ini akan dimasukkan langsung ke jaringan saraf. Agar dapat diubah menjadi bentuk vektor, dan dapat digunakan pada machine learning. Disebabkan machine learning hanya mengerti bilangan vektor saja. Ilustrasinya, Ketika ingin menggunakan file suara dalam machine learning, misalnya untuk melihat jam. Machine learning tidak memahami rekaman suara melainkan vektor. Maka rekaman tersebut akan diubah kedalam bentuk vektor kemudian vektor akan menyesuaikan dengan kata kata yang sudah disediakan. Jika cocok maka akan mengembalikan waktu yang diinginkan

2. Konsep dasar neural network dilengkapi dengan ilustrasi atau gambar

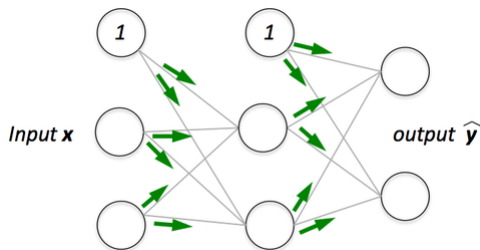
Neural Network ini terinspirasi dari jaringan saraf otak manusia. Dimana setiap neuron terhubung ke setiap neuron di lapisan berikutnya. Lapisan pertama

menerima input dan lapisan terakhir memberikan keluaran. Struktur jaringan, yang berarti jumlah neuron dan koneksinya, diputuskan sebelumnya dan tidak dapat berubah, setidaknya tidak selama training. Juga, setiap input harus memiliki jumlah nilai yang sama. Ini berarti bahwa gambar, misalnya, mungkin perlu diubah ukurannya agar sesuai dengan jumlah neuron input.

Ilustrasinya. misalkan kita ingin encode sebuah kalimat yaitu "what time is it" kemudian Anda menginisialisasi lapisan jaringan Anda dan hidden statel. Bentuk dan dimensi hidden state akan tergantung pada bentuk dan dimensi jaringan saraf berulang Anda. Kemudian Anda mengulangi input Anda, meneruskan kata dan hidden state ke NN. NN mengembalikan output dan kondisi tersembunyi yang dimodifikasi. Anda terus mengulang sampai Anda kehabisan kata-kata. Terakhir Anda melewati output ke layer feedforward, dan itu mengembalikan prediksi. Bahwa kita ingin mengetahui pukul berapa sekarang.

3. Konsep pembobotan dalam neural network. dilengkapi dengan ilustrasi atau gambar

Bobot mewakili kekuatan koneksi antar unit. Jika bobot dari node 1 ke node 2 memiliki besaran lebih besar, itu berarti bahwa neuron 1 memiliki pengaruh lebih besar terhadap neuron. 2. Bobot penting untuk nilai input. Bobot mendekati nol berarti mengubah input ini tidak akan mengubah output. Bobot negatif berarti meningkatkan input ini akan mengurangi output. Bobot menentukan seberapa besar pengaruh input terhadap output. Seperti contoh berikut :



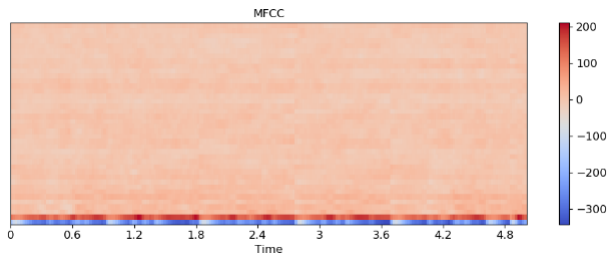
Gambar 6.1 Contoh Pembobotan Neural Network Tasya

4. Konsep fungsi aktivasi dalam neural network. dilengkapi dengan ilustrasi atau gambar

Fungsi aktivasi digunakan untuk memperkenalkan non-linearitas ke jaringan saraf. Ini menekan nilai dalam rentang yang lebih kecil yaitu. fungsi aktivasi Sigmoid memeras nilai antara rentang 0 hingga 1. Ada banyak fungsi aktivasi yang digunakan dalam industri pembelajaran yang dalam dan ReLU, SeLU dan TanH lebih disukai daripada fungsi aktivasi sigmoid. Ilustrasinya, ketika fungsi aktivasi linier, jaringan saraf dua lapis mampu mendekati hampir semua fungsi. Namun, jika fungsi aktivasi identik dengan fungsi aktivasi $F(X) = X$, properti

ini tidak puas, dan jika MLP menggunakan fungsi aktivasi yang sama, seluruh jaringan setara dengan jaringan saraf lapis tunggal.

5. Cara membaca hasil plot dari MFCC, dilengkapi dengan ilustrasi atau gambar Berikut merupakan hasil plot dari rekaman suara : Dari gambar tersebut dapat

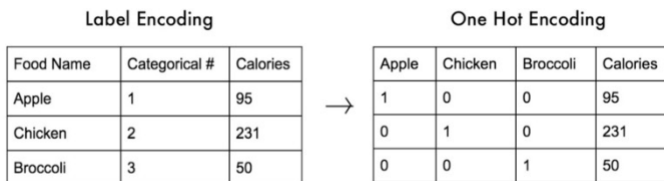


Gambar 6.2 Cara Membaca Hasil Plot MFCC Tasya

diketahui :

- Terdapat 2 dimensi yaitu x sebagai waktu, dan y sebagai power atau desibel.
 - Dapat dilihat bahwa jika berwarna biru maka power dari suara tersebut rendah, dan jika merah power dari suara tersebut tinggi
 - Dibagian atas terdapat warna merah pudar yang menandakan bahwa tidak ada suara sama sekali dalam jangkauan tersebut.
6. Jelaskan apa itu one-hot encoding, dilengkapi dengan ilustrasi kode dan atau gambar.

One-hot encoding adalah representasi variabel kategorikal sebagai vektor biner. Mengharuskan nilai kategorikal dipetakan ke nilai integer. Kemudian, setiap nilai integer direpresentasikan sebagai vektor biner yang semuanya bernilai nol kecuali indeks integer, yang ditandai dengan 1.



Gambar 6.3 One Hot Encoding Tasya

7. fungsi dari np.unique dan to categorical dalam kode program, dilengkapi dengan ilustrasi atau gambar. Untuk np unique fungsinya yaitu menemukan elemen

unik array. Mengembalikan elemen unik array yang diurutkan. Ada tiga output opsional selain elemen unik:

- Indeks array input yang memberikan nilai unik
- Indeks array unik yang merekonstruksi array input
- Berapa kali setiap nilai unik muncul dalam array input.

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

Gambar 6.4 Numpy Unique Tasya

Untuk To Categorical fungsinya untuk mengubah vektor kelas (integer) ke matriks kelas biner.

```
# 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
```

Gambar 6.5 To Categorical Tasya

8. Fungsi dari Sequential dalam kode program, dilengkapi dengan ilustrasi atau gambar. Sequential berfungsi sebagai tumpukan linear lapisan. Contohnya sebagai berikut :

```
from keras.models import Sequential
from keras.layers import Dense

model = Sequential()
model.add(Dense(2, input_dim=1))
model.add(Dense(1))
```

Gambar 6.6 Sequential Tasya

6.2 Praktek Program

6.2.1 GTZAN Genre Collection dan data dari freesound

1. GTZAN Genre Collection berisikan klasifikasi genre musik. Terdapat 1000 audio dengan durasi maksimal 30 detik dan terdapat 10 genre musik didalamnya. Dalam setiap genre berisikan 100 tracks musik.

2. Data dari Freesound berisikan instrument alat musik tertentu dalam bentuk wav

Untuk Meload Data tersebut untuk digunakan pada MFCC caranya dapat dilihat seperti pada listing berikut.

```

1 \begin{verbatim}
2 import librosa
3 import librosa.feature
4 import librosa.display
5 import glob
6 import numpy as np
7 import matplotlib.pyplot as plt
8 from keras.models import Sequential
9 from keras.layers import Dense, Activation
10 from keras.utils.np_utils import to_categorical
11
12 # In []
13 audio_path = 'genres/classical/classical.00010.au'
14 x, sr = librosa.load(audio_path)
15 print(type(x), type(sr))
16 \end{verbatim}

```

Listing 6.1 Kode Load Data Untuk MFCC

- Penjelasannya sebagai berikut :
- Codingan Diatas akan meload libray librosa yang akan digunakan untuk menggunakan mfcc
- Librosafeature akan meload feature dari librosa
- Librosadisplay akan mengambil fungsi display pada librosa
- glob merupakan modul pada python yang digunakan untuk meload segala jenis format file termasuk musik
- mengimport numpy sebagai np yang digunakan untuk data array dari musik
- import matplotlib untuk melakukan plotting dari audio
- Mengimport modul Sequential dari librari Keras untuk membuat suatu model
- Dense dan Activation sebagai Operasi linier di mana setiap input terhubung ke setiap output dengan bobot atau weight.
- Dari library keras akan meload modul to_categorical
- Kemudian untuk me load datanya, disini variabel audio_path berisikan direktori file tujuan yang digunakan.
- Variabel x dan sr berguna untuk meload variabel audio_path menggunakan librari Librosa
- Kemudia print atau tampilkan x dan s dalam bentuk array.Hasilnya seperti berikut :

```
In [28]: audio_path = 'genres/classical/classical.00010.au'
...: x, sr = librosa.load(audio_path)
...: print(type(x), type(sr))
<class 'numpy.ndarray'> <class 'int'>
```

Gambar 6.7 Meload Data Genre Collection Tasya

6.2.2 Fungsi Display MFCC

Berikut merupakan Code dari fungsi Display mfcc: Penjelasan dari code diatas yaitu

```
In [3]: def display_mfcc(song):
...:     y, _ = librosa.load(song)
...:     mfcc = librosa.feature.mfcc(y)
...:
...:     plt.figure(figsize=(10, 4))
...:     librosa.display.specshow(mfcc, x_axis='time', y_axis='mel')
...:     plt.colorbar()
...:     plt.title(song)
...:     plt.tight_layout()
...:     plt.show()
```

Gambar 6.8 Display MFCC Tasya

:

- def display_mfcc yaitu kita akan mendefinisikan fungsi yang diberinama display_mfcc dengan inputan song
- Variabel y akan meload variabel song
- Variabel MFCC akan menggunakan feauture mfcc pada Librosa untuk melakukan konversi audio menjadi bentuk vektor
- Kemudian hasil tadi akan diplotting.
- Berikut merupakan contoh dari plotting audio dari genre Classical. Ketikan kode berikut yang dimana akan memanggil fungsi dispkay mfcc utuk plotting dari audioyang dituju dapat dilihat dilisting berikut .

```
1 \begin{verbatim}
2 display_mfcc('genres/classical/classical.00010.au')
3 \end{verbatim}
```

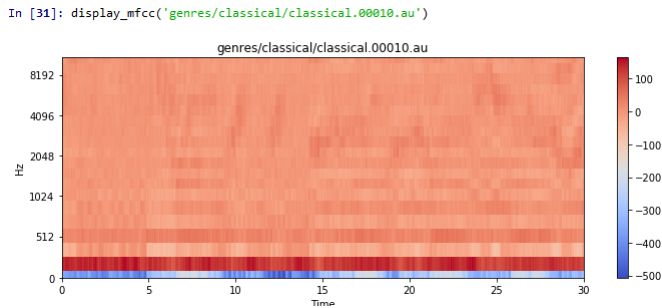
Listing 6.2 Code Fungsi Display MFCC

- Hasilnya sebagai berikut

6.2.3 Fungsi Extract Features Song

Berikut merupakan code dari fungsi extract features song : Penjelasan dari code diatas yaitu :

- Variabel y akan melaod variabel atau inputan f menggunakan librari Librosa
- Variabel mfcc akan melakukan mfcc dari variabel y



Gambar 6.9 Hasil Display MFCC Tasya

```
In [7]: def extract_features_song(f):
...:     y, _ = librosa.load(f)
...:     # get Mel-frequency cepstral coefficients
...:     mfcc = librosa.feature.mfcc(y)
...:     # normalize values between -1,1 (divide by max)
...:     mfcc /= np.amax(np.absolute(mfcc))
...:     return np.ndarray.flatten(mfcc)[:25000]
```

Gambar 6.10 Extract Features Tasya

- Variabel mfcc kemudian akan dibagi oleh numpy amax dan dikembalikan lagi nilainya ke variabel mfcc.
- Hasil Tadi kemudian akan ditampilkan dalam bentuk array dengan mengambil 25000 row pertama
- Mengapa mengambil 25000 row pertama? dikarenakan Audio yang terdapat pada dataset ini tidak menentu durasinya. Dan kita harus mengambil data yang memiliki durasi yang sama untuk mempermudah dalam melakukan training.

6.2.4 Fungsi Generate Features And Labels

Berikut merupakan code dari fungsi Generate Features And Labels : Penjelasan dari code diatas yaitu :

- Variabel all features berisikan array kosong
- Variabel all labels berisikan array kosong
- Variabel genres disesuaikan dengan nama folder sebelumnya, dan berisikan folder folder dari genre yang ada
- Melakukan looping, untuk folder tadi
- Variabel sound_files akan mengambil file dari folder genres dan mengambil semua file dengan ekstensi au didalamnya.

```

In [8]: def generate_features_and_labels():
...:     all_features = []
...:     all_labels = []
...:
...:     genres = ['blues', 'classical', 'country', 'disco', 'hiphop', 'jazz',
'metal', 'pop', 'reggae', 'rock']
...:     for genre in genres:
...:         sound_files = glob.glob('genres/'+genre+'/*.au')
...:         print('Processing %d songs in %s genre...' % (len(sound_files), genre))
...:         for f in sound_files:
...:             features = extract_features_song(f)
...:             all_features.append(features)
...:             all_labels.append(genre)
...:
...:     # convert Labels to one-hot encoding cth blues : 1000000000 classic
0100000000
...:     label_uniq_ids, label_row_ids = np.unique(all_labels,
return_inverse=True)#ke integer
...:     label_row_ids = label_row_ids.astype(np.int32, copy=False)
...:     onehot_labels = to_categorical(label_row_ids, len(label_uniq_ids))#ke one
hot
...:     return np.stack(all_features), onehot_labels

```

Gambar 6.11 Fungsi Generate Features And Labels Tasya

- print akan menampilkan Tulisan Prossesing dengan jumlah file didalam folder dan nama folder tersebut
- Memanggil fungsi `extract_features_song` kedalam inputan `f` dalam `sound_files` dan melakukan vektorisasi dimasukan kedalam variabel `features`.
- Semua fitur akan dimasukan kedalam `all_features`.
- Semua genre dimasukan ke `all_labels`.
- Variabel `label_uniq_ids` dan `label_row_ids` mendefinisikan label unique dari `all_labels` kedalam bentuk integer.
- Kemudian variabel `onehot_labels` akan mengubahnya ke dalam bentuk one hot encoding dengan menggunakan `to_categorical`. Sehingga dimensinya menjadi 1000 x 10 dikarenakan terdapat 1000 lagu dan 10 binari untuk merepresentasikan one-hot encodingnya.
- Mengembalikan `all_features` dan `onehot_labels` kedalam satu matriks.

6.2.5 Penggunaan Fungsi Generate Features And Labels Sangat Lama Ketika Meload Dataset Genre

Dikarenakan Terdapat 10 folder dengan genre berbeda, dan didalamnya terdapat 100 audio. Dari setiap folder itu akan dilakukan features dan perubahan label. Karena banyaknya jumlah file maka proses loadnya pun lama. Berikut codingannya :

```

1 \begin{verbatim}
2 features , labels = generate_features_and_labels ()
3 \end{verbatim}

```

Listing 6.3 Panggil Genenrate Labels

Akan didapatkan hasil seperti berikut yang dimana menunjukkan proses bahwa sedang dilakukan ekstraksi audio ke `features` dan `labels` :

```
In [9]: features, labels = generate_features_and_labels()
Processing 100 songs in blues genre...
Processing 100 songs in classical genre...
Processing 100 songs in country genre...
Processing 100 songs in disco genre...
Processing 100 songs in hip-hop genre...
Processing 100 songs in jazz genre...
Processing 100 songs in metal genre...
Processing 100 songs in pop genre...
Processing 100 songs in reggae genre...
Processing 100 songs in rock genre...
```

Gambar 6.12 Hasil Fungsi Generate Features And Labels Tasya

6.2.6 Pemisahan Data Training Dan Data Set Sebesar 80%

Pemisahan 80% digunakan untuk memudahkan dalam melakukan pengacakan atau pengocokan nantinya. Dimana 80% merupakan data training dan sisanya 20% merupakan data testnya. data training perlu lebih banyak agar saat dilakukan pengocokan tidak teracak dalam urutan yang berbeda. Berikut code programnya pada listing ini :

```
1 \begin{verbatim}
2 training_split = 0.8
3
4 # last column has genre, turn it into unique ids
5 alldata = np.column_stack((features, labels))
6
7 np.random.shuffle(alldata)
8 splitidx = int(len(alldata) * training_split)
9 train, test = alldata[:splitidx, :], alldata[splitidx :, :]
10
11 print(np.shape(train))
12 print(np.shape(test))
13
14 train_input = train[:, :-10]
15 train_labels = train[:, -10:]
16
17 test_input = test[:, :-10]
18 test_labels = test[:, -10:]
19
20 print(np.shape(train_input))
21 print(np.shape(train_labels))
22 \end{verbatim}
```

Listing 6.4 Code Pemisahan Data Training Dan Testing

Penjelasan dari code diatas :

- Training split akan memisahkan training set sebanyak 80%
- Melakukan penumpukan features dan labels
- Melakukan Pengocokan untuk alldata dengan mengalikan isi dari alldata dengan training_split
- Memisahkan mana yang termasuk data train dan mana yang termasuk data test

- Menampilkan isi dari train dan test. Dapat dilihat bahwa untuk training terdapat 800 kolom dan untuk test 200 kolom dengan jumlah baris yang sama yaitu 25010

```
In [13]: np.random.shuffle(alldata)
...: splitidx = int(len(alldata) * training_split)
...: train, test = alldata[:splitidx,:], alldata[splitidx,:]

In [14]: print(np.shape(train))
...: print(np.shape(test))
(800, 25010)
(200, 25010)
```

Gambar 6.13 Pemisahan Data Training dan Data Set Tasya

- Variabel `train_input` akan berisikan train dengan mengecualikan 10 baris terakhir
- Variabel `train_labels` berisikan train dengan mengambil sisa dari `train_input` atau hanya mengambil 10 baris terakhir saja
- Untuk variabel `test_input` dan `test_label` sama penjelasannya seperti diatas.
- Baris selanjutnya digunakan untuk menampilkan isi atau shape dari hasil training dan testing barusan, seperti berikut :

```
In [15]: train_input = train[:,-10]
...: train_labels = train[:,-10:]

In [16]: test_input = test[:,-10]
...: test_labels = test[:,-10:]

In [17]: print(np.shape(train_input))
...: print(np.shape(train_labels))
(800, 25000)
(800, 10)
```

Gambar 6.14 Pemisahan Data Training dan Data Set Tasya

6.2.7 Fungsi Sequential

Berikut code lengkapnya :

```
1 \begin{verbatim}
2 model = Sequential([
3     Dense(100, input_dim=np.shape(train_input)[1]),
4     Activation('relu'),
5     Dense(10),
6     Activation('softmax'),
7 ])
8 \end{verbatim}
```

Listing 6.5 Code Fungsi Sequential

Hasilnya seperti berikut : Dari hasil diatas dapat dijelaskan bahwa :

- Layer pertama dense dari 100 neuron untuk inputan

```
In [10]: model = Sequential([
...:     Dense(100, input_dim=np.shape(train_input)[1]),
...:     Activation('relu'),
...:     Dense(10),
...:     Activation('softmax'),
...: ])
WARNING:tensorflow:From E:\Anaconda3\lib\site-packages\tensorflow\python\framework\
\op_def_library.py:263: colocate_with (from tensorflow.python.framework.ops) is
deprecated and will be removed in a future version.
Instructions for updating:
Colocations handled automatically by placer.
```

Gambar 6.15 Pemisahan Data Training dan Data Set Tasya

- 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 output nya.
- Untuk dense diatas aktivasinya menggunakan fungsi Softmax

6.2.8 Fungsi Compile

Berikut code lengkapnya :

```
1 \begin{verbatim}
2 model.compile(optimizer='adam',
3               loss='categorical_crossentropy',
4               metrics=['accuracy'])
5 print(model.summary())
6 \end{verbatim}
```

Listing 6.6 Code Fungsi Compile

Hasilnya seperti berikut : Dari hasil diatas dapat dijelaskan bahwa :

```
In [19]: model.compile(optimizer='adam',
...:                   loss='categorical_crossentropy',
...:                   metrics=['accuracy'])
...: print(model.summary())
```

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

None

Gambar 6.16 Fungsi Compile Tasya

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

6.2.9 Fungsi Fit

Berikut code lengkapnya :

```
1 \begin{verbatim}
2 model.fit(train_input , train_labels , epochs=10, batch_size=32,
3           validation_split=0.2)
4 \end{verbatim}
```

Listing 6.7 Code Fungsi Fit

Hasilnya seperti berikut : Dari hasil diatas dapat dijelaskan bahwa :

```
In [20]: model.fit(train_input, train_labels, epochs=10, batch_size=32,
...:               validation_split=0.2)
WARNING:tensorflow:From E:\Anaconda3\lib\site-packages\tensorflow\python\ops\
\math_ops.py:3066: to_int32 (from tensorflow.python.ops.math_ops) is deprecated and
be removed in a future version.
Instructions for updating:
Use tf.cast instead.
Train on 640 samples, validate on 160 samples
Epoch 1/10
640/640 [=====] - 8s 13ms/step - loss: 2.1030 - acc: 0.2781
val_loss: 1.8095 - val_acc: 0.3500
Epoch 2/10
640/640 [=====] - 2s 4ms/step - loss: 1.3568 - acc: 0.5422 -
val_loss: 1.7641 - val_acc: 0.3937
Epoch 3/10
640/640 [=====] - 2s 3ms/step - loss: 1.0330 - acc: 0.6750 -
val_loss: 1.5691 - val_acc: 0.4125
Epoch 4/10
640/640 [=====] - 2s 3ms/step - loss: 0.7832 - acc: 0.7797 -
val_loss: 1.7463 - val_acc: 0.4188
Epoch 5/10
640/640 [=====] - 2s 3ms/step - loss: 0.5983 - acc: 0.8406 -
val_loss: 1.5068 - val_acc: 0.4688
Epoch 6/10
640/640 [=====] - 2s 3ms/step - loss: 0.4406 - acc: 0.9266 -
val_loss: 1.5222 - val_acc: 0.4875
Epoch 7/10
640/640 [=====] - 2s 3ms/step - loss: 0.3159 - acc: 0.9688 -
val_loss: 1.5670 - val_acc: 0.4500
```

Gambar 6.17 Fungsi Fit Tasya

- Melakukan pelatihan dengan epoch atau iterasi dengan ramatan balik sebanyak 10, kemudian dalam sekali epochs dilakukan 32 sampel yang diproses sebelum model diperbarui.
- Validation_split sebesar 20% untuk melakukan pengecekan pada cross score validation

6.2.10 Fungsi Evaluate

Berikut code lengkapnya :

```
1 \begin{verbatim}
2 loss , acc = model.evaluate(test_input , test_labels , batch_size=32)
3
4 print("Done!")
5 print("Loss: \%.4f, accuracy: \%.4f" % (loss , acc))
6 \end{verbatim}
```

Listing 6.8 Code Fungsi Evaluate

```
In [21]: loss, acc = model.evaluate(test_input, test_labels, batch_size=32)
200/200 [=====] - 0s 2ms/step
```

Gambar 6.18 Fungsi Evaluate Tasya

Hasilnya seperti berikut : Dari hasil diatas dapat dijelaskan bahwa :

- Melakukan evaluasi atau menemukan model terbaik yang mewakili data dan seberapa baik model yang dipilih akan bekerja di masa depan. Menggunakan test input dan test label.
- Kemudian Hasilnya dapat dilihat seperti berikut :

```
In [22]: print("Done!")
...: print("Loss: %.4f, accuracy: %.4f" % (loss, acc))
Done!
Loss: 1.3620, accuracy: 0.5350
```

Gambar 6.19 Fungsi Evaluate Tasya

- Dimana Loss yaitu hasil prediksi yang salah sebanyak 1,3620 dan keakurasian prediksinya yaitu 53%

6.2.11 Fungsi Predict

Berikut code lengkapnya :

```
1 \begin{verbatim}
2 model.predict(test_input[:1])
3 \end{verbatim}
```

Listing 6.9 Code Fungsi Predict

Hasilnya seperti berikut : Dari hasil diatas dapat dijelaskan bahwa :

```
In [23]: model.predict(test_input[:1])
Out[23]:
array([[2.5113127e-01, 7.7192662e-03, 5.1440198e-02, 6.4860745e-03,
        6.4208927e-03, 6.2923378e-01, 3.7388336e-06, 2.6338635e-02,
        3.9711967e-03, 1.7254945e-02]], dtype=float32)
```

Gambar 6.20 Fungsi Predict Tasya

- Untuk melakukan prediksi diambil satu baris dari test_input
- Nilai yang tertinggi terdapat pada label kedua atau genre Classical
- Untuk baris yang dipilih prediksi yang tepat yaitu lagu tersebut termasuk kedalam genre Classical

6.3 Penanganan Error

6.3.1 Module Error

Berikut merupakan eror yang dijumpai ketika menjalankan skrip diatas

```
...: 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-1-3bc11cc81b24>", line 1, in <module>
    import librosa

ModuleNotFoundError: No module named 'librosa'
```

Gambar 6.21 Module Error Tasya

- Jenis error tersebut adalah ModuleNotFoundError : No Module named 'librosa' . Error ini terjadi dikarenakan target atau tujuannya tidak terinstall. Maka yang perlu dilakukan yaitu :

- Buka Anconda atau conda prompt
- Ketikan 'conda install -c conda-forge librosa dan enter
- Tunggu sampai proses instalasi berhasil
- Jika Sudah, jalankan kembali skrip tadi maka hasilnya seperti berikut

```
In [2]: 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
Using TensorFlow backend.
```

Gambar 6.22 Penyelesaian Module Error Tasya

- Tandanya eror sudah berhasil ditangani

BAB 7

CNN

7.1 Teori

7.1.1 Teks Tokenizer

Untuk memudahkan mesin memahami maksud dari apa yang kita inginkan dalam machine learning, kata pada teks disebut token, dan proses vektorisasi dari bentuk kata ke dalam token tersebut disebut tokenizer dan tokenizer akan merubah sebuah teks menjadi simbol, kata, ataupun biner dan bentuk lainnya kedalam token. Untuk lebih jelasnya perhatikan ilustrasi berikut. Disini saya mempunyai sebuah kalimat yaitu "Nama Saya Tasya Wiendhyra" maka ketika kita lakukan proses tokenizer maka akan berubah menjadi ['Nama', 'Saya', 'Tasya', 'Wiendhyra'].

7.1.2 konsep dasar K Fold Cross Validation pada dataset komentar Youtube

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

Listing 7.1 K Fold Cross Validation

StratifiedKFold berisikan presentasi sampel untuk setiap kelas. Dimana dalam ilustrasi ini sampel dibagi menjadi 5 dalam setiap class nya. Kemudian sampel tadi akan dimasukkan kedalam class dari dataset youtube tadi.

Untuk ilustrasi lebih jelasnya, ada pada gambar berikut :

```
>>> from sklearn.model_selection import StratifiedKFold
>>> X = np.array([[1, 2], [3, 4], [1, 2], [3, 4]])
>>> y = np.array([0, 0, 1, 1])
>>> skf = StratifiedKFold(n_splits=2)
>>> skf.get_n_splits(X, y)
2
>>> print(skf)
StratifiedKFold(n_splits=2, random_state=None, shuffle=False)
>>> for train_index, test_index in skf.split(X, y):
...     print("TRAIN:", train_index, "TEST:", test_index)
...     X_train, X_test = X[train_index], X[test_index]
...     y_train, y_test = y[train_index], y[test_index]
TRAIN: [1 3] TEST: [0 2]
TRAIN: [0 2] TEST: [1 3]
```

7/1164086/Teori/chapter7tasya1.PNG

Gambar 7.1 Ilustrasi KFold Cross Tasya

7.1.3 kode program for train, test in splits

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.

7.1.4 Jelaskan apa maksudnya kode program *train_content = df['CONTENT'].iloc[train_idx]* dan *test_content = df['CONTENT'].iloc[test_idx]*. dilengkapi dengan ilustrasi atau gambar

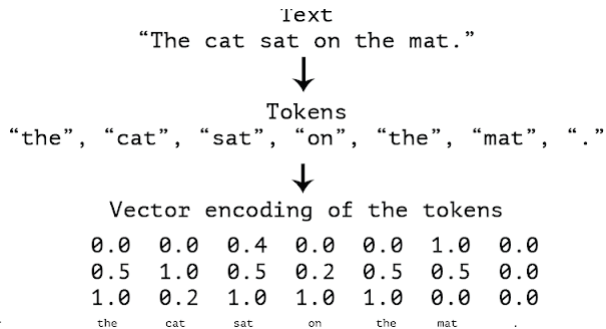
Maksudnya yaitu mengambil data pada kolom atau index CONTENT yang merupakan bagian dari train_idx dan test_idx. Ilustrasinya, ketika data telah diubah menjadi train dan test maka kita dapat memilihnya untuk ditampilkan pada kolom yang diinginkan.

7.1.5 Soal No. 5 Jelaskan apa maksud dari fungsi *tokenizer = Tokenizer(num_words=2000)* dan *tokenizer.fit_on_texts(train_content)*, dilengkapi dengan ilustrasi atau gambar

Dimana variabel tokenizer akan melakukan vektorisasi kata menggunakan fungsi Tokenizer yang dimana jumlah kata yang ingin diubah kedalam bentuk token adalah 2000 kata. Dan untuk *tokenizer.fit_on_texts(train_content)* maksudnya kita akan melakukan fit tokenizer hanya untuk data trainnya saja tidak dengan data test nya untuk kolom CONTENT. 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.

7.1.6 Jelaskan apa maksud dari fungsi `d_train_inputs = tokenizer.texts_to_matrix(mode='tfidf')` dan `d_test_inputs = tokenizer.texts_to_matrix(test_content, mode='tfidf')`, dilengkapi dengan ilustrasi kode dan atau gambar

Maksudnya yaitu untuk variabel `d_train_inputs` akan melakukan tokenizer dari bentuk teks ke matrix dari data `train_content` dengan mode matriksnya yaitu `tfidf` begitu juga dengan variabel `d_test_inputs` untuk data test. Berikut gambar ilustrasinya



7/1164086/Teori/chapter7tasya2.png

Gambar 7.2 Ilustrasi Text To Matrix Tasya

7.1.6.1 Jelaskan apa maksud dari fungsi `d_train_inputs = d_train_inputs/np.amax(np.abs(d_train_inputs))` dan `d_test_inputs = d_test_inputs/np.amax(np.absolute(d_test_inputs))`, dilengkapi dengan ilustrasi atau gambar Fungsi tersebut akan membagi matrix `tfidf` tadi dengan `amax` yaitu mengembalikan maksimum array atau maksimum sepanjang sumbu. Yang hasilnya akan dimasukkan kedalam variabel `d_train_inputs` untuk data train dan `d_test_inputs` untuk data test dengan nominal absolut atau tanpa ada bilangan negatif dan koma.

```
>>> x = np.array([-1.2, 1.2])
>>> np.absolute(x)
array([ 1.2,  1.2])
```

7/1164086/Teori/chapter7tasya4.png

Gambar 7.3 Ilustrasi np Absolute Tasya

7.1.6.2 Jelaskan apa maksud fungsi dari `d_train_outputs = np_utils.to_categorical(d['CLASS'])` dan `d_test_outputs = np_utils.to_categorical(d['CLASS']).iloc[test_idx]` dalam kode program, dilengkapi dengan ilustrasi atau gambar Dalam variabel `d_train_output` dan `d_test_outputs` akan dilakukan one hot encoding, dimana `np_utils` akan mengubah vektor dengan bentuk integer ke matriks kelas biner untuk kolom `CLASS` dimana nantinya hanya akan ada dua pilihan yaitu 1 atau 0. 1 untuk spam 0 untuk non spam atau sebaliknya. Berikut gambar ilustrasinya :

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

```

> 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

```

7/1164086/Teori/chapter7tasya5.png

Gambar 7.4 Ilustrasi One Hot Encoding Tasya

```

1 model = Sequential()
2 model.add(Dense(512, input_shape=(2000,)))
3 model.add(Activation('relu'))
4 model.add(Dropout(0.5))
5 model.add(Dense(2))
6 model.add(Activation('softmax'))

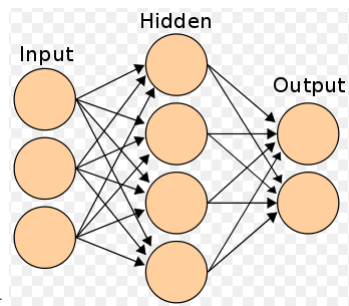
```

Listing 7.2 Membuat model Neural Network

Penjelasannya sebagai berikut :

- Melakukan pemodelan Sequential
- Layer pertama dense dari 512 neuron untuk inputan dengan inputan tadi yang sudah dijadikan matriks sebanyak 2000
- Activationnya menggunakan fungsi relu yaitu jika ada inputan dengan nilai maksimum maka inputan itu yang akan terpilih.
- Dropout ini untuk melakukan pembobotan, dimana pembobotan hanya dilakukan 50% saja agar tidak terjadi penumpukan data dari dense inputan tadi
- Dense 2 mengkategorikan 2 neuron untuk output nya yaitu 1 dan 0.
- Untuk dense diatas aktivasinya menggunakan fungsi Softmax.

Ilustrasinya seperti berikut :



7/1164086/Teori/chapter7tasya6.png

Gambar 7.5 Ilustrasi Neural Network Pemodelan Tasya

7.1.6.4 *Jelaskan apa maksud dari fungsi di listing 7.3 dengan parameter tersebut*

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

Listing 7.3 Compile model

Melakukan peng compile-an dari model Sequential tadi dengan Loss yang merupakan fungsi optimisasi skor menggunakan `categorical_crossentropy`, dan menggunakan algoritma adam sebagai optimizer. Adam yaitu algoritma pengoptimalan yang dapat digunakan sebagai ganti dari prosedur penurunan gradien stokastik klasik untuk memperbarui bobot jaringan yang berulang berdasarkan data training. Dengan metrik yaitu fungsi yang digunakan untuk menilai kinerja mode Anda disini menggunakan fungsi `accuracy`.

7.1.6.5 *Jelaskan apa itu Deep Learning* Deep Learning adalah subbidang machine learning yang berkaitan dengan algoritma yang terinspirasi oleh struktur dan fungsi otak yang disebut jaringan saraf tiruan atau Artificial Neural Networks. Jaringan saraf tiruan, algoritma yang terinspirasi oleh otak manusia, belajar dari sejumlah besar data. Demikian pula dengan bagaimana kita belajar dari pengalaman, algoritma pembelajaran yang mendalam akan melakukan tugas berulang kali, setiap kali sedikit mengubahnya untuk meningkatkan hasilnya.

7.1.6.6 *Jelaskan apa itu Deep Neural Network, dan apa bedanya dengan Deep Learning* Deep Neural Network adalah jaringan syaraf tiruan (JST) dengan beberapa lapisan antara lapisan input dan output. DNN menemukan manipulasi matematis yang benar untuk mengubah input menjadi output, apakah itu hubungan linear atau hubungan non-linear. Merupakan jaringan syaraf dengan tingkat kompleksitas tertentu, jaringan syaraf dengan lebih dari dua lapisan. Deep Neural Network menggunakan pemodelan matematika yang canggih untuk memproses data dengan cara yang kompleks.

DNN hanya terdiri dari dua lapisan yaitu input dan output, sedangkan dalam Deep learning kita dapat mendefinisikan layer sebanyak yang kita inginkan atau butuhkan.

7.1.6.7 *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* Stridenya 3

- terdapat data seperti berikut
- Kemudian hitung konvolusi untuk setiap matriksnya seperti berikut :
 - pertama
 - Kedua
 - Ketiga
 - Keempat
 - Kelima
- Didapatkan hasil akhir nilai konvolusi dan juga max poolingnya seperti berikut

7/1164086/Teori/chapter7tasya7.png

1	2	5	2	1	4
3	2	5	1	1	5
3	2	4	1	5	3
5	3	5	1	3	5
5	1	1	5	5	1
5	5	4	3	3	1

Dengan $g(x,y)$

0	2	-1
-1	0	2
0	1	1

Gambar 7.6 Algoritma Konvolusi Tasya

1	2	5	2	1	4
3	2	5	1	1	5
3	2	4	1	5	3
5	3	5	1	3	5
5	1	1	5	5	1
5	5	4	3	3	1

0	2	-1
-1	0	2
0	1	1

konvolusi→

	12				

$$(0*1)+(2*2)+(-1*5)+(-1*3)+(0*2)+(2*5)+(0*3)+(1*2)+(1*4)=12$$

1	2	5	2	1	4
3	2	5	1	1	5
3	2	4	1	5	3
5	3	5	1	3	5
5	1	1	5	5	1
5	5	4	3	3	1

0	2	-1
-1	0	2
0	1	1

konvolusi→

	12	6			

$$(0*5)+(2*2)+(-1*1)+(-1*5)+(0*1)+(2*1)+(0*4)+(1*1)+(1*5)=6$$

7/1164086/Teori/chapter7tasya8.png

Gambar 7.7 Algoritma Konvolusi Tasya

1	2	5	2	1	4
3	2	5	1	1	5
3	2	4	1	5	3
5	3	5	1	3	5
5	1	1	5	5	1
5	5	4	3	3	1

0	2	-1
-1	0	2
0	1	1

konvolusi→

	12	6	15		

$$(0*2)+(2*1)+(-1*4)+(-1*1)+(0*1)+(2*5)+(0*1)+(1*5)+(1*3)=15$$

1	2	5	2	1	4
3	2	5	1	1	5
3	2	4	1	5	3
5	3	5	1	3	5
5	1	1	5	5	1
5	5	4	3	3	1

0	2	-1
-1	0	2
0	1	1

konvolusi→

	12	6	15		
	7				

$$(0*3)+(2*2)+(-1*4)+(-1*5)+(0*3)+(2*5)+(0*5)+(1*1)+(1*1)=7$$

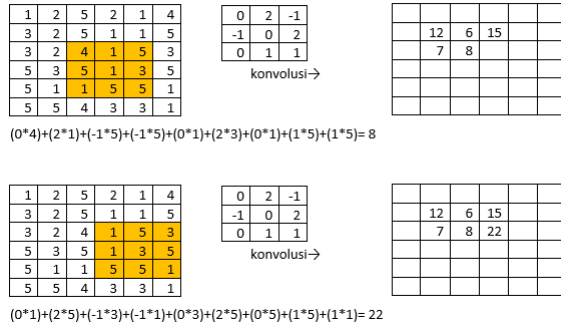
7/1164086/Teori/chapter7tasya9.png

Gambar 7.8 Algoritma Konvolusi Tasya

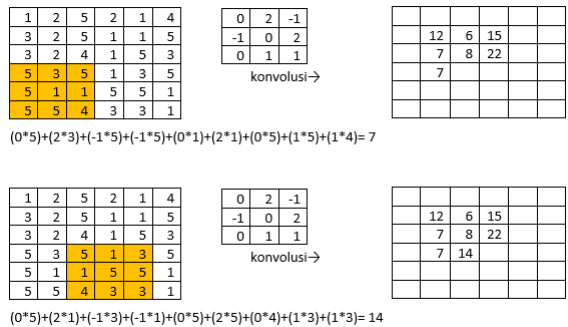
7.1.7 Praktek

Keterangannya sebagai berikut :

- Pertama kita akan mengimpor librari csv
- Dimana dari librai PIL atau Pillow atau Python Imaging Library akan diimpor modul Image yang di inisiasikan sebagai pil_image. Modul Image menyedi-



7/1164086/Teori/chapter7tasya10.png

Gambar 7.9 Algoritma Konvolusi Tasya

7/1164086/Teori/chapter7tasya11.png

Gambar 7.10 Algoritma Konvolusi Tasya

akan kelas dengan nama yang sama yang digunakan untuk mewakili gambar PIL. Modul ini juga menyediakan sejumlah fungsi pabrik, termasuk fungsi untuk memuat image dari file, dan untuk membuat image baru.

- mengimpor librari image dari keras .Yang menghasilkan kumpulan data gambar tensor dengan augmentasi data waktu nyata. Data akan diulang (dalam batch).
- Berikut Hasilnya :

```

1  for row in csvreader:
2      if i > 0:
3          img = keras.preprocessing.image.img_to_array(pil_image.
open("HASYv2/" + row[0]))

```

1	2	5	2	1	4
3	2	5	1	1	5
3	2	4	1	5	3
5	3	5	1	3	5
5	1	1	5	5	1
5	5	4	3	3	1

0	2	-1
-1	0	2
0	1	1

konvolusi→

	12	6	15		
	7	8	22		
	7	14	1		

$$(0*1)+(2*3)+(-1*5)+(-1*5)+(0*5)+(2*1)+(0*3)+(1*3)+(1*1)=1$$

7/1164086/Teori/chapter7tasya12.png

Gambar 7.11 Algoritma Konvolusi Tasya

Hasil konvolusi akhir

1	2	5	2	1	4
3	12	6	15	1	5
3	7	8	22	5	3
5	7	14	1	3	5
5	1	1	5	5	1
5	5	4	3	3	1

Kemudian untuk max pooling 2x2 dengan stride 3|karena $(1164086 \bmod 3 + 1 = 3)$ adalah

1	2	5	2	1	4
3	12	6	15	1	5
3	7	8	22	5	3
5	7	14	1	3	5
5	1	1	5	5	1
5	5	4	3	3	1

Menjadi

12	15	5
7	22	5
5	5	5

7/1164086/Teori/chapter7tasya13.png

Gambar 7.12 Algoritma Konvolusi Tasya

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

7/1164086/Praktek/chapter7tasya14.png

Gambar 7.13 Kode Program Blok In 1 Tasya

```
4         # neuron activation functions behave best when input
         values are between 0.0 and 1.0 (or -1.0 and 1.0),
5         # so we rescale each pixel value to be in the range 0.0
         to 1.0 instead of 0-255
6         img /= 255.0
7         imgs.append((row[0], row[2], img))
8         classes.append(row[2])
9         i += 1
```

Keterangannya sebagai berikut :

- variabel imgs berisikan array kosong
- Variabel classes berisikan array kosong
- Membuka file csv dari Folder HSYv2 dengan nama file hasy-data-labels.csv sebagai csvfile

- Variabel csvreader akan menggunakan fungsi reader pada library csv untuk membaca file csv tadi yang disimpan di csvfile.
- Dimana variabel i dimuali dari nol.
- Untuk setiap baris pada csvreader
- Jika i lebih besar dari 0
- Jadi itu akan mengambil contoh Gambar PIL dan mengubahnya menjadi array numpy dengan mengambil data dari HSYv2 dan dimulai dari baris ke nol.
- Hasil dari variabel img akan dibagi dengan 255.0
- .append akan membuat list array baru untuk baris 0 baris 2 pada img.
- Menyimpan setiap class nya pada baris 2
- Penambahan i sebanyak 1.
- Hasilnya seperti berikut :

```
In [2]: imgs = []
...: classes = []
...: with open('HASYv2/hasy-data-labels.csv') as csvfile:
...:     csvreader = csv.reader(csvfile)
...:     i = 0
...:     for row in csvreader:
...:         if i > 0:
...:             img =
keras.preprocessing.image.img_to_array(pil_image.open("HASYv2/" + row[0]))
...:         # neuron activation functions behave best when input values
between 0.0 and 1.0 (or -1.0 and 1.0),
...:         # so we rescale each pixel value to be in the range 0.0 to 1.
instead of 0-255
...:         img /= 255.0
...:         imgs.append((row[0], row[2], img))
...:         classes.append(row[2])
...:         i += 1
```

7/1164086/Praktek/chapter7tasya15.png

Gambar 7.14 Kode Program Blok In 2 Tasya

Keterangannya sebagai berikut :

- Impor librari Random dari Python
- Melakukan pengacakan untuk imgs dengan Metode Shuffle untuk mengocok urutan di tempat. yaitu, mengubah posisi item dalam daftar.
- Membagi data dari imgs dengan cara mengalikan 80% dengan jumlah data dari imgs.
- Untuk data train mengambil hasil dari perhitungan sebelumnya.
- Untuk data test mengambil sisa dari jumlah yang telah dijadikan data train
- Hasilnya seperti berikut :

7/1164086/Praktek/chapter7tasya16.png

```
In [3]: import random
...: random.shuffle(imgs)
...: split_idx = int(0.8*len(imgs))
...: train = imgs[:split_idx]
...: test = imgs[split_idx:]
```

Gambar 7.15 Kode Program Blok In 3 Tasya

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

Keterangannya sebagai berikut :

- Impor librari Numpy yang di inisiasikan sebagai np
- Variabel train_input mengubah input menjadi sebuah array yang diambil dari baris 2, data train.
- Variabel test_input mengubah input menjadi sebuah array yang diambil dari baris 2, data test.
- Variabel train_output mengubah input menjadi sebuah array yang diambil dari baris 1, data train.
- Variabel train_output mengubah input menjadi sebuah array yang diambil dari baris 1, data test.
- Hasilnya seperti berikut

```
In [4]: import numpy as np
...:
...: train_input = np.asarray(list(map(lambda row: row[2], train)))
...: test_input = np.asarray(list(map(lambda row: row[2], test)))
...:
...: train_output = np.asarray(list(map(lambda row: row[1], train)))
...: test_output = np.asarray(list(map(lambda row: row[1], test)))
```

7/1164086/Praktek/chapter7tasya17.png

Gambar 7.16 Kode Program Blok In 4 Tasya

Keterangannya sebagai berikut :

- Impor Fungsi LabelEncoder
- Impor Fungsi OneHotEncoder
- Berikut hasilnya :

7/1164086/Praktek/chapter7tasya18.png

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

Gambar 7.17 Kode Program Blok In 5 Tasya

Keterangannya sebagai berikut :

- Variabel `label_encoder` akan memanggil fungsi `LabelEncoder` tadi.
- variabel `integer_encoded` akan menggunakan `labelencoder` untuk melakukan fit pada `classes` agar berubah datanya menjadi integer.
- Berikut hasilnya :

7/1164086/Praktek/chapter7tasya19.png

```
In [6]: label_encoder = LabelEncoder()
...: integer_encoded = label_encoder.fit_transform(classes)
```

Gambar 7.18 Kode Program Blok In 6 Tasya

Keterangannya sebagai 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`.
- Berikut hasilnya :

7/1164086/Praktek/chapter7tasya20.png

```
In [7]: onehot_encoder = OneHotEncoder(sparse=False)
...: integer_encoded = integer_encoded.reshape(len(integer_encoded), 1)
...: onehot_encoder.fit(integer_encoded)
C:\Users\DidinIrfandi\Anaconda3\lib\site-packages\sklearn\preprocessing\_encoders.py:371: FutureWarning: The handling of integer data will change in version 0.22. Currently the categories are determined based on the range [0, max(values)], while in the future they will be determined based on the unique values.
If you want the future behaviour and silence this warning, you can specify "categories='auto'".
In case you used a LabelEncoder before this OneHotEncoder to convert the categories into integers, then you can now use the OneHotEncoder directly.
warnings.warn(msg, FutureWarning)
Out[7]:
OneHotEncoder(categorical_features=None, categories=None,
dtype=<class 'numpy.float64'>, handle_unknown='error',
n_values=None, sparse=False)
```

Gambar 7.19 Kode Program Blok In 7 Tasya

```
1 print("Number of classes: %d" % num_classes)
```

Keterangannya sebagai 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" dimana mengembalikan nilai integer dari `num_classes`.
- Hasilnya sebagai berikut :

```
In [8]: train_output_int = label_encoder.transform(train_output)
...: train_output =
onehot_encoder.transform(train_output_int.reshape(len(train_output_int), 1))
...: test_output_int = label_encoder.transform(test_output)
...: test_output =
onehot_encoder.transform(test_output_int.reshape(len(test_output_int), 1))
...:
...: num_classes = len(label_encoder.classes_)
...: print("Number of classes: %d" % num_classes)
Number of classes: 369
```

7/1164086/Praktek/chapter7tasya21.png

Gambar 7.20 Kode Program Blok In 8 Tasya

Keterangannya sebagai 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.
- Hasilnya seperti berikut :

7/1164086/Praktek/chapter7tasya22.png

```
In [9]: from keras.models import Sequential
...: from keras.layers import Dense, Dropout, Flatten
...: from keras.layers import Conv2D, MaxPooling2D
```

Gambar 7.21 Kode Program Blok In 9 Tasya

```
1 model.add(Flatten())
2 model.add(Dense(1024, activation='tanh'))
3 model.add(Dropout(0.5))
4 model.add(Dense(num_classes, activation='softmax'))
5
6 model.compile(loss='categorical_crossentropy', optimizer='adam',
7               metrics=['accuracy'])
8
9 print(model.summary())
```

Keterangannya sebagai berikut :

- Melakukan pemodelan Sequential.
- Menambahkan Konvolusi 2D dengan 32 filter konvolusi masing-masing berukuran 3x3 dengan algoritma 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 algoritma 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.
- Hasilnya sebagai berikut :

Layer (type)	Output Shape	Param #
conv2d_1 (Conv2D)	(None, 30, 30, 32)	896
max_pooling2d_1 (MaxPooling2)	(None, 15, 15, 32)	0
conv2d_2 (Conv2D)	(None, 13, 13, 32)	9248
max_pooling2d_2 (MaxPooling2)	(None, 6, 6, 32)	0
flatten_1 (Flatten)	(None, 1152)	0
dense_1 (Dense)	(None, 1024)	1180672
dropout_1 (Dropout)	(None, 1024)	0
dense_2 (Dense)	(None, 369)	378225
Total params: 1,569,041		
Trainable params: 1,569,041		
Non-trainable params: 0		
None		

7/1164086/Praktek/chapter7tasya23.png

Gambar 7.22 Kode Program Blok In 10 Tasya

Keterangannya sebagai 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.
- Hasilnya sebagai berikut :

```
In [11]: import keras.callbacks
...: tensorboard = keras.callbacks.TensorBoard(log_dir='./logs/mnist-style')
```

7/1164086/Praktek/chapter7tasya24.png

Gambar 7.23 Kode Program Blok In 11 Tasya

```
1
2 score = model.evaluate(test_input, test_output, verbose=2)
3 print('Test loss:', score[0])
4 print('Test accuracy:', score[1])
```

Keterangannya sebagai 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 split 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 lost dan data accuracy dari test
- Menampilkan data loss dengan menghitung jumlah kemunculan nol .
- Menampilkan data accuracy dengan menghitung jumlah kemunculan 1.
- Berikut hasilnya :

```

Train on 107668 samples, validate on 26918 samples
Epoch 1/10
- 240s - loss: 1.5635 - acc: 0.6235 - val_loss: 1.0161 - val_acc: 0.7296
Epoch 2/10
- 232s - loss: 0.9918 - acc: 0.7267 - val_loss: 0.9090 - val_acc: 0.7448
Epoch 3/10
- 231s - loss: 0.8709 - acc: 0.7520 - val_loss: 0.8584 - val_acc: 0.7662
Epoch 4/10
- 244s - loss: 0.8009 - acc: 0.7660 - val_loss: 0.8609 - val_acc: 0.7572
Epoch 5/10
- 241s - loss: 0.7482 - acc: 0.7770 - val_loss: 0.8634 - val_acc: 0.7593
Epoch 6/10
- 242s - loss: 0.7076 - acc: 0.7847 - val_loss: 0.8703 - val_acc: 0.7624
Epoch 7/10
- 235s - loss: 0.6778 - acc: 0.7917 - val_loss: 0.8623 - val_acc: 0.7653
Epoch 8/10
- 232s - loss: 0.6474 - acc: 0.7988 - val_loss: 0.8708 - val_acc: 0.7636
Epoch 9/10
- 233s - loss: 0.6232 - acc: 0.8043 - val_loss: 0.8925 - val_acc: 0.7608
Epoch 10/10
- 234s - loss: 0.6033 - acc: 0.8077 - val_loss: 0.8951 - val_acc: 0.7570
Test loss: 0.8862240902399043
Test accuracy: 0.7599191607031478

```

7/1164086/Praktek/chapter7tasya25.png

Gambar 7.24 Kode Program Blok In 12 Tasya

```

1         for dropout in [0.0, 0.25, 0.50, 0.75]:
2             model = Sequential()
3             for i in range(conv2d_count):
4                 if i == 0:
5                     model.add(Conv2D(32, kernel_size=(3, 3),
6 activation='relu', input_shape=np.shape(train_input[0])))
7                 else:
8                     model.add(Conv2D(32, kernel_size=(3, 3),
9 activation='relu'))
10                    model.add(MaxPooling2D(pool_size=(2, 2)))
11                    model.add(Flatten())
12                    model.add(Dense(dense_size, activation='tanh'))
13                    if dropout > 0.0:
14                        model.add(Dropout(dropout))
15                    model.add(Dense(num_classes, activation='softmax'))

```

Keterangannya sebagai 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
- Mendefinisikan 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
- Hasilnya sebagai berikut :

```

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: 1.13 sec
Conv2D count: 1, Dense size: 128, Dropout: 0.25 - Loss: 0.91, Accuracy: 0.76, Time: 1.13 sec
Conv2D count: 1, Dense size: 128, Dropout: 0.50 - Loss: 0.80, Accuracy: 0.78, Time: 1.13 sec
Conv2D count: 1, Dense size: 128, Dropout: 0.75 - Loss: 0.80, Accuracy: 0.78, Time: 1.13 sec
Conv2D count: 1, Dense size: 256, Dropout: 0.00 - Loss: 1.28, Accuracy: 0.74, Time: 1.13 sec
Conv2D count: 1, Dense size: 256, Dropout: 0.25 - Loss: 1.08, Accuracy: 0.76, Time: 1.13 sec
Conv2D count: 1, Dense size: 256, Dropout: 0.50 - Loss: 0.91, Accuracy: 0.78, Time: 1.13 sec
Conv2D count: 1, Dense size: 256, Dropout: 0.75 - Loss: 0.78, Accuracy: 0.78, Time: 1.13 sec

```

7/1164086/Praktek/chapter7tasya26.png

Gambar 7.25 Kode Program Blok In 13 Tasya

```

1 model.add(Dense(128, activation='tanh'))
2 model.add(Dropout(0.5))
3 model.add(Dense(num_classes, activation='softmax'))
4 model.compile(loss='categorical_crossentropy', optimizer='adam',
5               metrics=['accuracy'])
6 print(model.summary())

```

Keterangannya sebagai 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 Convulasi lagi dengan kriteria yang sama tanpa menambahkan input, ini dilakukan untuk mendapatkan data yang terbaik
- Flatten digubakan ntuk meratakan inputan
- Menambahkan dense input sebanyak 128 neuron dengan menggunakan func-tion 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
- Gambarnya seperti berikut :

```
.... |> plot_model(summary=y\|)
```

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		

7/1164086/Praktek/chapter7tasya27.png

Gambar 7.26 Kode Program Blok In 14 Tasya

Keterangannya sebagai berikut :

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

```
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>
```

7/1164086/Praktek/chapter7tasya29.png

Gambar 7.27 Kode Program Blok In 15 Tasya

Keterangannya sebagai berikut :

- Menyimpan atau save model yang telah di latih dengan nama `mathsymbols.model`
- Hasilnya seperti berikut :

7/1164086/Praktek/chapter7tasya30.png

In [16]: model.save("mathsymbols.model")

mathsymbols.model

13/04/2019 03:45

MODEL File

2.443 KB

Gambar 7.28 Kode Program Blok In 16 Tasya

Keterangannya sebagai berikut :

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

7/1164086/Praktek/chapter7tasya31.png

In [17]: np.save('classes.npy', label_encoder.classes_)

classes.npy

13/04/2019 03:45

NPY File

28 KB

Gambar 7.29 Kode Program Blok In 17 Tasya

Keterangannya sebagai berikut :

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

7/1164086/Praktek/chapter7tasya32.png

```
... model2 = keras.models.load_model('mathsymbols.model')
... 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

Gambar 7.30 Kode Program Blok In 18 Tasya

```

1
2 # do the prediction
3 prediction = model2.predict(newimg.reshape(1, 32, 32, 3))
4
5 # figure out which output neuron had the highest score, and
6 # reverse the one-hot encoding
7 inverted = label_encoder2.inverse_transform([np.argmax(prediction)]) # argmax finds highest-scoring output
print("Prediction: %s, confidence: %.2f" % (inverted[0], np.max(prediction)))

```

Keterangannya sebagai 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
- Hasilnya sebagai 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_p
...:     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
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)))

```

7/1164086/Praktek/chapter7tasya33.png

Gambar 7.31 Kode Program Blok In 19 Tasya

```

1
2 # do the prediction
3 prediction = model2.predict(newimg.reshape(1, 32, 32, 3))
4
5 # figure out which output neuron had the highest score, and
6 # reverse the one-hot encoding
7 inverted = label_encoder2.inverse_transform([np.argmax(prediction)]) # argmax finds highest-scoring output
print("Prediction: %s, confidence: %.2f" % (inverted[0], np.max(prediction)))

```

Keterangannya sebagai berikut :

- 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
- Hasilnya sebagai berikut :

```
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
```

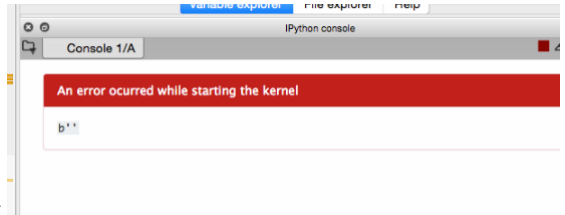
7/1164086/Praktek/chapter7tasya34.png

Gambar 7.32 Kode Program Blok In 20 Tasya

7.1.8 Penanganan Error

7.1.8.1 Error Starting Kernel

- Berikut merupakan screenshot error

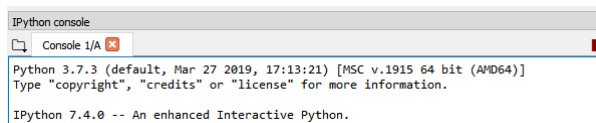


7/1164086/Praktek/chapter7error1.png

Gambar 7.33 Error Tasya

- Error tersebut merupakan error yang terjadi dan membuat kita tidak dapat mengakses dan menggunakan kernel atau konsol pada spyder.
- Untuk penanganannya sebagai berikut :
 1. Tutup spyder yang sedang dijalankan
 2. Kemudian buka kembali spyder
 3. Atau jika tidak berhasil, buka anaconda prompt dan ketikan "conda update spyder"
 4. Jika tidak berhasil juga bisa menginstall ulang anaconda
 5. Maka ketika dijaalakan lagi hasilnya seperti berikut :

7/1164086/Praktek/chapter7error2.png



Gambar 7.34 Penanganan Error Kernel Tasya

DAFTAR PUSTAKA

1. R. Awangga, "Sampeu: Servicing web map tile service over web map service to increase computation performance," in *IOP Conference Series: Earth and Environmental Science*, vol. 145, no. 1. IOP Publishing, 2018, p. 012057.

Index

disruptif, xxxix
modern, xxxix