

Modul Praktikum

Kecerdasan Buatan



Rolly Maulana Awangga

0410118609

Applied Bachelor of Informatics Engineering

Program Studi D4 Teknik Informatika

Applied Bachelor Program of Informatics Engineering

Politeknik Pos Indonesia

Bandung 2019

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

Acknowledgements

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

Abstract

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

Contents

List of Figures

Chapter 1

Mengenal Kecerdasan Buatan dan ScikitLearn

1.1 Teori

1.1.1 Kecerdasan Buatan

/BAGIAN TASYA WIENDHYRA 1164086 RESUME 1

Definisi

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.

Sejarah Singkat

Pada awal 50-an, studi tentang "mesin bepikir" 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 McCullough telah bekerja secara independen di bidang cybernetics, matematika, algoritma dan teori jaringan. Namun, seorang ilmuwan komputer dan kognitif John McCarthy adalah orang yang datang dengan ide untuk bergabung dengan upaya penelitian terpisah ini ke dalam satu bidang yang akan mempelajari topik 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 itu, Kecerdasan Buatan telah hidup melalui dekade kemuliaan dan cemoohan, yang dikenal luas sebagai musim panas dan musim dingin AI. Musim panasnya ditandai dengan optimisme dan dana besar, sedangkan musim dinginnya dihadapkan dengan pemotongan dana, ketidakpercayaan dan pesimisme.

Perkembangan Kecerdasan Buatan

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 membawa 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 di antaranya: 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 apa pun. Ross Quillian menunjukkan jaring semantik, sedangkan Jaime Carbonell (Sr.) mengembangkan Cendekia - program interaktif untuk instruksi yang dibantu komputer berdasarkan jaring semantik. Edward Feigenbaum dan Julian Feldman menerbitkan Computers and Thought, kumpulan artikel pertama tentang AI.

1.1.2 Definisi

RESUME 2

Supervised Learning

Supervised Learning adalah tugas pengumpulan data untuk menyimpulkan fungsi dari data pelatihan berlabel. Data pelatihan terdiri dari serangkaian contoh pelatihan. Dalam supervised learningi, 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 Learningi menyediakan algoritma pembelajaran dengan jumlah yang diketahui untuk mendukung penilaian di masa 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, di mana 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.

Regresi

Regresi adalah membahas masalah ketika variabel output adalah nilai riil atau berkelanjutan, seperti ”gaji” atau ”berat”. Banyak model yang berbeda dapat digu-

nakan, yang paling sederhana adalah regresi linier. Ia mencoba untuk menyesuaikan data dengan hyper-plane terbaik yang melewati poin.

Klasifikasi

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 kelamin 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.

Unsupervised Learning

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. Sistem AI yang mampu pembelajaran tanpa pengawasan sering dikaitkan dengan model pembelajaran generatif, meskipun mereka juga dapat menggunakan pendekatan berbasis pengambilan (yang paling sering dikaitkan dengan pembelajaran yang diawasi). Chatbots, mobil yang bisa mengemudi sendiri, program pengenalan wajah, sistem pakar dan robot adalah beberapa sistem yang dapat menggunakan pendekatan pembelajaran yang diawasi atau tidak terawasi. Dalam Unsupervised Learning, sistem AI disajikan dengan data yang tidak berlabel, tidak terkategorisasi 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 Learningi 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.

Data Set

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, 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.

Training Set

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.

Test Set

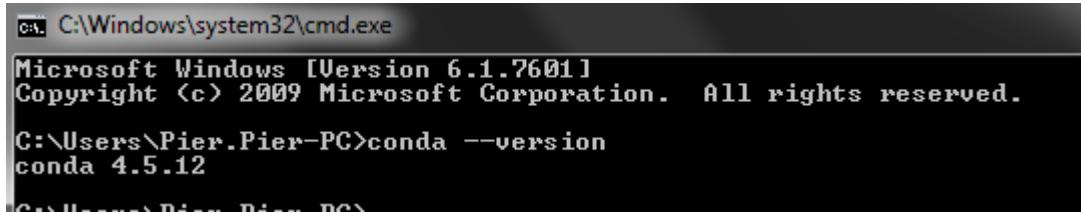
Ini berfungsi sebagai meterai persetujuan, dan Anda tidak menggunakan sampai akhir. Setelah Anda melatih dan mengoptimalkan data Anda, Anda menguji jaringan saraf Anda terhadap pengambilan sampel acak akhir ini. Hasil yang dihasilkannya harus memvalidasi bahwa jaring Anda secara akurat mengenali gambar, atau mengenalinya 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. / AKHIR BAGIAN TEORI TASYA WIENDHYRA 1164086

1.2 Instalasi

/BAGIAN TASYA WIENDHYRA 1164086 Membuka <https://scikit-learn.org/stable/tutorial/basic.html>
Dengan menggunakan bahasa yang mudah dimengerti dan bebas plagiat. Dan wajib skrinsut dari komputer sendiri.

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

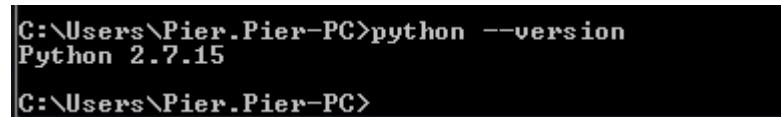
1. Pastikan bahwa sudah terinstal Anaconda pada PC anda, caranya buka CMD lalu ketikan "conda -version" jika hasilnya seperti



```
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
```

Figure 1.1: Versi Anaconda Yang Digunakan



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

Figure 1.2: Versi Python Yang Digunakan

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.
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.

1.2.2 Mencoba Loading an example dataset, menjelaskan makna dari tulisan tersebut dan mengartikan per baris

1. Mengimimport 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.

```

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
scikit-learn-0.20.2 | py27hf381715_0   5.1 MB
ca-certificates-2019.1.23 | 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

```

Figure 1.3: Instalasi Scikit Dari Anaconda

```

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

lt.xlim(min_x - .5 * max_x, max_x + .5 * max_x)
lt.ylim(min_y - .5 * max_y, max_y + .5 * max_y)
f 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,

```

Figure 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.

```

Figure 1.5: Hasil Yang Muncul Di CMD

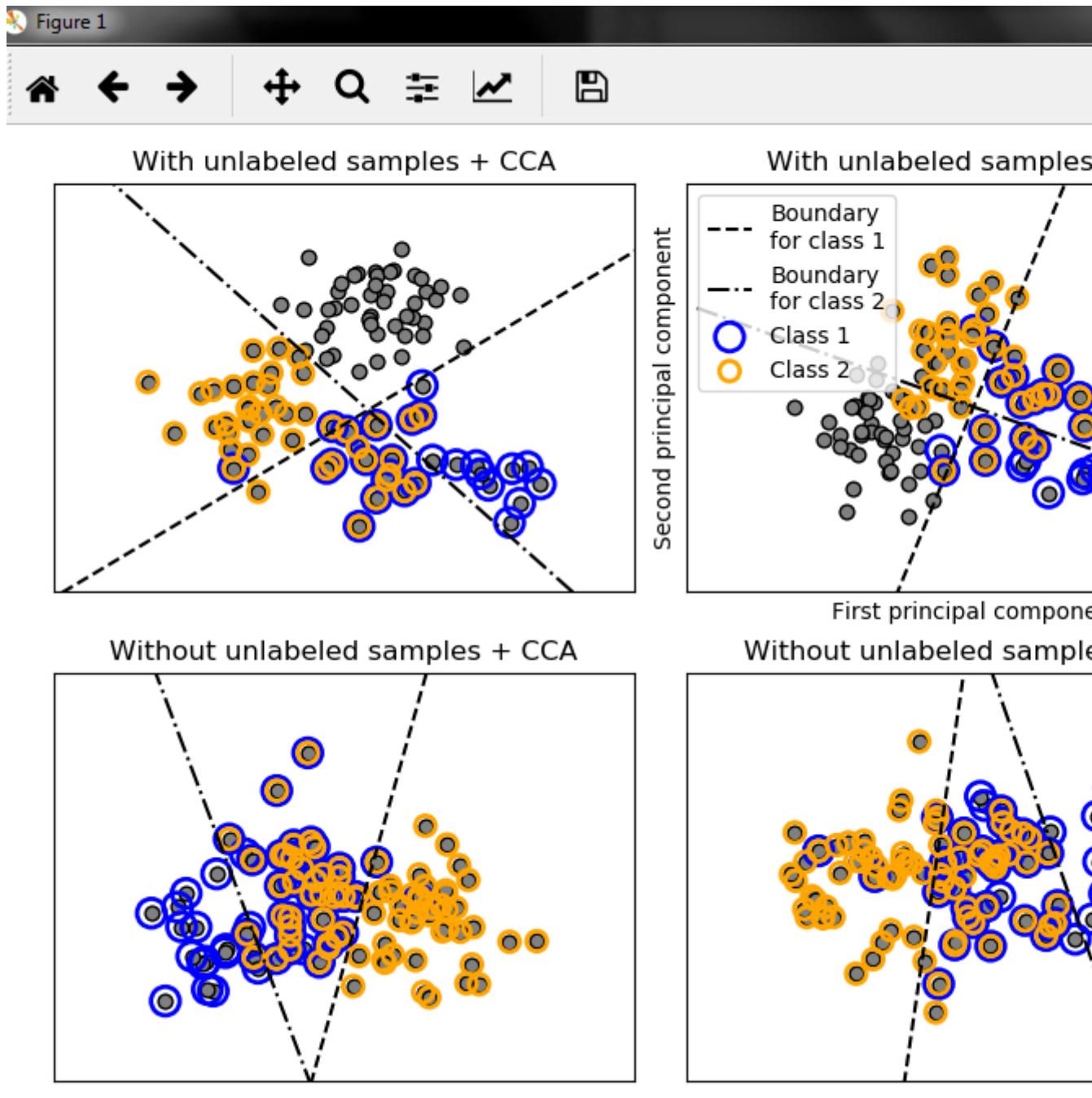


Figure 1.6: Gambar Yang Muncul Dari Matplotlib

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

Figure 1.7: Penjelasan

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

Figure 1.8: Penjelasan 2

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

Figure 1.9: Penjelasan 3

1.3 Teori/Annisa Fathoroni-1164067

1.3.1 Mencoba Learning and predicting, menjelaskan mak-sud dari tulisan tersebut dan mengartikan per baris

HARI 2

1. Pada website sebelumnya, cari Learning And Predicting dan ikuti langkah - langkahnya.
2. Buka Python Shell, atau dengan membuka Command Prompt di PC dan mengetikan python. maka akan masuk ke Python Shellnya.
3. Pada python shell ketikan "from sklearn import svm" yang dimana artinya akan memanggil dan menggunakan estimator dari kelas sklearn.svm.SVC
4. Kemudian, kita definisikan clf sebagai classifier, disini gamma didefinisikan secara manual
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,

```
>>> 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.]])
```

Figure 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 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>>
```

Figure 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 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> from sklearn import svm
>>>
```

Figure 1.12: Menggunakan Estimator Sklearn

yang dicadangan untuk prediksi. Pada skrip dibawah memilih training set dengan sintaks Python [: -1], yang menghasilkan array baru yang berisi semua kecuali item terakhir dari digits.data

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

/AKHIR BAGIAN INSTALASI TASYA WIENDHYRA 1164086

1.3.2 Mencoba Model Persistance, menjelaskan maksud dari tulisan tersebut dan mengartikan per baris

/BAGIAN TASYA WIENDHYRA 1164086

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.

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

Figure 1.13: Mendefinisikan Classifier

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

Figure 1.14: Memanggil Classifier Tanpa Baris Terakhir

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

Figure 1.15: Memprediksi Nilai Baru

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 ini

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

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

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

Figure 1.16: Hasil Pengujian Classifier

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

Figure 1.17: Hasil Pengujian Classifier

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

Figure 1.18: Pickle Pada Python

Pada gambar diatas 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 ketikan ”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

1.3.3 Mencoba Conventions, menjelaskan maksud dari tulisan tersebut dan mengartikan per baris

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

Pada gambar diatas dapat dijelaskan bahwa :

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

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

Figure 1.19: Pengujian Classifier Pickle

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

Figure 1.20: Penggunaan Joblib

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

Figure 1.21: Deklarasi Numpy

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

5. Refitting dan Memperbaharui Parameter

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

clf.set_params(kernel='linear').fit(X, y) mengubah kernel default menjadi linear

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

Figure 1.22: Contoh Type Casting

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

Figure 1.23: Menggunakan FitTransform

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

Figure 1.24: Regresi Yang Dilempar

clf.set_params(kernel='rbf', gamma='scale').fit(X, y) Di sini, kernel default rbf pertama kali diubah menjadi linear melalui SVC.set_params () setelah estimator dibuat, dan diubah kembali ke rbf untuk mereparasi estimator dan membuat prediksi kedua.

6. MultiClass VS MultiLabel Classifier

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

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

7. Di sini, classifier cocok () pada representasi label biner 2d dari y, menggunakan LabelBinarizer. Dalam hal ini predict () mengembalikan array 2d yang mewakili prediksi multilabel yang sesuai.

```

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

```

Figure 1.25: Refitting dan Memperbaharui Parameter

```

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

```

Figure 1.26: MultiClass Classifier

```

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

```

Figure 1.27: 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)
>>> classif.fit(X, y).predict(X)
array([[1, 1, 0, 0, 0],
       [1, 0, 1, 0, 0],
       [0, 1, 0, 1, 0],
       [1, 0, 1, 0, 0],
       [1, 0, 1, 0, 0]])
```

Figure 1.28: MultiLabel Classifier

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

Figure 1.29: Eror Import

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.

/AKHIR BAGIAN PEGUJIAN TASYA WIENDHYRA 1164086

1.4 Penanganan Error

/BAGIAN TASYA WIENDHYRA 1164086 HARI KEDUA

1. Berikut ini merupakan eror yang ditemui pada saat melakukan percobaan skrip.
2. Pada gambar eror diatas, kode erornya adalah ”ImportError: No Module Named” artinya mengalami masalah saat mengimpor modul yang ditentukan.
3. Solusinya bisa dilakukan seperti berikut :
eror diats terjadi dikarenakan Library Joblib belum terinstal pada PC. Maka dari itu sekarang kita harus menginstalnya dulu.
4. Buka CMD, kemudian ketikan ”pip install joblib” tunggu sampai instalasi berhasil seperti gambar berikut.

```
C:\Users\Pier.Pier-PC>pip install joblib
Collecting joblib
  Downloading https://files.pythonhosted.org/packages/cd/c1/50a758e8247561e58cb8
7305b1e90b171b8c767b15b12a1734001f41d356/joblib-0.13.2-py3-none-any.whl (278
kB)
    100% #####: 286kB 799kB/s
Installing collected packages: joblib
Successfully installed joblib-0.13.2
```

Figure 1.30: Instal Library Joblib

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

Figure 1.31: Berhasil Import Library Joblib

5. Apabila sudah terinstall, dapat dilakukan lagi import library joblib, maka akan berhasil seperti dibawah berikut

/AKHIR BAGIAN PENANGANAN EROR TASYA WIENDHYRA 1164086

1.5 Teori

Teori mencakup resume dari beberapa pembahasan. yaitu :

1.5.1 Kecerdasan Buatan

- Definisi Kecerdasan Buatan.

Jadi yang dimaksud dengan kecerdasan buatan yaitu salah satu dari cabang Ilmu pengetahuan yang kita punya dan berhubungan juga dengan pemanfaatan mesin yaitu untuk dapat memecahkan suatu persoalan yang rumit dengan menggunakan cara yang lebih manusiawi.

Jadi yang dimaksud dengan supervised learning adalah pembelajaran mesin yang diawasi menciptakan model yang melancarkan prediksi berdasarkan bukti adanya ketidakpastian. Algoritma pembelajaran yang diawasi memerlukan seperangkat data masukan dan tanggapan yang diketahui terhadap data (output) dan melatih model untuk menghasilkan prediksi yang masuk akal untuk respon terhadap data baru. Sedangkan untuk klasifikasi yaitu nilai output yang

bernilai diskrit (kelas) dan Bertujuan mengklasifikasi data baru dengan akurat diawasi dengan menggunakan teknik klasifikasi dan regresi untuk mengembangkan model prediktif. Yang dimaksud dengan regresi adalah nilai output yang bernilai kontinu (riil), Bertujuan memprediksi output dengan akurat untuk data baru.

Unsupervised learning tidak menggunakan data latih atau data training untuk melakukan prediksi maupun klasifikasi. Berdasarkan model matematisnya, algoritma ini tidak memiliki target variabel. Salah satu tujuan dari algoritma ini adalah mengelompokkan objek yang hampir sama dalam suatu area tertentu. Dan yang dimaksud dengan Dataset adalah objek yang merepresentasikan data dan relasinya di memory. Strukturnya mirip dengan data di database. Dataset berisi koleksi dari datatable dan datarelation. Sedangkan testing set digunakan untuk mengukur sejauh mana classifier berhasil melakukan klasifikasi dengan benar. Dan training set digunakan oleh algoritma klassifikasi untuk membentuk sebuah model classifier. Model yang dimaksud ini merupakan representasi pengetahuan yang akan digunakan untuk prediksi kelas data baru yang belum pernah ada.

Kecerdasan buatan (Artificial Intelligence) atau disingkat menjadi AI adalah suatu pengetahuan yang membuat suatu komputer dapat menirukan kecerdasan manusia sehingga diharapkan atau diinginkan komputer dapat melakukan hal-hal yang apabila dikerjakan manusia memerlukan kecerdasan. Misalnya melakukan penalaran untuk mencapai suatu kesimpulan atau melakukan translasi dari satu bahasa manusia ke bahasa manusia yang lain.

1.5.2 Perkembangan Kecerdasan Buatan



Figure 1.32: capturing

Ditemukannya pertama kali alat penyimpanan dan pemrosesan informasi yang disebut komputer elektronik. Tahun 1943, Warren McCulloch dan Walter Pitts berhasil membuat suatu model saraf tiruan di mana setiap neuron digambarkan sebagai ‘on’ dan ‘off’. Pada tahun 1950, Norbert Wiener membuat penelitian mengenai prinsip-prinsip teori feedback. Pada tahun 1956, John McCarthy meyakinkan Minsky,

Claude Shannon, dan Nathaniel Rochester untuk membantunya melakukan penelitian dalam bidang automata, jaringan saraf, dan pembelajaran intelijensia. Mereka mengerjakan proyek ini selama 2 bulan di Universitas Dartmouth. Hasilnya adalah program yang mampu berpikir non-numerik dan menyelesaikan masalah pemikiran, yang dinamakan Principia Mathematica. Hal ini menjadikan McCarthy disebut sebagai father of Artificial Intelligence/ Bapak Kecerdasan Buatan.

Pada tahun 1958, McCarthy di MIT AI Lab mendefinisikan bahasa pemrograman tingkat tinggi yaitu LISP, yang sekarang mendominasi pembuatan program-program AI. Kemudian, McCarthy membuat program yang dinamakan programs with common sense. Pada tahun 1959, Program komputer General Problem Solver berhasil dibuat oleh Herbert A. Simon, J.C. Shaw, dan Allen Newell. Program ini dirancang untuk memulai penyelesaian masalah secara manusiawi. Pada tahun yg sama Nathaniel Rochester dari IBM dan para mahasiswanya merilis program AI yaitu geometry theorem prover. Pada tahun 1963, program yang dibuat James Slagle mampu menyelesaikan masalah integral tertutup untuk mata kuliah Kalkulus. Pada tahun 1968, program analogi buatan Tom Evan menyelesaikan masalah analogi geometri yang ada pada tes IQ.

Perkembangan AI melambat disebabkan adanya beberapa kesulitan yang dihadapi seperti Program-program AI yang bermunculan hanya mengandung sedikit atau bahkan tidak mengandung sama sekali pengetahuan pada subjeknya, banyak terjadi kegagalan pada pembuatan program AI, terdapat beberapa batasan pada struktur dasar yang digunakan untuk menghasilkan perilaku intelijensia. Pada tahun 1960an, Ed Feigenbaum, Bruce Buchanan, dan Joshua Lederberg merintis proyek DENDRAL yaitu program untuk memecahkan masalah struktur molekul dari informasi yang didapatkan dari spectrometer massa. Pada tahun 1986, program ini telah berhasil menghemat 40 juta dolar per tahun. Pada tahun 1988, kelompok AI di DEC menjalankan 40 sistem pakar. Hampir semua perusahaan besar di USA mempunyai divisi Ai sendiri yang menggunakan ataupun mempelajari sistem pakar. Perusahaan yang sejak tahun 1982 hanya menghasilkan beberapa juta US dollar per tahun meningkat menjadi 2 miliar US dollar per tahun pada tahun 1988.

Meskipun bidang ilmu komputer menolak jaringan saraf tiruan setelah diterbitkannya buku ‘Perceptrons’ karangan Minsky dan Papert, tetapi para ilmuwan masih mempelajari bidang ilmu tersebut dari sudut pandang yang lain, yaitu fisika. Ahli fisika seperti Hopfield (1982) menggunakan teknik-teknik mekanika statistika untuk menganalisa sifat-sifat penyimpanan dan optimasi pada jaringan saraf. Para ahli psikolog, David Rumhelhart dan Geoff Hinton melanjutkan penelitian mengenai

model jaringan saraf pada memori. Pada tahun 1985-an sedikitnya empat kelompok riset menemukan algoritma Back-Propagation. Algoritma ini berhasil diimplementasikan ke dalam ilmu bidang komputer dan psikologi.

1.5.3 Supervised Learning, Klasifikasi dan Regres

- Supervised learning adalah sebuah pendekatan dimana sudah terdapat data yang dilatih, dan terdapat variable yang ditargetkan sehingga tujuan dari pendekatan ini adalah mengelompokan suatu data ke data yang sudah ada.
- Dalam klasifikasi, keluaran dari setiap data adalah bilangan bulat atau diskrit. Misalnya pengambilan keputusan untuk main sepak bola atau tidak maka keluaran bisa diubah kedalam bilangan bulat 1 (main bola), dan -1 (tidak main). Regresi, keluaran dari setiap data adalah bilangan kontinu. Misalnya peramalan harga rumah berdasarkan lokasi, umur rumah dan luas rumah, maka keluarannya berupa bilangan kontinu berupa bilangan Rp 120 juta, Rp 100 juta atau Rp 51 juta.

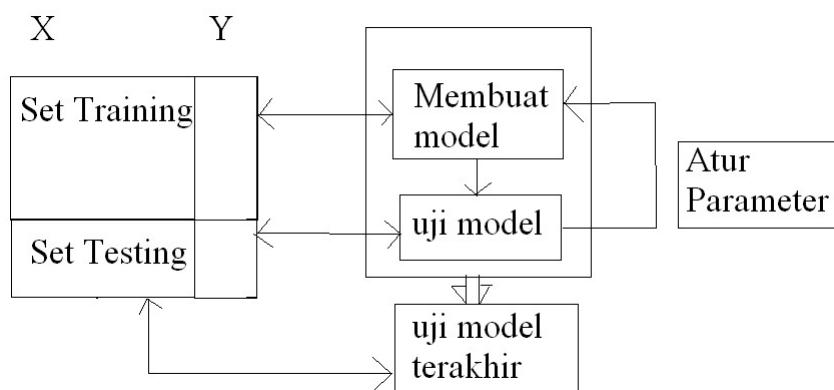


Figure 1.33: install anaconda 2

1.6 Instalasi/Annisa Fathoroni-1164067)

Untuk Instalasinya mencakup beberapa pembahasan dan tutorial. yaitu :

1. Instalasi Scikit-Learn Dari Anaconda

- Instalasi Anaconda
- Mencoba Learning and predicting, menjelaskan maksud dari tulisan tersebut dan mengartikan per baris[hari ke 2](10) Pada gambar diatas dapat

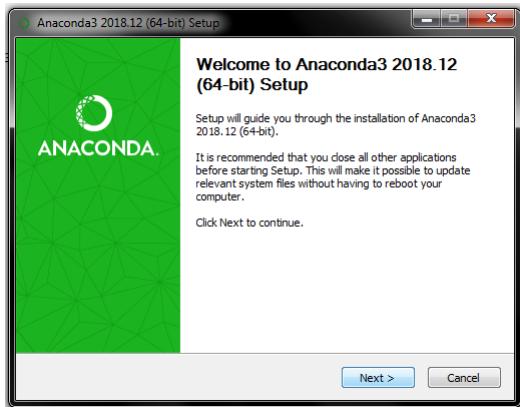


Figure 1.34: capturing

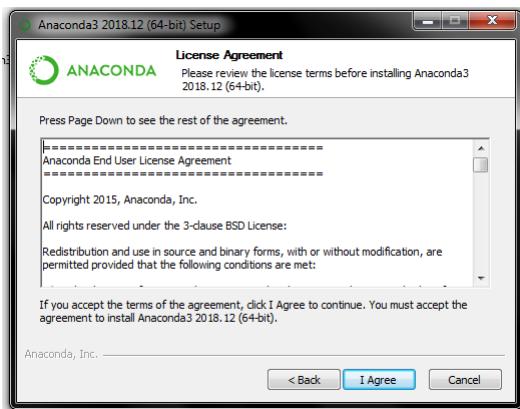


Figure 1.35: capturing

dijelaskan bahwa : Baris yang pertama kita memasukkan dan memanggil svm dari sklearn. Baris yang kedua kita membuat variable.

- Pada gambar dibawah dapat dijelaskan bahwa : Baris yang pertama itu clf dipasang pada model fit metode. Baris yang kedua kita implementasikan klasifikasi dukungan vektor.
- Pada gambar dibawah dapat dijelaskan bahwa : Baris yang pertama itu untuk prediksi nilai baru. Baris yang kedua untuk set array.
- Setelah proses download selesai, run package yang telah didownload lalu next.

Pilih 'I Agree' untuk menyetujui persyaratan dan peraturan mengenai aplikasi ini.

Lalu, pilih 'All Users' untuk dapat diakses oleh semua user, dan pilih "Just Me' hanya untuk dapat diakses oleh 1 user pc.

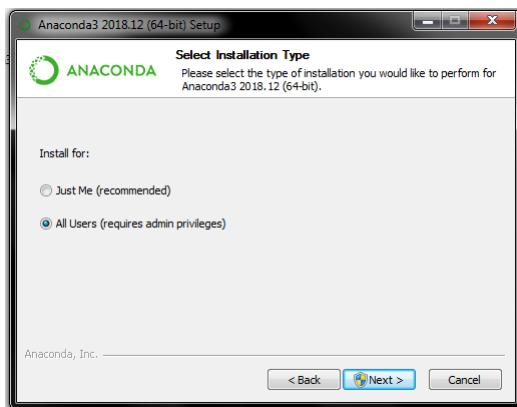


Figure 1.36: capturing

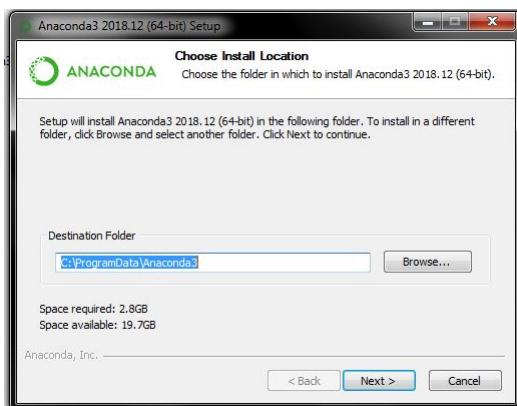


Figure 1.37: capturing

Kemudian, pilih penyimpanan package.

Ceklis pada bagian cek box pertama untuk otomatis pengaturan dan penambahan enviroment variabel pada PATH dan cek box kedua untuk register anaconda sistem.

Proses instalasi Lalu pada proses ini, skip untuk mempercepat proses penginstalan.

Proses instalasi selesai.

Setelah proses instalasi selesai, cek penginstalan conda dan python.

Lalu install scikit dengan perintah 'pip install -U scikit-learn' atau 'conda install scikit-learn'

- Loading an Example Dataset
- mencoba Model persistence, menjelaskan maksud dari tulisan tersebut dan mengartikan per baris[hari ke 2](10) Pada gambar diatas dapat dijelaskan bahwa :

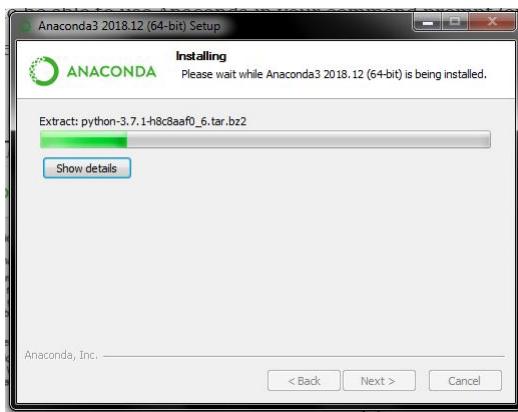


Figure 1.38: capturing

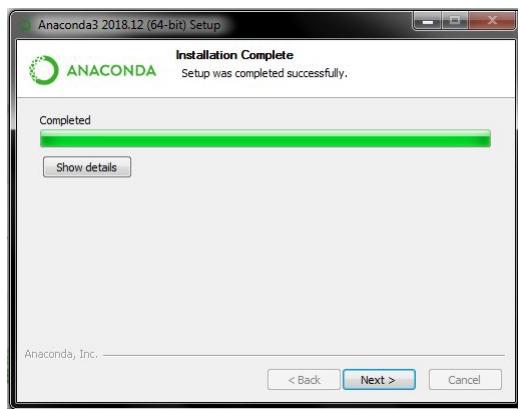


Figure 1.39: capturing

Baris yang pertama kita memasukkan dan memanggil datasets dari sklearn.
Baris yang kedua kita memasukkan dan memanggil svm dari sklearn.
Baris yang ketiga kita membuat variable clf.
Baris yang keempat kita membuat variable iris.
Baris yang kelima kita membuat variable x, y.
Baris yang keenam clf dipasang pada model fit metode.
Baris yang ke tujuh kita implementasikan klasifikasi dukungan vektor.
Baris yang kedelapan kita memanggil library pickle.
Baris yang kesembilan untuk membuat variable s.
Baris yang kesepuluh untuk membuat variable clf2.
Baris yang kesebelas untuk prediksi nilai baru.
Baris yang keduabelas untuk set array.

1. Mencoba Conventions, menjelaskan maksud dari tulisan tersebut dan mengartikan per baris[hari ke 2](10)

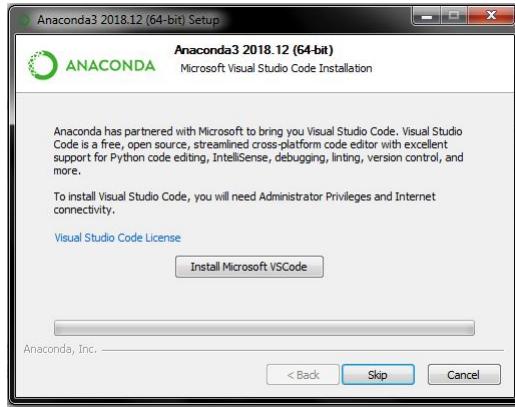


Figure 1.40: capturing

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

Figure 1.41: capturing

2. Baris 1: Memanggil data dari sklearn.
3. Baris 2: Terdapat variabel 'iris', dimana variabel iris memanggil datasets dan di dalamnya akan memproses digits.
4. Baris 3: Kemudian ada variabel lainnya 'digits' yang digunakan untuk memanggil dataset dan di dalamnya akan memproses digits.
5. Kemudian untuk perintah Print(digits.data) untuk menampilkan output dari hasil variabel digits dan akan berupa data.

- Mencoba Learning and Predicting.

Baris 1 = Memasukkan dan memanggil SVM dari sklearn.

Baris 2 = Membuat variabel.

Baris 1 = clf dipasang pada model fit metode.

Baris 2 = Mengimplementasikan klasifikasi dukungan vektor.

Baris 1 = Prediksi nilai baru.

Baris 2 = set array.

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

Figure 1.42: capturing

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

Figure 1.43: capturing

```
C:\Windows\system32>pip install -U scikit-learn
Collecting scikit-learn
  Using cached https://files.pythonhosted.org/packages/d1/6c/6dd21e203ff95d7988
  aee2105b4f6610a02483e0d4ac950f3630969c9/scikit_learn-0.20.2-cp37-cp37m-win_and_
  64.whl
Requirement already satisfied, skipping upgrade: numpy>=1.8.2 in c:\programdata\
  anaconda3\lib\site-packages (from scikit-learn) (1.15.4)
Requirement already satisfied, skipping upgrade: scipy>=0.13.3 in c:\programdata\
  anaconda3\lib\site-packages (from scikit-learn) (1.1.0)
Installing collected packages: scikit-learn
  Found existing installation: scikit-learn 0.20.1
    Uninstalling scikit-learn-0.20.1:
      Successfully uninstalled scikit-learn-0.20.1
  Successfully installed scikit-learn-0.20.2
C:\Windows\system32>
```

Figure 1.44: capturing

- Model Persistence

Baris 1 = Memasukkan dan memanggil datasets dari sklearn.

Baris 2 = Memasukkan dan memanggil SVM dari sklearn.

Baris 3 = Membuat variable clf.

Baris 4 = Membuat variable iris.

Baris 5 = Membuat variable x dan y.

Baris 6 = clf dipasang pada model fit metode.

Baris 7 = Mengimplementasikan klasifikasi dukungan vektor.

Baris 8 = Memanggil library pickle.

Baris 9 = Membuat variable s.

Baris 10 = Membuat variable clf2.

Baris 11 = Memprediksi nilai baru.

Baris 12 = Set array.

- Conventions

Baris 1 = Memasukkan dan memanggil numpy sebagai np.

Baris 2 = Memasukkan dan memanggil randomprojection dari sklearn.

Baris 3 = Membuat variable rng.

Baris 4 = Membuat variable rng.

Baris 5 = Membuat variable x.

Baris 6 = Proses pemanggilan variable x.

Baris 7 = Proses pemanggilan dtype.

```
C:\Windows\system32>python
Python 3.7.1 (default, Dec 10 2018, 22:54:23) [MSC v.1915 64 bit (AMD64)] :: Anaconda, Inc. on Windows-10-10.0.17134-SP0
Type "help", "copyright", "credits" or "license" for more information.
>>> print('Annisa Fathoroni')
Annisa Fathoroni
>>>
```

Figure 1.45: capturing

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

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

Figure 1.46: capturing

Baris 8 = Membuat variable tranformer.

Baris 9 = Membuat variable xnew.

Baris 10 = Proses pemanggilan xnew.

Baris 11 = Pemanggilan dtype.

1.7 Penanganan Error/Annisa Fathoroni-1164067)

Untuk penanganan error mencakup beberapa pembahasan dan tutorial. yaitu :

1. Screenshoot Error
2. Kode Error dan Jenis Errornya

SyntaxError: invalid syntax

```
import numpy as np
from sklearn import random_projection

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

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

Figure 1.47: capturing

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

Figure 1.48: capturing

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

Figure 1.49: capturing

3. Solusi Pemecahan Error. Pada gambar dibawah dapat dijelaskan bahwa :
 - Baris yang pertama kita masukkan dan memanggil numpy sebagai np.
 - Baris yang kedua kita masukkan dan memanggil randomprojection dari sklearn.
 - Baris yang ketiga kita membuat variable rng.
 - Baris yang keempat untuk membuat variable rng.
 - Baris yang kelima untuk membuat variable x.
 - Baris yang keenam untuk pemanggilan variable x.
 - Baris yang ketujuh untuk pemanggilan dtype.
 - Baris yang kedelapan kita membuat variable transformer.
 - Baris yang kesembilan kita membuat variable xnew.
 - Baris yang kesempulah untuk pemanggilan xnew.
 - Baris yang keseblas untuk pemanggilan dtype.

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

Figure 1.50: capturing

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

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

Figure 1.51: capturing

```
import numpy as np
from sklearn import random_projection

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

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

Figure 1.52: capturing

```
>>> print('Annisa Fathoroni')
      File "<stdin>", line 1
        print('Annisa Fathoroni')
SyntaxError: invalid syntax
```

Figure 1.53: capturing

```
>>> print('Annisa Fathoroni')
Annisa Fathoroni
>>> -
```

Figure 1.54: capturing

Chapter 2

Related Works

Your related works, and your purpose and contribution which.

2.1 Tasya Wiendhyra/1164086

2.1.1 binary classification dilengkapi ilustrasi gambar

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

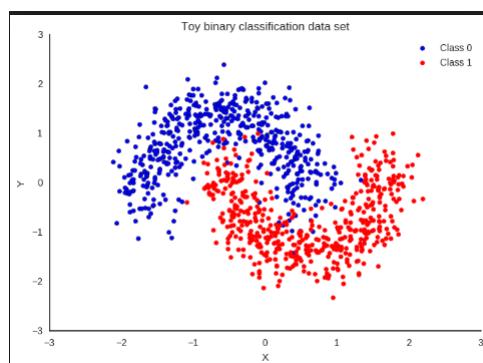


Figure 2.1: Binary Classification

2.1.2 supervised learning dan unsupervised learning dan clustering dengan ilustrasi gambar

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

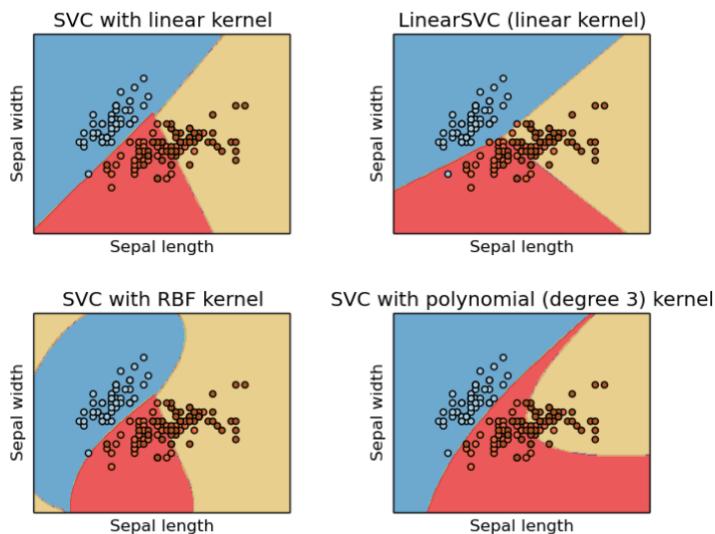


Figure 2.2: Supervised Learning

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

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

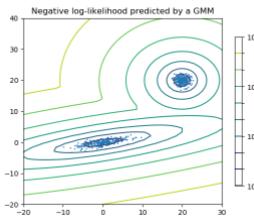


Figure 2.3: Unsupervised Learning

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

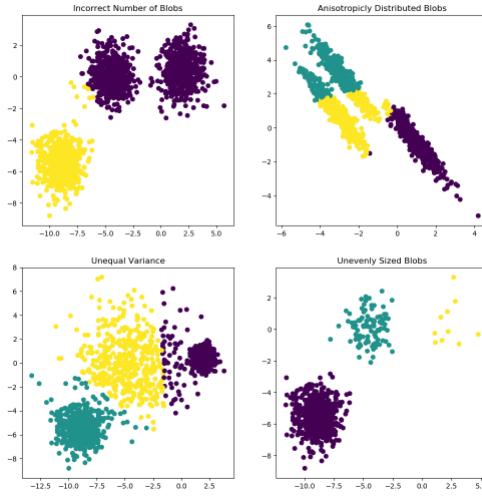


Figure 2.4: Cluster

2.1.3 evaluasi dan akurasi dari buku dan disertai ilustrasi contoh dengan gambar

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

```
>>> 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')
```

Figure 2.5: Evaluasi dan Akurasi

2.1.4 bagaimana cara membuat dan membaca confusion matrix, buat confusion matrix

1. Cara membuat dan membaca confusion matrix :

- 1) Tentukan pokok permasalahan dan atributanya, misal gaji dan listik.
- 2) Buat pohon keputusan

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

2.1.5 bagaimana K-fold cross validation bekerja dengan gambar ilustrasi

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

2.1.6 decision tree dengan gambar ilustrasi

1. Decision Tree adalah metode pembelajaran yang diawasi non-parametrik yang digunakan untuk klasifikasi dan regresi. Tujuannya adalah untuk membuat model yang memprediksi nilai variabel target dengan mempelajari aturan keputusan sederhana yang disimpulkan dari fitur data.
- Misalnya, dalam contoh di bawah ini, decision tree belajar dari data untuk

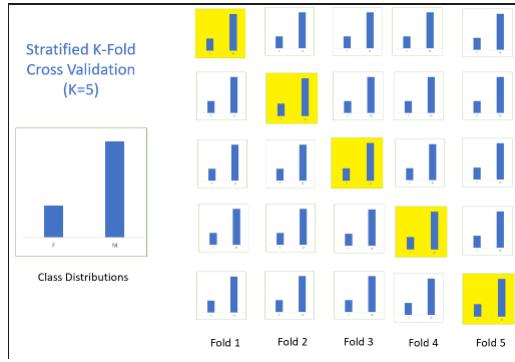


Figure 2.6: K-fold cross validation

memperkirakan kurva sinus dengan seperangkat aturan keputusan if-then-else. Semakin dalam pohon, semakin rumit aturan keputusan dan semakin bugar modelnya.

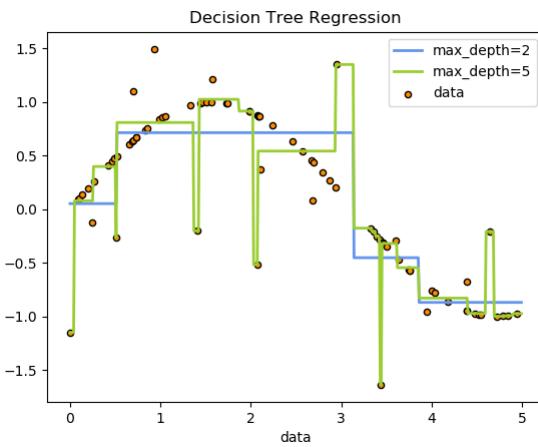


Figure 2.7: Decision Tree

2.1.7 Information Gain dan entropi dengan gambar ilustrasi

1. Information gain didasarkan pada penurunan entropi setelah dataset dibagi pada atribut. Membangun decision tree adalah semua tentang menemukan atribut yang mengembalikan perolehan informasi tertinggi (mis., Cabang yang paling homogen).
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

```

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

```

Figure 2.8: Information gain

dari sejumlah lemparan sulit diprediksi. Mengapa? Karena tidak ada hubungan antara membalik dan hasilnya. Inilah inti dari entropi.

```

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), 255, dtype=np.uint8)
noise_mask[32:-32, 32:-32] = 0

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()

```

Figure 2.9: Entropi

2.2 scikit-learn

HARI KEDUA TASYA WIENDHYRA 1164086

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
```

Figure 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>=30
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
```

Figure 2.11: Generate Binary Label

```
3. # use one-hot encoding on categorical columns
durian = pd.get_dummies(durian, columns=['sex', 'school', 'address',
'famsize',
'Pstatus', 'Mjob', 'Fjob',
'reason', 'guardian', 'schoolsupsup',
'famsup', 'paid', 'activities',
'nursery', 'higher', 'internet',
'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 pdgetdummies 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',
       ...,
       'paid', 'activities',
       ...,
       ...: 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]
```

Figure 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 yang 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.

```
In [87]: durian = durian.sample(frac=1)
...: # split training and testing data
...: durian_train = durian[:500]
...: durian_test = durian[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 = durian['pass']
...
...: # number of passing students in whole dataset:
...: import numpy as np
...: print("Passing: %d out of %d (%.2f%%)" % (np.sum(durian_pass),
len(durian_pass), 100*float(np.sum(durian_pass)) / len(durian_pass)))
Passing: 166 out of 395 (42.03%)
```

Figure 2.13: Shuffle Rows

```
5. # fit a decision tree
from sklearn import tree
mangga = tree.DecisionTreeClassifier(criterion="entropy", max_depth=5)
mangga = mangga.fit(durian_train_att, durian_train_pass)
```

Dari librari scikitlearn import modul tree. Kemudian definisikan variabel Mangga dengan menggunakan DecisionClassifier. Kemudian pada variabel mangga terdapat Criterion yaitu suatu fungsi untuk mengukur kualitas split, setelah itu agar DecisionTreeClassifier dapat dijalankan gunakan perintah fit. hasilnya seperti dibawah

```
In [32]: from sklearn import tree
...: mangga = tree.DecisionTreeClassifier(criterion="entropy", max_depth=5)
...: mangga = mangga.fit(durian_train_att, durian_train_pass)
```

Figure 2.14: Fit Decision Tree

```
6. # visualize tree
import graphviz
dot_data = tree.export_graphviz(mangga, out_file=None, label="all",
impurity=False, proportion=True,
feature_names=list(durian_train_att),
class_names=["fail", "pass"],
```

```

    filled=True, rounded=True)

graph = graphviz.Source(dot_data)
graph

```

Graphviz adalah perangkat lunak visualisasi grafik open source. Visualisasi grafik adalah cara mewakili informasi struktural sebagai diagram grafik dan jaringan abstrak. TREEEXPORTGRAPHVIZ merupakan fungsi yang menghasilkan representasi Graphviz dari decision tree, yang kemudian ditulis ke outfile. Sehingga akan muncul gambardiagram grafik bercabang.

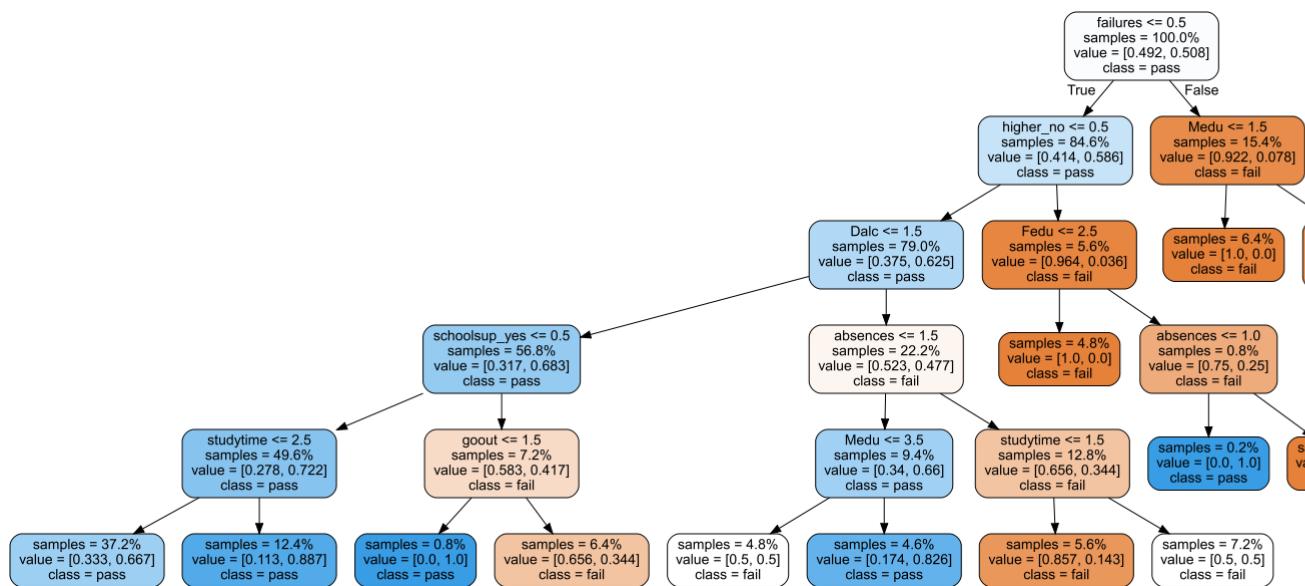


Figure 2.15: Fit Decision Tree

```

7. # save tree

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)

```

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)
```

Figure 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

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

Figure 2.17: Score

9. 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))

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)
```

Figure 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, cv=5)
print("Max depth: %d, Accuracy: %.2f (+/- %.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

```

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: %.2f (+/- %.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)

```

Figure 2.19: Max Depth

```

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()
    depth_acc[i,2] = nangka.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 nangka akan mengvalidasi score secara silang. dan pada akhirnya nangka 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):
...:     manga = tree.DecisionTreeClassifier(criterion="entropy",
max_depth=max_depth)
...:     scores = cross_val_score(manga, 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([[1.0000000e+00, 5.79751704e-01, 6.30768599e-03],
[2.0000000e+00, 5.62381532e-01, 1.05526897e-01],
[3.0000000e+00, 5.39466082e-01, 9.97942035e-02],
[4.0000000e+00, 5.67281727e-01, 0.36345308e-01],
[5.0000000e+00, 5.4692791e-01, 8.51209423e-02],
[6.0000000e+00, 5.1870334e-01, 5.25994737e-02],
[7.0000000e+00, 5.5692791e-01, 5.89895792e-02],
[8.0000000e+00, 5.39051444e-01, 6.59884566e-02],
[9.0000000e+00, 5.49018987e-01, 9.31993778e-02],
[1.0000000e+01, 5.69369523e-01, 7.14387282e-02],
[1.1000000e+01, 5.59371146e-01, 6.92276696e-02],
[1.2000000e+01, 5.46616358e-01, 7.98864960e-02],
[1.3000000e+01, 5.59339500e-01, 8.47496130e-02],
[1.4000000e+01, 5.59337877e-01, 5.37281193e-02],
[1.5000000e+01, 5.62030185e-01, 8.37479468e-02],
[1.6000000e+01, 5.54241318e-01, 8.45534525e-02],
[1.7000000e+01, 5.34149627e-01, 7.60417853e-02],
[1.8000000e+01, 5.36583901e-01, 3.95368622e-02],
[1.9000000e+01, 5.61008453e-01, 5.71807401e-02]])
```

Figure 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

2.3 Penanganan Error

HARI KEDUA TASYA WIENDHYRA 1164086

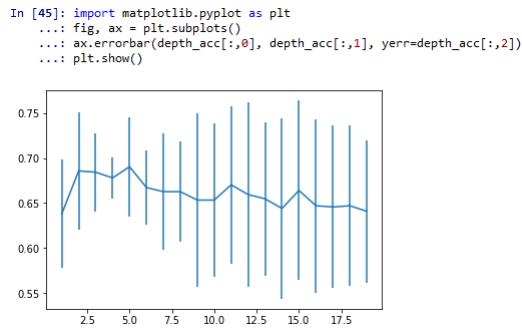


Figure 2.21: Matplotlib

2.3.1 Error Graphviz

1. Berikut ini merupakan eror 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>
```

Figure 2.22: Error Graphviz

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

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

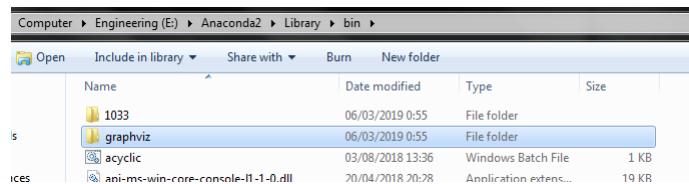


Figure 2.23: Folder Graphviz

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

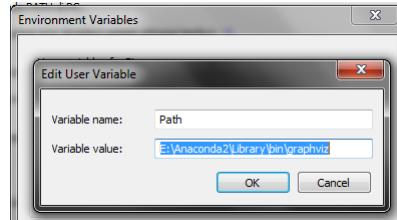


Figure 2.24: Menambahkan Graphviz kePATH

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

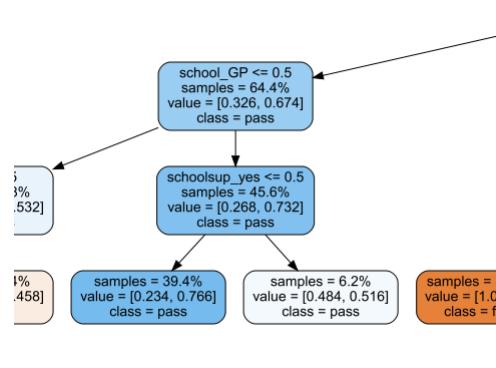


Figure 2.25: Evaluasi Eror

2.3.2 Error File Not Exist

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

```
IOError: File student-mat.csv does not exist
```

Figure 2.26: Error File Not Exist

2. Pada gambar diatas kode erornya adalah IOError File student-mat.csv does not exist. Eror 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



Figure 2.27: Kolom Direktori

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



Figure 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
```

Figure 2.29: Evaluasi Error

2.4 Annisa Fathoroni/1164067

2.4.1 Teori

Penyelesaian Tugas Harian 3 (No. 1-7)

1. Binary Classification Dan Ilustrasi Gambarnya

- Pengertian Binary Classification / Klasifikasi Biner:

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

- Ilustrasi Gambar Binary Classification:

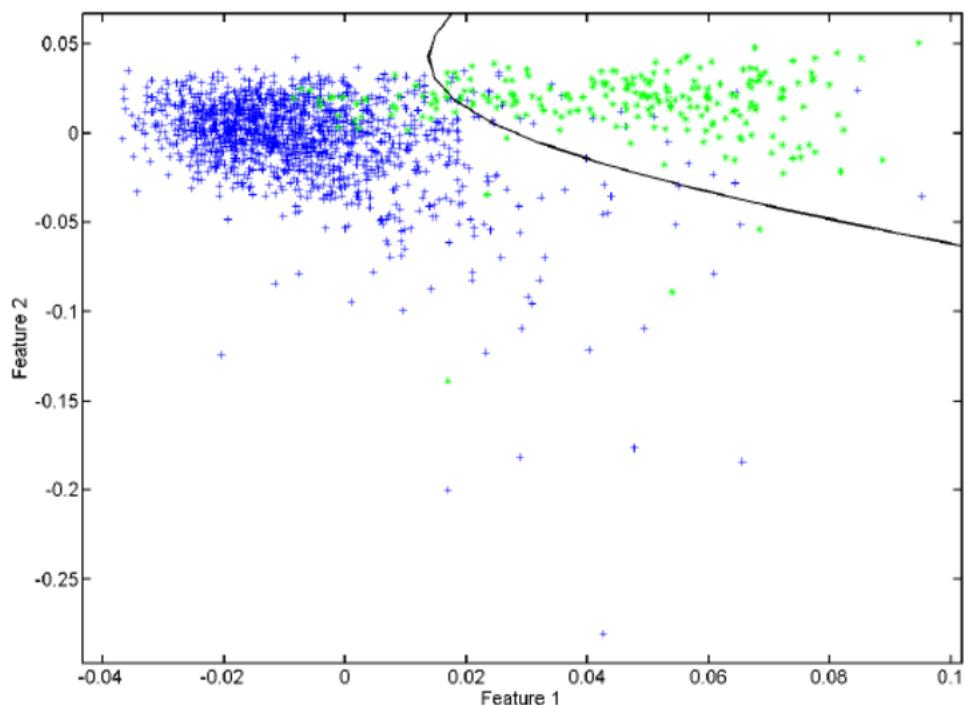


Figure 2.30: capturing

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

- Pengertian Supervised Learning dan Unsupervised Learning:

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

- Ilustrasi Gambar Supervised Learning dan Unsupervised Learning:

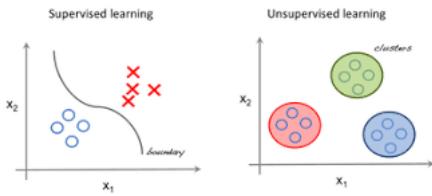


Figure 2.31: capturing

- Pengertian Clustering:

Clustering atau klasterisasi adalah metode pengelompokan data. Menurut Tan, 2006 clustering adalah sebuah proses untuk mengelompokan data ke dalam beberapa cluster atau kelompok sehingga data dalam satu cluster memiliki tingkat kemiripan yang maksimum dan data antar cluster memiliki kemiripan yang minimum. Clustering merupakan proses partisi satu set objek data ke dalam himpunan bagian yang disebut dengan cluster. Objek yang di dalam cluster memiliki kemiripan karakteristik antar satu sama lainnya dan berbeda dengan cluster yang lain. Partisi tidak dilakukan secara manual melainkan dengan suatu algoritma clustering. Oleh karena itu, clustering sangat berguna dan bisa menemukan group atau kelompok yang tidak dikenal dalam data. Clustering banyak digunakan dalam berbagai aplikasi seperti misalnya pada business intelligence, pengenalan pola citra, web search, bidang ilmu biologi, dan untuk keamanan (security). Di dalam business intelligence, clustering bisa mengatur banyak customer ke dalam banyaknya kelompok. Contohnya mengelompokan customer ke dalam beberapa cluster dengan kesamaan karakteristik yang kuat. Clustering juga dikenal sebagai data segmentasi karena clustering mempartisi banyak data set ke dalam banyak group berdasarkan kesamaannya. Selain itu clustering juga bisa sebagai outlier detection.

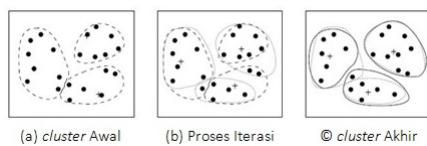


Figure 2.32: capturing

- Evaluasi, Akurasi Dan Ilustrasi Gambar

Evaluasi digunakan untuk memeriksa/memastikan dan mengevaluasi model dalam bekerja (seberapa baik) dengan mengukur keakuratannya. Kita juga dapat menanalis kesalahan yang dibuat pada model yang dijalankan, tingkat kebingungan dan menggunakan matriks kebingungan. Accuracy akan didefinisikan sebagai presentasi kasus yang diklasifikasikan dengan benar. Accuracy lebih jelasnya adalah perbandingan kasus yang diidentifikasi benar dengan jumlah semua kasus.

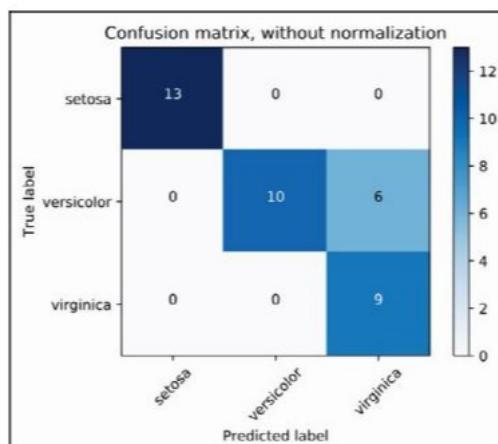


Figure 2.33: capturing

	Prediksi "Apel"	Prediksi "Jeruk"
Benar "Apel"	20	5
Benar "Jeruk"	3	22

Figure 2.34: Akurasi

- Cara membuat dan membaca confussion matrix:

Confusion matrix adalah suatu metode yang biasanya digunakan untuk melakukan perhitungan akurasi pada konsep data mining. Rumus ini melakukan perhitungan dengan 4 keluaran, yaitu: recall, precision, accuracy dan error rate. Rumus dari Error Rate = $(b+c)/(a+b+c+d)$ Keterangan: jika hasil prediksi negatif dan data sebenarnya negatif. jika hasil prediksi positif sedangkan nilai sebenarnya negatif. jika hasil prediksi negatif sedangkan nilai sebenarnya positif. jika hasil prediksi positif dan nilai sebenarnya positif. Contoh perhitungan confusion matrix adalah sebagai berikut, yaitu pengambilan keputusan untuk mendapatkan bantuan beasiswa. Saya menggunakan dua atribut, yaitu rekening listrik dan gaji.

Ini adalah pohon keputusannya: yang pertama kita lakukan yaitu mencari 4 nilai yaitu a,b,c, dan d: a= 5 b= 1 c= 1 d= 3 Kemudian kita dapat mencari nilai Recall, Precision, accuracy dan Error Rate: Recall = $3/(1+3) = 0,75$ Precision = $3/(1+3) = 0,75$ Accuracy = $(5+3)/(5+1+1+3) = 0,8$ Error Rate = $(1+1)/(5+1+1+3) = 0,2$

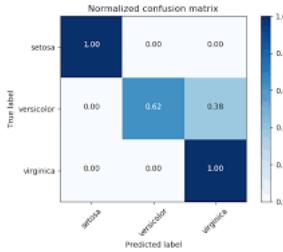


Figure 2.35: capturing

- Cara Bekerja K-Fold Cross Validation:

Total instance dibagi menjadi N bagian. Fold ke-1 adalah ketika bagian ke-1 menjadi data uji (testing data) dan Sisanya menjadi data latih (training data). Selanjutnya, hitung akurasi berdasarkan porsi data tersebut. Perhitungan akurasi tersebut dengan menggunakan persamaan sebagai berikut: Fold ke-2 adalah ketika bagian ke-2 menjadi data uji (testing data) dan sisanya menjadi data latih (training data). Selanjutnya, hitung akurasi berdasarkan porsi data tersebut. Demikian seterusnya hingga mencapai fold ke-K. Hitung rata-rata akurasi dari K buah akurasi di atas. Rata-rata akurasi ini menjadi akurasi final.

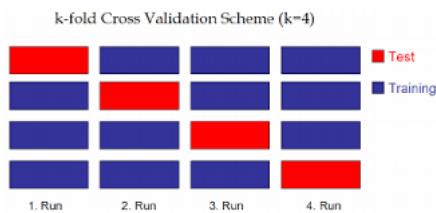


Figure 2.36: capturing

- Decision Tree

Decision tree adalah salah satu metode klasifikasi yang paling populer karena mudah diinterpretasikan oleh manusia. Decision tree digunakan untuk pengenalan pola dan termasuk dalam pengenalan pola secara statistik.

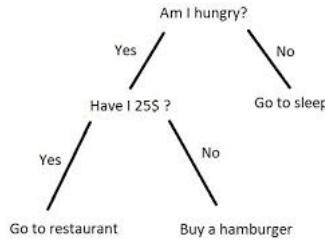


Figure 2.37: capturing

- Information Gain dan Entropi

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

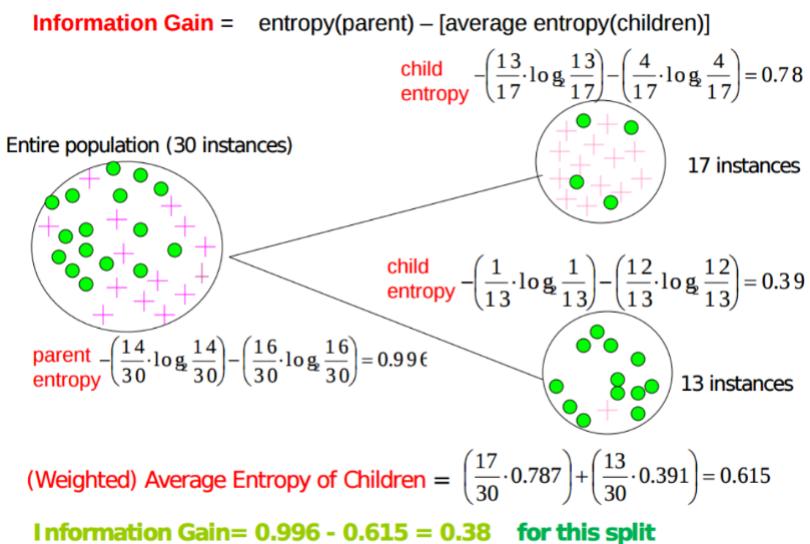


Figure 2.38: capturing

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

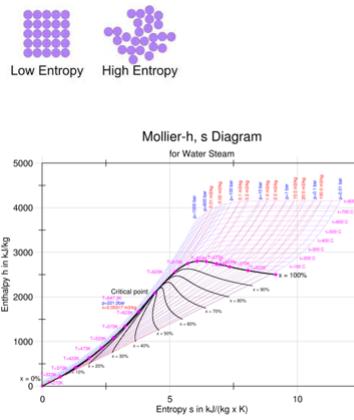


Figure 2.39: capturing

2.5 Annisa Fathoroni/1164067

2.5.1 Praktek scikit-learn

Penyelesaian Tugas Harian 4 (No. 1-12)

- Pembahasan Code dan Hasil.

1. Code 1.

Penjelasan:

Pada code pertama menjelaskan membaca berapa banyak jumlah file yang terdapat pada file 'student-por.csv'.

– Hasil Code 1:

```
In [28]: import pandas as pd
...: jakarta = pd.read_csv('student-por.csv', sep=';')
...: len(jakarta)
Out[28]: 649
```

Figure 2.40: Hasil Code 1

2. Code 2.

Penjelasan:

Pada code kedua menjelaskan variabel jakarta akan menampilkan baris G1, G2 dan G3 dengan ketentuan lebih dari 35 maka akan muncul secara live atau langsung data yang diperintahkan pada code tersebut.

– Hasil Code 2:

```

In [29]: jakarta['pass'] = jakarta.apply(lambda row: 1 if (row['G1']+row['G2']
+row['G3']) >= 35 else 0, axis=1)
...: jakarta = jakarta.drop(['G1', 'G2', 'G3'], axis=1)
...: jakarta.head()
Out[29]:
   school sex age address famsize ... Dalc Walc health absences pass
0    GP   F  18      U   GT3 ...   1    1     3     4    0
1    GP   F  17      U   GT3 ...   1    1     3     2    0
2    GP   F  15      U   LE3 ...   2    3     3     6    1
3    GP   F  15      U   GT3 ...   1    1     5     0    1
4    GP   F  16      U   GT3 ...   1    2     5     0    1
[5 rows x 31 columns]

```

Figure 2.41: Hasil Code 2

3. Code 3.

Penjelasan:

Pada code ketiga menjelaskan variabel jakarta memanggil fungsi dummies yang memiliki beberapa kolom yang akan ditampilkan apabila perintah print dilakukan.

— Hasil Code 3:

```

In [30]: jakarta = pd.get_dummies(jakarta, columns=['sex', 'school', 'address',
       'famsize', 'Pstatus', 'Mjob', 'Fjob',
       'reason', 'guardian', 'schoolsup', 'famsup',
       'activities', 'nursery', 'higher', 'internet', 'romantic'])
...: jakarta.head()
Out[30]:
   age Medu Fedu ... internet_yes romantic_no romantic_yes
0    17    4    4 ...          0           1           0
1    17    1    1 ...          1           1           0
2    15    1    1 ...          1           1           0
3    15    4    2 ...          1           0           1
4    16    3    3 ...          0           1           0
[5 rows x 57 columns]

```

Figure 2.42: Hasil Code 3

4. Code 4.

Penjelasan:

Pada code keempat variabel solok akan menampilkan sampel data dari 500 training data dan 500 testing data. Kemudian data akan dicetak atau di print dari training data dan testing data.

— Hasil Code 4:

```

In [31]: jakarta = jakarta.sample(frac=1)
...: #split training and testing data
...: jakarta_train = jakarta[:500]
...: jakarta_test = jakarta[500:]
...:
...: jakarta_train_att = jakarta_train.drop(['pass'], axis=1)
...: jakarta_train_pass = jakarta_train['pass']
...:
...: jakarta_test_att = jakarta_test.drop(['pass'], axis=1)
...: jakarta_test_pass = jakarta_test['pass']
...:
...: jakarta_att = jakarta.drop(['pass'], axis=1)
...: jakarta_pass = jakarta['pass']
...:
...: # number of passing students in whole dataset:
...: import numpy as np
...: print("Passing: %d out of %d (%.2f%%)" % (np.sum(jakarta_pass),
len(jakarta_pass), 100*float(np.sum(jakarta_pass)) / len(jakarta_pass)))
Passing: 322 out of 649 (50.54%)

```

Figure 2.43: Hasil Code 4

5. Code 5.

Penjelasan:

Pada code kelima menguji klasifikasi Decision Tree apakah berjalan dengan yang sudah ditentukan atau tidak.

– Hasil Code 5:

```
In [59]: from sklearn import tree
...: bandung = tree.DecisionTreeClassifier(criterion="entropy", max_depth=5)
...: bandung = bandung.fit(jakarta_train_att, jakarta_train_pass)
```

Figure 2.44: Hasil Code 5

6. Code 6.

Penjelasan:

Pada code keenam menampilkan decision tree.

– Hasil Code 6:

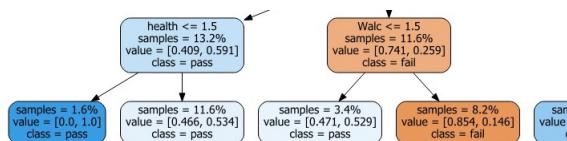


Figure 2.45: Hasil Code 6

7. Code 7.

Penjelasan:

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

– Hasil Code 7:

```
In [60]: tree.export_graphviz(bandung, out_file="student-performance.dot", label="all",
...: impurity=False, proportion=True,
...: feature_names=list(jakarta_train_att),
...: class_names=["fail", "pass"],
...: filled=True, rounded=True)
```

Figure 2.46: Hasil Code 7

8. Code 8.

Penjelasan:

Pada code kedelapan menampilkan hasil perhitungan dari kedua parameter yang terdapat pada code tersebut.

– Hasil Code 8:

```
In [33]: bandung.score(jakarta_test_att, jakarta_test_pass)
Out[33]: 0.6308724832214765
```

Figure 2.47: Hasil Code 8

9. Code 9.

Penjelasan:

Pada code kesembilan membaca isi dari library, memanggil score yang terdapat pada parameter code tersebut. Sehingga apabila di print memanggil hasil output akan menghasilkan sebuah angka akurasi.

— Hasil Code 9:

```
In [34]: from sklearn.model_selection import cross_val_score
...: scores = cross_val_score(bandung, jakarta_att, jakarta_pass, cv=5)
...: # show average score and +/- two standard deviations away (covering 95% of
...: scores
...: print("Accuracy: %0.2f (+/- %0.2f)" % (scores.mean(), scores.std() * 2))
Accuracy: 0.65 (+/- 0.03)
```

Figure 2.48: Hasil Code 9

10. Code 10.

Penjelasan:

Pada code kesepuluh menampilkan hasil dari fungsi Max Depth dan Accuracy dari dari Decission Tree. Yaitu menmpilkan data dari angka 1-20.

— Hasil Code 10:

```
In [35]: for max_depth in range(1, 20):
...:     bandung = tree.DecisionTreeClassifier(criterion="entropy",
...:     max_depth=max_depth)
...:     scores = cross_val_score(bandung, jakarta_att, jakarta_pass, cv=5)
...:     print("Max depth: %d, Accuracy: %0.2f (+/- %0.2f)" % (max_depth,
...:     scores.mean(), scores.std() * 2))
Max depth: 1, Accuracy: 0.65 (+/- 0.02)
Max depth: 2, Accuracy: 0.69 (+/- 0.06)
Max depth: 3, Accuracy: 0.69 (+/- 0.08)
Max depth: 4, Accuracy: 0.68 (+/- 0.08)
Max depth: 5, Accuracy: 0.67 (+/- 0.11)
Max depth: 6, Accuracy: 0.67 (+/- 0.08)
Max depth: 7, Accuracy: 0.67 (+/- 0.07)
Max depth: 8, Accuracy: 0.65 (+/- 0.07)
Max depth: 9, Accuracy: 0.65 (+/- 0.07)
Max depth: 10, Accuracy: 0.65 (+/- 0.06)
Max depth: 11, Accuracy: 0.64 (+/- 0.07)
Max depth: 12, Accuracy: 0.63 (+/- 0.05)
Max depth: 13, Accuracy: 0.64 (+/- 0.06)
Max depth: 14, Accuracy: 0.64 (+/- 0.06)
Max depth: 15, Accuracy: 0.64 (+/- 0.08)
Max depth: 16, Accuracy: 0.65 (+/- 0.09)
Max depth: 17, Accuracy: 0.65 (+/- 0.04)
Max depth: 18, Accuracy: 0.62 (+/- 0.08)
Max depth: 19, Accuracy: 0.61 (+/- 0.08)
```

Figure 2.49: Hasil Code 10

11. Code 11.

Penjelasan:

Pada code kesebelas menjelaskan variable scores akan menampilkan atau mendefinisikan nilai dari variabel score.

— Hasil Code 11:

```
In [44]: depth_acc = np.empty((19,3), float)
...: bali = 0
...: for max_depth in range(1, 20):
...:     bandung = tree.DecisionTreeClassifier(criterion="entropy",
max_depth=max_depth)
...:     scores = cross_val_score(bandung, jakarta_att, jakarta_pass, cv=5)
...:     depth_acc[bali,0] = max_depth
...:     depth_acc[bali,1] = scores.mean()
...:     depth_acc[bali,2] = scores.std() * 2
...:     bali += 1
...:
...:
...: depth_acc
```

Figure 2.50: Hasil Code 11

```
Out[44]:
array([[ 1.        ,  0.63798714,  0.0604711 ],
       [ 2.        ,  0.68570475,  0.06518688],
       [ 3.        ,  0.6842373 ,  0.043406  ],
       [ 4.        ,  0.6779174 ,  0.02305337],
       [ 5.        ,  0.69034417,  0.05515479],
       [ 6.        ,  0.667219 ,  0.04153057],
       [ 7.        ,  0.66261463,  0.06484386],
       [ 8.        ,  0.66260361,  0.05592344],
       [ 9.        ,  0.65330183,  0.09688048],
      [10.        ,  0.65326569,  0.08495537],
      [11.        ,  0.67026051,  0.08772849],
      [12.        ,  0.65938412,  0.10294259],
      [13.        ,  0.654757 ,  0.08537065],
      [14.        ,  0.64396355,  0.10036248],
      [15.        ,  0.66403529,  0.09982166],
      [16.        ,  0.64705222,  0.09644585],
      [17.        ,  0.6455732 ,  0.09033473],
      [18.        ,  0.64700506,  0.08952031],
      [19.        ,  0.64085085,  0.078958 ]])
```

Figure 2.51: Hasil Code 11

12. Code 12.

Penjelasan:

Pada code kedua belas menjelaskan pada library matplotlib akan menampilkan gambar grafik pada gambar 12 dari eksekusi fungsi ax.errorbar.

– Hasil Code 12:

2.5.2 Praktek Penanganan Error

Traceback (most recent call last):

```
File "<ipython-input-33-b4843d06cf2>", line 1, in <module>
import graphviz
```

```
ModuleNotFoundError: No module named 'graphviz'
```

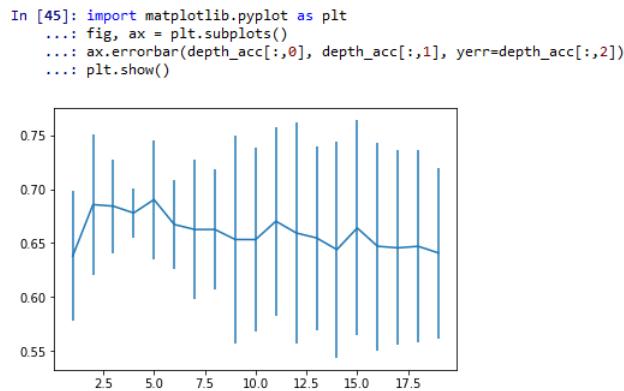


Figure 2.52: Hasil Code 12

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

Figure 2.53: HASIL YANG MASIH ERROR

Solusi dari error tersebut adalah memasukkan path dari graphviz ke environment variabels, maka akan menampilkan hasil yaitu sebuah decision tree.

2.6 Annisacahyani/1164066

2.6.1 Teori

Penyelesaian Tugas Harian 3 (No. 1-7)

1. Binary Classification Dan Ilustrasi Gambarnya

- Pengertian Binary Classification / Klasifikasi Biner:

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

1. Supervised Learning, Unsupervised Learning, Clustering Dan Ilustrasi Gambar

- Pengertian Supervised Learning, unsuvsised learning,dan clustering :

Supervised learning adalah tugas pembelajaran mesin untuk mempelajari fungsi yang memetakan input ke output berdasarkan contoh pasangan input-output. Unsupervised learning yaitu istilah yang digunakan

untuk pembelajaran bahasa Ibrani, yang terkait dengan pembelajaran tanpa guru, juga dikenal sebagai organisasi mandiri dan metode pemodelan kepadatan probabilitas input. Dan Clustering dapat dianggap sebagai masalah pembelajaran tanpa pengawasan yang paling penting jadi, karena setiap masalah lain dari jenis ini, ini berkaitan dengan menemukan struktur dalam kumpulan data yang tidak berlabel.

1. Evaluasi, Akurasi Dan Ilustrasi Gambar

- Pengertian Evaluasi

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

1. Membuat Dan Membaca Confusion Matrix Beserta Contoh

- Pengertian Confusion Matrix

Confusion matrix adalah suatu metode yang biasanya digunakan untuk melakukan sebuah perhitungan akurasi pada konsep data mining atau Sistem Pendukung Keputusan. Pada pengukuran kinerja menggunakan confusion matrix, terdapat 4 (empat) istilah sebagai representasi hasil proses klasifikasi. Keempat istilah tersebut adalah True Positive (TP), True Negative (TN), False Positive (FP) dan False Negative(FN).

- Pembacaan Confusion Matrix

1. Jika hasil prediksi menghasilkan negatif dan data yang sebenarnya itu merupakan negatif.
2. Jika hasil prediksi menghasilkan positif sedangkan nilai yang sebenarnya itu merupakan negatif.
3. Jika hasil prediksi menghasilkan negatif sedangkan nilai yang sebenarnya itu merupakan positif.
4. Jika hasil prediksi menghasilkan yang positif dan nilai sebenarnya merupakan positif.

- Pembuatan Confusion Matrix

1. Yang pertama yaitu untuk menentukan 4 proses klasifikasi yang akan kita gunakan dalam confusion matrix.
2. Kemudian 4 Istilah yang ada tersebut yaitu ada True Positive (TP), True Negative (TN), False Positive (FP) dan False Negative (FN).
3. Kemudian kita kelompokkan klasifikasi tersebut dengan menggunakan klasifikasi biner
4. Kemudian dari hal tersebut akan menghasilkan keluaran berupa 2 Kelas yaitu (Positif dan Negatif) dan penentuan TP, FP (1 klasifikasi positif), FN dan TN (1 klasifikasi negatif).

- Penjelasan

1. Recall

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

- Cara Kerja K-Fold Classification Dan Ilustrasi Gambar

1. Pertama kita total instance dibagi menjadi N bagian.
2. Kemudian Fold ke-1 adalah ketika bagian ke-1 menjadi data uji (testing data) dan sisanya menjadi data latih (training data).
3. Setelah itu fold yang ke-2 adalah ketika bagian ke-2 menjadi data uji (testing data) dan sisanya menjadi data latih (training data).
4. Demikian sampai seterusnya hingga mencapai fold ke-K. Hitung rata-rata akurasi dari K buah akurasi di atas. Rata-rata akurasi ini menjadi akurasi final

1. Decision Tree Dan Ilustrasi Gambar

- Pengertian Decision Tree

Decision Tree yaitu alat pendukung keputusan yang menggunakan model keputusan seperti pohon dan konsekuensinya yang mungkin, termasuk hasil acara kebetulan, biaya sumber daya, dan utilitas.

1. Information Gain Dan Entropi

- Pengertian Information Gain dan entropi

information gain didasarkan pada penurunan entropi setelah dataset dibagi pada atribut. Membangun pohon keputusan adalah semua tentang menemukan atribut yang mengembalikan perolehan informasi tertinggi. Sedangkan Entropi itu dibangun dari atas ke bawah dari simpul akar dan melibatkan mempartisi data ke dalam himpunan bagian yang berisi instance dengan nilai yang sama yaitu homogen.

2.7 Annisa Cahyani/1164066

2.7.1 Praktek Scikit-Learn

Penyelesaian Tugas Harian 4 (No. 1-12)

1. Pembahasan Codingan Dan Hasilnya

(a) Codingan Pertama :

Penjelasan : Codingan pertama ini akan meload (menampilkan) data pada file yang ditentukan. Untuk codingan ini file yang dieksekusi ialah "student-mat.csv" .

• Hasil Codingan Pertama :

```
In [4]: import pandas as burger
...: aci = burger.read_csv('dataset/student-mat.csv', sep=';')
...: len(aci)
Out[4]: 395
```

Figure 2.54: codingan pertama

(b) Codingan Kedua :

Penjelasan : Codingan kedua ini secara keseluruhan yaitu akan menampilkan baris G1, G2 dan G3 (berdasarkan kriterianya) untuk kolom PASS pada variabel.

• Hasil Codingan Kedua :

(c) Codingan Ketiga :

Penjelasan : Secara keseluruhan, codingan ini akan mendefinisikan pemanggilan ke get dummies dari pizza dalam variabel bakso. Di dalam get dummies sendiri akan terdefinisikan variabel dengan kolom-kolom yang akan dieksekusi seperti school, address dll

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

Figure 2.55: codingan kedua

```
In [6]: aci = burger.get_dummies(aci, columns=['sex', 'school', 'address',
...: 'famsiz', 'Pstatus', 'Mjob', 'Fjob', 'reason', 'guardian', 'schoolsup',
...: 'famsup', 'paid', 'activities', 'nursery', 'higher', 'internet',
...: 'romantic'])
...: aci.head()
Out[6]:
   age Medu Fedu ... internet_yes romantic_no romantic_yes
0   16    4    4 ...          0           1           0
1   17    1    1 ...          1           1           0
2   15    1    1 ...          1           1           0
3   15    4    2 ...          1           0           1
4   16    3    3 ...          0           1           0
[5 rows x 57 columns]
```

Figure 2.56: codingan ketiga

- Hasil Codingan Ketiga :

(d) Codingan Keempat :

Penjelasan : Secara keseluruhan codingan ini difungsikan untuk mendefinisikan pembagian data yang berupa training dan testing data. Secara jelasnya pertama-tama variabel akan mendefinisikan sampel yang akan digunakan (berupa shuffle row). Nah kemudian masing-masing dari parameter tersebut akan berjumlah 500 data (telah dibagi untuk training dan testing).

- Hasil Codingan Keempat :

```
In [8]: aci = aci.sample(frac=1)
...: # split training and testing data
...: aci_train = aci[:500]
...: aci_test = aci[500:]
...: aci_train.att = aci_train.drop(['pass'], axis=1)
...: aci_train_pass = aci_train['pass']
...: aci_test_att = aci_test.drop(['pass'], axis=1)
...: aci_test_pass = aci_test['pass']
...: aci.att = aci.drop(['pass'], axis=1)
...: aci_pass = aci['pass']
...:
...: # number of passing students in whole dataset:
...: import numpy as np
...: print("Passing: %d out of %d (%.2f%%)" % (np.sum(aci_pass),
len(aci_pass), 100*float(np.sum(aci_pass)) / len(aci_pass)))
Passing: 166 out of 395 (42.05%)
```

Figure 2.57: codingan keempat

(e) Codingan Kelima :

Penjelasan : Secara keseluruhan, codingan ini hanya membuktikan pada bagian pengujian dari Klasifikasi Decision Tree yang ada, apakah true atau

tidak dan hasilnya true. Apabila dibahas dengan lengkap maka pada codingan ini di definisikan library sklearn untuk mengimpor atau menampilkan tree.

- Hasil Codingan Kelima :

```
In [9]: from sklearn import tree
...: ayam = tree.DecisionTreeClassifier(criterion="entropy", max_depth=5)
...: ayam = ayam.fit(aci_train_att, aci_train_pass)
```

Figure 2.58: codingan kelima

(f) Codingan Keenam :

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

- Hasil Codingan Keenam :

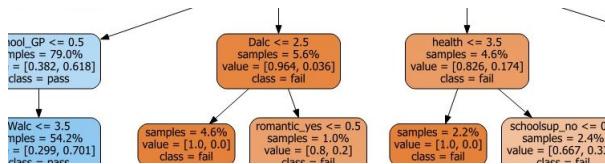


Figure 2.59: supervised

(g) Codingan Ketujuh :

Penjelasan : Secara keseluruhan, codingan ini membahas tentang penyimpanan tree dari library graphiz yang dieksekusi bersamaan dengan variabel dan parameter lainnya. Dilakukan pengecekan dan pengujian apakah klasifikasi decision treenya dapat berjalan atau tidak

- Hasil Codingan Ketujuh :

```
In [10]: tree.export_graphviz(ayam, out_file="student-performance.dot",
label="all", impurity=False, proportion=True,
...: feature_names=list(aci_train_att),
...: class_names=["fail", "pass"],
...: filled=True, rounded=True)
```

Figure 2.60: codingan ketujuh

(h) Codingan Kedelapan :

Penjelasan : Secara keseluruhan, codingan ini membaca score dari variabel sate dimana terdapat 2 parameter yang dihitung dan diuji. Untuk hasilnya

sendiri mengapa hasilnya berupa angka, dikarenakan pada parameter yang dieksekusi memang memiliki data sehingga dieksekusi dan menghasilkan keluaran dari score tersebut.

- Hasil Codingan Kedelapan :

```
In [33]: ayam.score( aci_test_att, aci_test_pass)
Out[33]: 0.6308724832214765
```

Figure 2.61: codingan kedelapan

- (i) Codingan Kesembilan :

Penjelasan : Secara keseluruhan, codingan ini membahas tentang pengksekusian fungsi dan variabel dari library yang didefinisikan dan yang diimport. Penjelasan lebih jelasnya ialah codingan ini mendefinisikan library sklearn.model.selection kemudian mengimpor cross val score. Kemudian variabel score mendefinisikan cross val score yang telah diimport tadi dengan 4 parameter.

- Hasil Codingan Kesembilan :

```
In [12]: from sklearn.model_selection import cross_val_score
...: scores = cross_val_score(ayam, aci_att, aci_pass, cv=5)
...: # show average score and +/- two standard deviations away (covering 95%
of scores)
...: print("Accuracy: %.2f (+/- %.2f)" % (scores.mean(), scores.std() * 2))
Accuracy: 0.53 (+/- 0.09)
```

Figure 2.62: codingan kesembilan

- (j) Codingan Ke-10 :

Penjelasan : Codingan ini mendefinisikan max depth dalam jarak angka antara parameter yang ke 1 dan 20. Variabel mendefinisikan klasifier decision tree dengan 2 parameter.

- Hasil Codingan Ke-10 :

- (k) Codingan Ke-11 :

Penjelasan : Codingan ini mendefinisikan bahwa variabel cahya akan mengeksekusi empty dari importan library numpy yang dinamakan np dengan 2 parameter yaitu 19,3 dan float. i didefinisikan dengan angka 0 kemudian untuk perhitungan jarak max depth diantara parameter 1 dan 20. Variabel yang mendefinisikan klasifikasi decision tree dengan 2 parameter.

```

In [13]: for max_depth in range(1, 20):
    ...:     ayam = tree.DecisionTreeClassifier(criterion="entropy",
    max_depth=max_depth)
    ...:     scores = cross_val_score(ayam, aci_att, aci_pass, cv=5)
    ...:     print("Max depth: %d, Accuracy: %.2f (+/- %.2f)" % (max_depth,
scores.mean(), scores.std() * 2))
Max depth: 1, Accuracy: 0.58 (+/- 0.01)
Max depth: 2, Accuracy: 0.58 (+/- 0.06)
Max depth: 3, Accuracy: 0.55 (+/- 0.04)
Max depth: 4, Accuracy: 0.53 (+/- 0.09)
Max depth: 5, Accuracy: 0.53 (+/- 0.07)
Max depth: 6, Accuracy: 0.55 (+/- 0.10)
Max depth: 7, Accuracy: 0.53 (+/- 0.04)
Max depth: 8, Accuracy: 0.53 (+/- 0.05)
Max depth: 9, Accuracy: 0.52 (+/- 0.08)
Max depth: 10, Accuracy: 0.52 (+/- 0.09)
Max depth: 11, Accuracy: 0.53 (+/- 0.11)
Max depth: 12, Accuracy: 0.50 (+/- 0.07)
Max depth: 13, Accuracy: 0.53 (+/- 0.12)
Max depth: 14, Accuracy: 0.53 (+/- 0.11)
Max depth: 15, Accuracy: 0.51 (+/- 0.15)
Max depth: 16, Accuracy: 0.53 (+/- 0.10)
Max depth: 17, Accuracy: 0.52 (+/- 0.11)
Max depth: 18, Accuracy: 0.51 (+/- 0.12)
Max depth: 19, Accuracy: 0.53 (+/- 0.11)

```

Figure 2.63: codingan ke-10

```

...: i = 0
...: for max_depth in range(1, 20):
...:     satu = tree.DecisionTreeClassifier(criterion="entropy", max_depth=max_depth)
...:     scores = cross_val_score(satu, aci_att, aci_pass, cv=5)
...:     cahya[i,0] = max_depth
...:     cahya[i,1] = scores.mean()
...:     cahya[i,2] = scores.std() * 2
...:     i += 1
...:
...: cahya
Out[14]:
array([[ 1.,  0.53661636,  0.09689923],
[ 2.,  0.55718922,  0.08339407],
[ 3.,  0.53361717,  0.12523223],
[ 4.,  0.53161717,  0.12523222],
[ 5.,  0.5164273,  0.12252795],
[ 6.,  0.5216473,  0.12252797],
[ 7.,  0.52481899,  0.11077078],
[ 8.,  0.54562204,  0.11304654],
[ 9.,  0.52755001,  0.11304653],
[10., 0.5348839,  0.080883913],
[11., 0.5336157,  0.11465665],
[12., 0.53386157,  0.11465665],
[13., 0.51139322,  0.13613236],
[14., 0.51581798,  0.13393507],
[15., 0.53921398,  0.0712946 ]])

```

Figure 2.64: codingan ke-11

- Hasil Codingan Ke-11 :

- (1) Codingan Ke-12 :

Penjelasan : Codingan ini mendefinisikan untuk pemanggilan dari library matplotlib.pyplot sebagai cahya sehingga nanti hasilnya akan berbentuk gambar grafik/gelombang.

- Hasil Codingan Ke-12 :

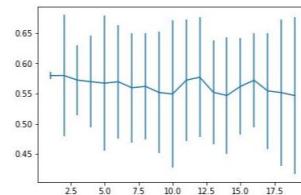


Figure 2.65: supervised

2.7.2 Penanganan Error

Pembahasan dan Penyelesaian Error

1. Error 1 :

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

Figure 2.66: error 1

- Penjelasan :

- Pada error tersebut dapat dikatakan bahwa untuk file -b”student-mat.csv” tidak ada. Mengapa? karena file pada codingan yang dieksekusi yaitu student-performance.py tidak berada pada folder yang sama.
- Silahkan menambahkan fungsi yang mendefinisikan folder tempat file ”student-mat.csv” berada.
- Silahkan tambahkan perintah ” dataset/student-mat.csv ” pada codingannya
- Dengan penambahan perintah tersebut maka tidak akan terjadi error lagi.

Chapter 3

Methods

3.1 ANNISA FATHORONI/1164067

3.1.1 Teori

Penyelesaian Tugas Harian 5 (No. 1-6)

1. Random Forest Dan Ilustrasi Gambarnya

- Pengertian Random Forest:

Random forests atau random decision forests adalah metode pembelajaran ensembel untuk klasifikasi, regresi dan tugas-tugas lain yang beroperasi dengan membangun banyak pohon keputusan pada waktu pelatihan dan menghasilkan kelas yang merupakan mode kelas (klasifikasi) atau prediksi rata-rata (regresi) dari masing-masing pohon. Random decision forests tepat untuk kebiasaan pohon keputusan 'overfitting' pada set pelatihan mereka.

- Ilustrasi Gambar Random Forest :

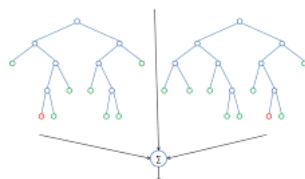


Figure 3.1: Random Forest

2. Cara Membaca Dataset

Berikut adalah cara membaca dataset :

- Buka Anaconda Navigator lalu jalankan Syder, kemudian import libraries yang dibutuhkan.
- Masukkan kode python untuk membaca file csv, lalu jalankan

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

Figure 3.2: Code Python

- Maka pada window console akan menampilkan pesan berikut:

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

Figure 3.3: Output

- Dari explorer dapat terlihat dataset yang terimport.

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

Figure 3.4: Import Dataset

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

3. Cross Validation

Cross-validation (CV) adalah metode statistik yang dapat digunakan untuk mengevaluasi kinerja model atau algoritma dimana data dipisahkan menjadi dua subset yaitu data proses pembelajaran dan data validasi / evaluasi. Model atau algoritma dilatih oleh subset pembelajaran dan divalidasi oleh subset validasi. Selanjutnya pemilihan jenis CV dapat didasarkan pada ukuran dataset.

Index	Country	Age	Salary	Purchased
0	France	44	72000	No
1	Spain	27	48000	Yes
2	Germany	38	54000	No
3	Spain	38	61000	No
4	Germany	40	nan	Yes
5	France	35	58000	Yes
6	Spain	nan	52000	No
7	France	48	76000	Yes
8	Germany	58	83000	No
9	France	37	67000	Yes

Figure 3.5: Hasil Dataset Sel

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

Figure 3.6: Perintah

Biasanya CV K-fold digunakan karena dapat mengurangi waktu komputasi dengan tetap menjaga keakuratan estimasi.

4. Arti score 44% pada random forest, 27% pada decision tree dan 29% dari SVM.

Kalau maksud arti score 27% pada decision tree adalah presentasi hasil dari perhitungan dataset, sedangkan maksud arti score 29% dari SVM adalah hasil pendekatan jaringan saraf. Hasil tersebut didapat dari hasil validasi silang untuk memastikan bahwa membagi training test dengan cara yang berbeda. Sehingga didapat outputnya 44% untuk hutan acak, 27% untuk pohon keputusan, dan 29% untuk SVM.

5. Confusion Matrix Dan Ilustrasinya

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

Kemudian data testingnya adalah

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

$$a= 5$$

$$b= 1$$

$$c= 1$$

$$d= 3$$

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

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

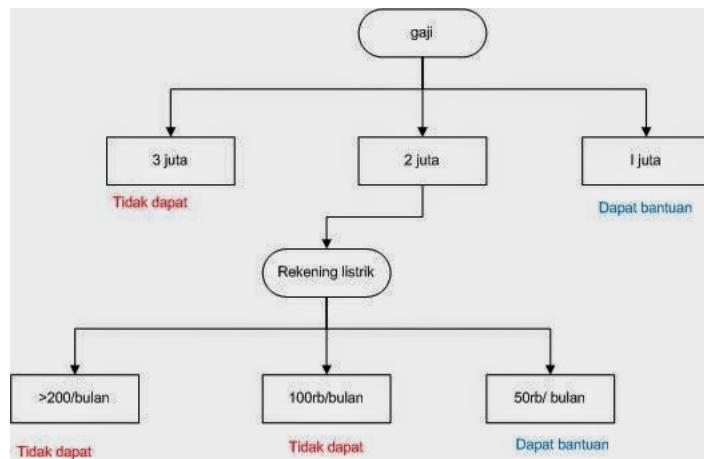


Figure 3.7: Pohon Keputusan

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

Figure 3.8: Data Testing

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

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

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

6. Voting Random Forest Dan Ilustrasi Gambarnya.

- Pengertian Voting pada Random Forest

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

- Ilustrasi Gambar Voting Random Forest :

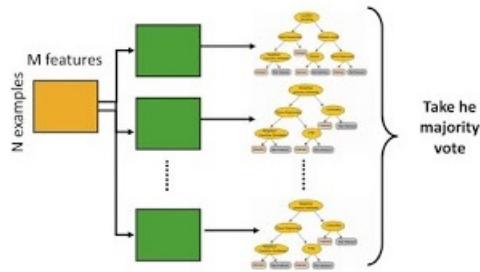


Figure 3.9: Voting Random Forest

3.2 Tasya Wiendhyra/1164086

HARI PERTAMA TEORI

3.3 Random Forest Dan Contohnya

3.3.1 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 hyper-parameter, 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

```
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
```

Figure 3.10: Random Forest Spyder

Setelah di plotting hasilnya seperti berikut

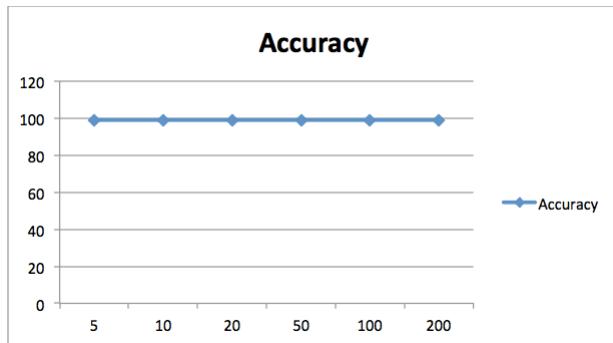


Figure 3.11: Random Forest Graphic

3.4 Dataset

3.4.1 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.

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

1. Gunakan librari 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()
```

Figure 3.12: 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 :

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

Figure 3.13: 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/ml/datasets/Car+Evaluation>.

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 kedapalm mobil, dengan value 2,4, more /lebih.
- Atribut Lug Boot merupakan ukuran bagasi boot mobil, dengan value small,med,big.
- Atribut Safety merupakan perkiraan keselamatan mobil, dengan value low,med,high.
- Yang terakhir yaitu Value, yang dimana merupakan merupakan Class nya atau disebut dengan targetnya menyatakan apakah mobil tersebut dapat diterima atau tidak dan apakah mobil tersebut bagus atau tidak, dengan value unacc, acc, good,v good .

3.5 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.

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

Itu merupakan presentase keakurasiyan prediksi yang dilakukan pada saat testing menggunakan label pada dataset yang digunakan. Score merupakan mendefinisikan aturan evaluasi model. Maka pada saat dijalankan akan muncuk persentase tersebut yang menunjukan keakurasiyan 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% .

3.7 Confusion Matriks

3.7.1 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=[

def plot_confusion_matrix(df_confusion, title='Confusion matrix', cmap=plt.cm.

    plt.matshow(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 :

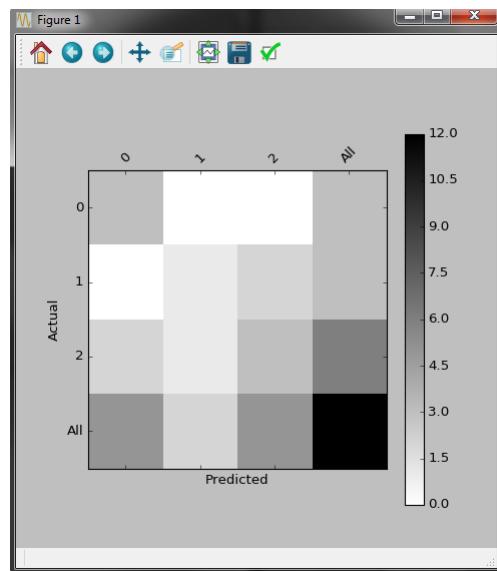


Figure 3.14: Confusion Matrix

3.8 Voting Pada Random Forest

3.8.1 Pengertian

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

3.8.2 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, random_state=123)
clf2 = RandomForestClassifier(n_estimators=100, random_state=123)
clf3 = GaussianNB()

X = np.array([-1.0, -1.0], [-1.2, -1.4], [-3.4, -2.2], [1.1, 1.2])
y = np.array([1, 1, 2, 2])

eclf = VotingClassifier(estimators=[('lr', clf1), ('rf', clf2), ('gnb', clf3),
                                    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, clf3, eclf)]

# 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]])), width,
             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 left')
plt.tight_layout()
plt.show()

```

- Hasilnya sebagai berikut

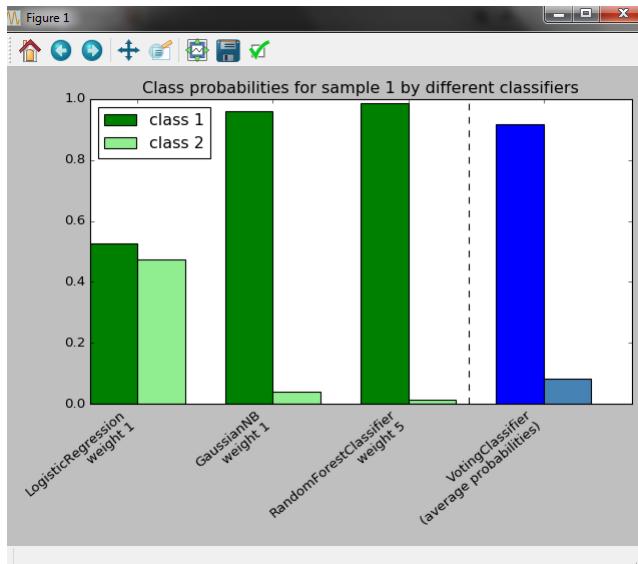


Figure 3.15: Voting Random Forest

3.9 Annisa Cahyani/1164066

3.9.1 Teori

Penyelesaian Tugas Harian 5 (No. 1-6)

1. Random Forest Dan Ilustrasi Gambarnya

- Pengertian Random Forest:

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

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

- Cara Membaca Dataset :

Membaca Dataset

1. Langkah-langkah yang harus di lakukan yaitu:

- Pertama kita harus membuka Aplikasi yang digunakan untuk membuka datasetnya terlebih dahulu
- saya menggunakan spyder dari Anaconda Navigator
- kemudian kita import libraries seperti numpy, pandas, matplotlib dll (sesuai yang dibutuhkan).
- lalu untuk memasukkan kode python yang akan digunakan untuk membaca file csv yang berada pada dataset

2. Cross Validation Dan Ilustrasi Gambar :

- Pengertian Cross Validation :

Cross Validation merupakan salah satu dari berbagai teknik validasi untuk menilai bagaimana hasil analisis statistik yang akan digeneralisasikan ke set data independen. Biasanya yang digunakan dalam pengaturan itu tujuannya untuk memprediksi, dan memperkirakan seberapa akurat model prediksi yang akan dilakukan dalam praktik.

3. Penjelasan / Maksud Dari Score pada :

- Random forest (44 %)
- Decision Tree (27 %)
- SVM (29 %)

4. Confusion Matrix Dan Ilustrasinya :

- Pengertian Confusion Matrix :

Confusion matrix merupakan salah satu metode yang dapat digunakan untuk mengukur kinerja suatu metode klasifikasi. Pada dasarnya confusion matrix mengandung informasi yang membandingkan hasil klasifikasi yang dilakukan oleh sistem dengan hasil klasifikasi yang seharusnya.

- Cara Membaca Confusion Matrix :

Untuk cara pembacaan dan pemahaman pada confusion matrix sebagai berikut :

- Ketika hasil prediksinya itu negatif dan data sebenarnya merupakan negatif.

- Ketika hasil prediksinya itu positif sedangkan nilai sebenarnya merupakan negatif.
- Ketika hasil prediksinya itu negatif sedangkan nilai sebenarnya merupakan positif.
- Ketika hasil prediksinya itu positif dan nilai sebenarnya merupakan positif.

5. Voting Random Forest Dan Ilustrasi Gambarnya.

- Pengertian Voting pada Random Forest :

Voting pada random forest yaitu suatu penetuan hasil klasifikasi yang akan diambil atau hasil dari hasil pengklasifikasian data

3.10 Praktek Program

3.11 Tasya Wiendhyra / 1164086

HARI KEDUA PRAKTEK MINGGU KETIGA

3.11.1 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.

Listing 3.1: Code Program Sederhana Pandas

```
import pandas as pd
d = {'kol1': [1, 4, 3, 4, 5], 'kol2': [4, 5, 6, 7, 8], 'kol3': [7, 8, 9, 10, 11]}
df = pd.DataFrame(data=d)
print("Original DataFrame")
print(df)
print('Baris Untuk kolom2 dengan value 7')
print(df.loc[df['kol2'] == 7] )
```

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			
	koll	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			
	koll	kol2	kol3
3	4	7	0

Figure 3.16: Aplikasi Sederhana Menggunakan Pandas

3.11.2 Aplikasi Sederhana Menggunakan Numpy

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

Listing 3.2: Code Program Sederhana Numpy

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

Dari code diatas dapat dijelaskan perbarisnya sebagai berikut :

- Baris pertama, yaitu import numpy yang artinya kita akan mengimport librari 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]
```

Figure 3.17: Aplikasi Sederhana Menggunakan Numpy

3.11.3 Aplikasi Sederhana Menggunakan Matplotlib

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

Listing 3.3: Code Program Sederhana Matplotlib

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

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 :

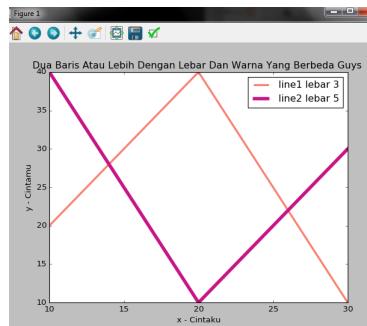


Figure 3.18: Aplikasi Sederhana Menggunakan Matplotlib

3.11.4 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_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:
...: #
```

Figure 3.19: Program Random Forest Tasya

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

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

Figure 3.20: Program Random Forest Tasya

- Output ini menampilkan jumlah baris dan kolom dari dataframe imgatt.

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

Figure 3.21: 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')
```

Figure 3.22: Program Random Forest Tasya

- Sama seperti output sebelumnya, imgatt2 head itu berfungsi untuk mengembalikan nilai atau value teratas dari dataframe imgatt2.

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

[5 rows x 312 columns]

Figure 3.23: Program Random Forest Tasya

- Output ini menampilkan jumlah baris dan kolom dari dataframe imgatt2

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

Figure 3.24: Program Random Forest Tasya

- Dan melakukan pivot yang mana imgid menjadi index yang artinya unik.

```
In [7]: imglabels = pd.read_csv("C09_200_2011/image_class_labels.txt",
...: sep=" ", header=None, names=['imgid',
...: 'label'])
...: # 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:
...: # i.e.
...: # it is a cimage_id <--> class_id
...: # where cimage_id and class_id correspond to the IDs in
...: # (mages.txt and classes.txt,
...: # respectively).
```

Figure 3.25: Program Random Forest Tasya

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

Figure 3.26: Program Random Forest Tasya

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

Figure 3.27: 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.
- Output diatas akan drop label yang didepan, dan menggunakan label yang paling belakang yang baru di join.
- Output berikut mengecek isinya. Ini mengecek 5 data teratas dari df att
- Output berikut mengecek isinya. Ini mengecek 5 data teratas dari df label
- Output diatas membagi menjadi dua bagian, 8000 row pertama sebagai data training sisanya sebagai data testing.
- Memanggil kelas RandomForestClassifier. max features diartikan sebagai berapa banyak kolom pada setiap tree disini kolom pada setiap tree adalah 50.

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

Figure 3.28: Program Random Forest Tasya

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

Figure 3.29: Program Random Forest Tasya

- Output ini melakukan fit untuk membangun random forest yang sudah ditentukan dengan maksimum fitur sebanya 50 untuk perpohonnya.
- Menampilkan hasil prediksi dari random forest sebelumnya.
- Menampilkan besaran akurasinya dari prediksi diatas atau Score perolehan dari klasifikasi.

3.11.5 Menjalankan Program Confusion Matrix

Berikut adalah output dari percobaan Confusion Matrix yang telah dilakukan

- Memetakan Random Forest ke dalam Confusion Matrix
- Melihat hasil dari gambar diatas
- Plotting Confusion Matrix dengan Matplotlib
- Set plot sumbunya sesuai dengan nama datanya dan membaca file classes txt
- Plot hasil perubahan label

3.11.6 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.
- Mencoba klasifikasi dengan SVM dengan dataset yang sama dan akan muncul akurasi prediksinya.

```
In [12]: df_att.head()
Out[12]:
   1    2    3    4    5    6    7  ...  306  307  308  309
310  311  312
imgid
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]
```

Figure 3.30: Program Random Forest Tasya

```
In [13]: df_label.head()
Out[13]:
label
imgid
10298  175
673    13
5793   99
8683   148
7830   134
```

Figure 3.31: Program Random Forest Tasya

3.11.7 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

3.11.8 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
- Output berikut merupakan hasil dari plotting komponen informasi agar dapat dibaca

3.12 Penanganan Error

HARI KEDUA TASYA WIENDHYRA 1164086

```
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']
```

Figure 3.32: Program Random Forest Tasya

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

Figure 3.33: Program Random Forest Tasya

3.12.1 Error Index

1. Berikut ini merupakan eror yang didapatkan saat menjalankan program diatas
 2. Pada gambar diatas kode erornya adalah IndentationError unexpected indent.
Eror ini terjadi karena adanya inkosisten pemberian indent di kode program.
 3. Solusi yang bisa dilakukan untuk mengatasi eror tersebut adalah sebagai berikut :
- Buka file code program dan lihat pada bagian erornya
 - Dapat dilihat pada baris kedua terdapat spasi dibagian depan, hilangkan atau hapus spasi tersebut seperti berikut
 - Save, kemudian ketika dijalankan eror akan teratasi

3.12.2 ANNISA FATHORONI/1164067

Penyelesaian Tugas Harian 6 (No. 1-8)

1. Aplikasi Sederhana Pandas dan Penjelasan Code

- Code Pandas:

```
import pandas as pd
d1 = {'a': 100, 'b': 200, 'c':300, 'd':400, 'e':800}
print("Original dictionary:")
print(d1)
new_series = pd.Series(d1)
print("Converted series:")
print(new_series)
```

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

Figure 3.34: Program Random Forest Tasya

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

Figure 3.35: Program Random Forest Tasya

- Penjelasan Code Baris 1 : Memasukkan modul pandas sebagai variabel pd.
- Penjelasan Code Baris 2 : Besaran nilai a, b , c, d dan e.
- Penjelasan Code Baris 3 : Output atau keluaran bacaan 'Original Dictionary'
- Penjelasan Code Baris 4 : Nilai dari D1 dikeluarkan.
- Penjelasan Code Baris 5 : Membuat series baru dengan patokan nilai dari variabel d1.
- Penjelasan Code Baris 6 : Mengeluarkan atau output dari Converted series.
- Penjelasan Code Baris 7 : Mengeluarkan atau output dari series baru yang telah dibuatkan.

- Hasil Eksekusi :

2. Aplikasi Sederhana Numpy dan Penjelasan Code.

- Code Numpy:

```
import numpy as np
x = np.arange(12, 38)
print(x)
```

- Penjelasan Code Baris 1 : Memasukkan modul numpy sebagai variabel np.
- Penjelasan Code Baris 2 : Nilai antara angka 12-38.
- Penjelasan Code Baris 3 :Mengeluarkan atau output dari proses.

- Hasil Eksekusi :

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

Figure 3.36: Program Random Forest Tasya

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

Figure 3.37: Program Confusion Matrix Tasya

3. Aplikasi Sederhana Matplotlib dan Penjelasan Code.

- Code Matplotlib:

```
import matplotlib.pyplot as plt
X = range(1, 50)
Y = [value * 3 for value in X]
plt.plot(X, Y)
plt.xlabel('x - axis')
plt.ylabel('y - axis')
plt.title('Draw a line.')
print(plt.axis())
plt.axis([0, 100, 0, 200])
plt.show()
```

- Penjelasan Code Baris 1 : Memasukkan modul Matplotlib sebagai variabel plt
- Penjelasan Code Baris 2 : Nilai X dengan ukuran 1-50.
- Penjelasan Code Baris 3 : Nilai Y dikalikan 3 untuk menjadi nilai di dalam X
- Penjelasan Code Baris 4 : Matplotlib dengan koordinat X dan Y
- Penjelasan Code Baris 5 : Judul atau keterangan label X
- Penjelasan Code Baris 6 : Judul atau keterangan label Y
- Penjelasan Code Baris 7 : Judul atau keterangan Diatas.
- Penjelasan Code Baris 8 : Print, mengeluarkan atau output.
- Penjelasan Code Baris 9 : Nilai plt antara 0-200.
- Penjelasan Code Baris 10 : Menampilkan matplotlib

- Hasil Eksekusi :

4. Program Aplikasi Random Forest dan Penjelasan Keluarannya.

```

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)

```

Figure 3.38: 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)

```

Figure 3.39: Program Confusion Matrix Tasya

- Code Random Forest 1:
 - Penjelasan :
Membaca dataset file txt.
- Code Random Forest 2:
 - Penjelasan :
Melihat sebagian data awal, dengan menggunakan listing.
- Code Random Forest 3:
 - Penjelasan :
Melihat jumlah data menggunakan listing.
- Code Random Forest 4:
 - Penjelasan :
Privot dataset.
- Code Random Forest 5:
 - Penjelasan :
Mengubah atribut menjadi kolom dengan menggunakan ptivot layaknya excel, lalu cek isi menggunakan perintah pada listing.
- Code Random Forest 6:

```

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_Chat
20         021.Eastern_Towhee
21         022.Chuck_will_Widow
22         023.Brandt_Cormorant

```

Figure 3.40: Program Confusion Matrix Tasya

```

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. 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]]

```

Figure 3.41: Program Confusion Matrix Tasya

- Penjelasan :

Mengubah atribut menjadi kolom dengan menggunakan ptivot layaknya excel, lalu cek isi menggunakan perintah pada listing.

- Code Random Forest 7:

- Penjelasan :

Me-load jawabannya dan melakukan pivot.

- Code Random Forest 8:

- Penjelasan :

Membaca dataset label file.

- Code Random Forest 9:

- Penjelasan :

Membaca dataset label file.

- Code Random Forest 10:

- Penjelasan :

Menggabungkan field dari dua file terpisah.

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

Figure 3.42: Program Decission Tree Tasya

```
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
```

Figure 3.43: Program SVM Tasya

- Code Random Forest 11:
 - Penjelasan :
Memisahkan dan memilih label.
- Code Random Forest 12:
 - Penjelasan :
Melihat isi masing-masing data frame.
- Code Random Forest 13:
 - Penjelasan :
Melihat isi masing-masing data frame.
- Code Random Forest 14:
 - Penjelasan :
Pembagian data training dan set.
- Code Random Forest 15:
 - Penjelasan :
Instalasi kelas Random Forest.
- Code Random Forest 16:
 - Penjelasan :
Fitting random forest dengan dataset training.
- Code Random Forest 17:
 - Penjelasan :
Melihat hasil prediksi.

```
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
```

Figure 3.44: 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)
```

Figure 3.45: Program Cross Validation Tasya

- Code Random Forest 18:

- Penjelasan :
- Score perolehan dari klasifikasi.

5. Program Aplikasi Confusion Matrix dan Penjelasan Keluarannya:

- Code Confusion Matrix 1:

- Penjelasan :
- Membuat confusion matrix.

- Code Confusion Matrix 2:

- Penjelasan :
- Array.

- Code Confusion Matrix 3:

- Penjelasan :
- Plotting Confucision Matrix.

- Code Confusion Matrix 4:

- Penjelasan :
- Membaca file classes.

- Code Confusion Matrix 5:

- Penjelasan :
- Plot hasil perubahan label.

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

- Code SVM :

```
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)
```

Figure 3.46: Program Cross Validation Tasya

```
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,
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)
```

Figure 3.47: Program Komponen Informasi Tasya

- Penjelasan :

Mencoba klasifikasi dengan decision tree dengan dataset yang sama

- Code Decision Tree :

- Penjelasan :

Mencoba klasifikasi dengan SVM dengan dataset yang sama.

7. Program Cross Validation dan Penjelasan Keluarannya.

- Code Cross Validation 1:

- Penjelasan :

Hasil Cross Validation random forest.

- Code Cross Validation 2 :

- Penjelasan :

Hasil Cross Validation Decission Tree

- Code Cross Validation 3:

- Penjelasan :

Hasil Cross Validation SVM

8. Program Pengamatan Komponen Informasi dan Penjelasan Keluarannya.

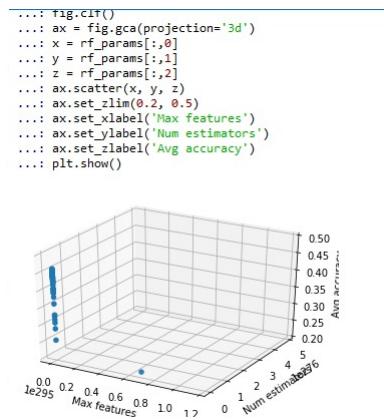


Figure 3.48: Program Komponen Informasi Tasya

```
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
```

Figure 3.49: Error Index

- Code Pengamatan Komponen Informasi 1 :

- Penjelasan :
- Melakukan pengamatan komponen informasi

9. Code Pengamatan Komponen Informasi 2:

- Penjelasan :
- Plot komponen informasi agar bisa dibaca

10. PENANGANAN ERROR.

- Kode Error serta jenis error:

```
FileNotFoundException: File b'data/CUB_200_2011/image_class_labels.txt' does
```

- Solusi penanganan error:
- Hapus direktori 'data' pada code dan pastikan satu folder.

- Screenshot Error :

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

Figure 3.50: File Codingan

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

Figure 3.51: Menghapus Spasi

3.12.3 Praktek Annisa Cahyani 1164066

Penyelesaian Tugas Harian 6 (No. 1-8)

1. Aplikasi Sederhana Pandas dan Penjelasan Code (Perbaris)

- Code Pandas:

- Penjelasan Code Baris 1 :
mengimport library pandas sebagai cahya
- Penjelasan Code Baris 2 :
kemudian membuat variable s dengan mengesekusi cahya.series
- Penjelasan Code Baris 3 :
kemudian melakukan pencetakan parameter (original data series)
- Penjelasan Code Baris 4 :
kemudian melakukan pencetakan variable s
- Penjelasan Code Baris 5 :
kemudian mengdefinisikan variables s untuk mengesekusi s,reindex
- Penjelasan Code Baris 6 :
melakukan pencetakan terhadap parameter (data series after chaging the order of index)
- Penjelasan Code Baris 7 :
kemudian melakukan lagi pencetakan variable s kembali

2. Aplikasi Sederhana Numpy dan Penjelasan Code (Perbaris)

- Code Numpy:

- Penjelasan Code Baris 1 :
pertama mengimport library numpy sebagai np

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

Figure 3.52: Eror Teratasi

```
In [54]: runfile('D:/Python-Artificial-Intelligence-Projects-for-Beginners-master/
Chapter02/contoh.py', wdir='D:/Python-Artificial-Intelligence-Projects-for-
Beginners-master/Chapter02')
Original dictionary:
{'a': 100, 'b': 200, 'c': 300, 'd': 400, 'e': 800}
Converted series:
a    100
b    200
c    300
d    400
e    800
dtype: int64
```

Figure 3.53: Hasil Pandas/Annisa Fathoroni

- Penjelasan Code Baris 2 :
kemudian melakukan pencetakan terhadap np.info dengan parameter variable np

3. Aplikasi Sederhana Matplotlib dan Penjelasan Code (Perbaris)

- Code Matplotlib:
 - Penjelasan Code Baris 1 :
yaitu untuk mengimport module matplotlib.pyplot sebagai plt
 - Penjelasan Code Baris 2 :
yaitu untuk membuat variable x dengan parameter
 - Penjelasan Code Baris 3 :
yaitu untuk membuat variable y dengan parameter
 - Penjelasan Code Baris 4 :
yaitu untuk mendefinisikan plt.plot
 - Penjelasan Code Baris 5 :
yaitu untuk mendefinisikan plt.xlabel
 - Penjelasan Code Baris 6 :
yaitu untuk mendefinisikan plt.ylabel
 - Penjelasan Code Baris 7 :
yaitu untuk mendefinisikan plt.title
 - Penjelasan Code Baris 8 :
yaitu untuk melakukan perintah snow

4. Program Aplikasi Random Forest dan Penjelasan Keluarannya :

```
In [55]: import numpy as np
...: x = np.arange(12, 38)
...: print(x)
[12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35
36 37]
```

Figure 3.54: Hasil Numpy/Annisa Fathoroni

```
In [59]: runfile('D:/Python-Artificial-Intelligence-Projects-for-Beginners-master/
Chapter02/gaguna.py', wdir='D:/Python-Artificial-Intelligence-Projects-for-
Beginners-master/Chapter02')
(-1.400000000000004, 51.4, -4.2, 154.2)
```

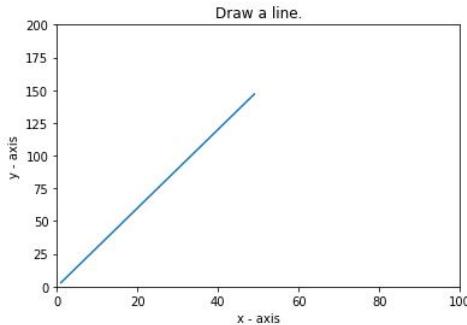


Figure 3.55: Hasil Matplotlib/Annisa Fathoroni

- Code Random Forest yang pertama adalah :
 - Penjelasan : Pada codingan tersebut akan menghasilkan suatu variabel baru adalah "imgatt" (yang telah kita buat sebelumnya) dengan menggunakan tipe dataframe yang berupa matrix 2 dimensi. Terdapat 3 kolom dan 3677856 baris data. Dapat kita check isi datanya karena datanya itu berupa data frame. Codingan tersebut akan "Membaca" dan "Menampilkan" data yang berasal dari file "image_attribute_labels.txt" tersebut.
- Code Random Forest yang kedua yaitu :
 - Penjelasan : Kemudian hasil codingan tersebut berfungsi untuk melihat suatu data awal dari data/dataset yang telah dibaca dan diproses. Maka untuk lebih jelasnya itu hanya untuk melihat isi data yang paling atas dari data yang ditampilkan pada console saat codingan tersebut dieksekusi.
- Code Random Forest yang ketiga adalah :
 - Penjelasan : Pada codingan yang diatas itu akan menghasilkan dan juga akan menampilkan dari jumlah data yang ada pada dataset yang telah dieksekusi. Jumlah data/ sizenya tersebut akan berupa baris dan kolom dari data

```

In [17]: import pandas as pd
...
...: # some lines have too many fields (?), so skip bad lines
...: imgatt = pd.read_csv("CUB_200_2011/attributes/image_attribute_labels.txt",
...: sep='\s+', header=None, error_bad_lines=False,
warn_bad_lines=False,
...: usecols=[0,1,2], names=['imgid', 'attid', 'present'])
...
...: # description from dataset README:
...
...: # The set of attribute labels as perceived by MTurkers for each image
...: # is contained in the file attributes/image_attribute_labels.txt, with
...: # each line corresponding to one image/attribute/worker triplet:
...
...: #
...: # <image_id> <attribute_id> <is_present> <certainty_id> <time>
...
...: #
...: # where <image_id>, <attribute_id>, <certainty_id> correspond to the IDs
...: # in images.txt, attributes/attributes.txt, and attributes/certainties.txt
...: # respectively. <is_present> is 0 or 1 (1 denotes that the attribute is
...: # present). <time> denotes the time spent by the MTurker in seconds.

In [18]: |

```

Figure 3.56: Random Forest 1/Annisa Fathoroni

```

In [18]: imgatt.head()
Out[18]:
      imgid  attid  present
0        1      1        0
1        1      2        0
2        1      3        0
3        1      4        0
4        1      5        1

In [19]: |

```

Figure 3.57: Random Forest 2/Annisa Fathoroni

yang telah ada.

- Code Random Forest yang keempat adalah:

– Penjelasan : Codingan ini akan menghasilkan tampilan berupa variabel baru yaitu ”imgatt2” . Yang dimana fungsinya itu untuk merubah atribut menjadi kolom dengan menggunakan pivot seperti excel. Yang membedakannya antara ”imgatt2” dan ”imgatt1” yaitu, jumlah dari data yang ditampilkan itu berbeda. Untuk imgatt1 kolomnya hanya ada 3 dan imgatt2 kolomnya diperbanyak (dijadikan) 312 sehingga dari segi pengelompokan baris datanya itu berbeda dan lebih efektif yang imgatt2.

- Code Random Forest yang kelima adalah :

– Penjelasan : Dari hasil codingan tersebut itu berfungsi untuk melihat data awal dari data/dataset yang telah dibaca dan diproses pada variabel” imgatt2 ”. Lebih jelasnya hanya untuk melihat isi yang berada paling atas dari data yang telah ditampilkan pada console saat codingan tersebut dieksekusi.

- Code Random Forest yang keenam adalah :

```
In [19]: imgatt.shape  
Out[19]: (3677856, 3)  
In [20]: |
```

Figure 3.58: Random Forest 3/Annisa Fathoroni

```
In [20]: imgatt2 = imgatt.pivot(index='imgid', columns='attid', values='present')  
In [21]: |
```

Figure 3.59: Random Forest 4/Annisa Fathoroni

- Penjelasan : Pada codingan diatas akan menghasilkan juga menampilkan beberapa dari jumlah data yang ada pada dataset yang telah dieksekusi pada variabel ” imgatt2 ”. Jumlah data/ sizenya tersebut akan berupa baris dan kolom dari data yang telah ada.
- Code Random Forest yang ketujuh :
 - Penjelasan : Dari hasil codingan tersebut akan menampilkan atau menunjukkan load dari jawabannya yang berisi ” apakah burung tersebut (subjek pada dataset) termasuk dalam spesies yang mana ?. Maka kedua kolom yang akan digunakan yaitu imgid dan label, lalu melakukan pivot yang dimana imgid menjadi index yang artinya itu unik sehubungan dengan dataset yang telah dieksekusi.
- Code Random Forest yang kedelapan :
 - Penjelasan : Dari hasil codingan tersebut itu berfungsi untuk melihat data awal dari data/dataset yang telah dibaca dan diproses pada variabel” imglables ”. Lebih jelasnya itu hanya untuk melihat isi yang berada paling atas dari data yang ditampilkan di console saat codingan tersebut dieksekusi.
- Code Random Forest yang kesembilan :
 - Penjelasan : Codingan yang dimaksud diatas itu akan menghasilkan dan juga menampilkan data dari jumlah data yang ada pada dataset yang telah dieksekusi pada variabel ” imglables ”. Jumlah data/ sizenya itu akan berupa baris dan kolom dari data yang telah ada.
- Code Random Forest yang kesepuluh adalah :
 - Penjelasan : Codingan tersebut dikarenakan isinya sama, maka dapat melakukan join antara dua data yang diesekusi (yaitu data dari

```

In [21]: imgatt2.head()
Out[21]:
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 0 1 0 0 ... 1 0 0 1 0 0 0
5 0 0 0 0 1 0 0 ... 0 0 0 0 0 0 0
[5 rows x 312 columns]

In [22]:

```

Figure 3.60: Random Forest 5/Annisa Fathoroni

```

In [22]: imgatt2.shape
Out[22]: (11788, 312)
In [23]: |

```

Figure 3.61: Random Forest 6/Annisa Fathoroni

imgatt2 dan imglabels), sehingga hasil yang akan didapatkan berupa data ciri dan data jawaban atau label sehingga dapat kita kategorikan/kelompokkan sebagai supervised learning. Maka perintah untuk menggabungkan kedua data tersebut, kemudian akan dilakukan pemisahan antara data set untuk training dan test pada dataset yang telah dieksekusi.

- Code Random Forest yang kesebelas adalah :
 - Penjelasan : pada codingan ini akan menghasilkan pemisahan dan pemilihan table. Pada codingan tersebut akan dilakukan kegiatan untuk drop label yang berada pada bagian depan kemudian menggunakan label yang berada pada posisi paling belakang kemudian selanjutnya melakukan join terhadap 2 variabel tersebut yang telah dieksekusi.
- Code Random Forest yang keduabelas adalah :
 - Penjelasan : Pada codingan ini akan menunjukkan hasil dari pengecekan isi variabel df.att bagian head (pada tampilan awal) dan dari dataset yang telah dieksekusi.
- Code Random Forest yang ketigabelas adalah :
 - Penjelasan : Pada codingan ini menunjukkan sebuah hasil pengecekan dari isi variabel df.label bagian head (pada tampilan awal) dari dataset yang telah dieksekusi.
- Code Random Forest yang keempatbelas adalah :

```

In [24]: imglabels = pd.read_csv("CUB_200_2011/image_class_labels.txt",
...:             sep=' ', header=None, names=['imgid', 'label'])
...:
...: imglabels = imglabels.set_index('imgid')
...:
...: # description from dataset README:
...: #
...: # The ground truth class labels (bird species labels) for each image are
...: contained
...: # in the file image_class_labels.txt, with each line corresponding to one
...: image:
...: #
...: # <image_id> <class_id>
...: #
...: # where <image_id> and <class_id> correspond to the IDs in images.txt and
...: classes.txt,
...: # respectively.
...:

In [25]: |

```

Figure 3.62: Random Forest 7/Annisa Fathoroni

```

In [25]: imglabels.head()
Out[25]:
      label
imgid
1        1
2        1
3        1
4        1
5        1

In [26]: |

```

Figure 3.63: Random Forest 8/Annisa Fathoroni

- Penjelasan : Dari hasil codingannya akan menunjukkan sebuah pembagian untuk data training dan data testing pada dataset. Kemudian akan melakukan pembagian data menjadi dua bagian yang dimana untuk row/baris pada data bagian pertama sebanyak 8000 dijadikan sebagai data training , kemudian 8000 data sisanya tersebut sebagai data testing pada pengeksekusianya.
- Code Random Forest yang kelimabelas adalah :
 - Penjelasan : Dari hasil codingan tersebut itu instalasi kelas pada random forest.yang dimana pada pemrosesannya tersebut, akan dilakukan pemanggilan kelas Random Forest Classifier (Klasifikasi Random Forest). Kemudian codingan yang terdapat max features yang diartikan sebagai berapa banyak kolom pada setiap tree yang telah dieksekusi pada dataset.
 - Code Random Forest yang keenambelas adalah :
 - Penjelasan : Hasil dari codingan tersebut akan menunjukkan fitting random forest dengan dataset training. Yang dimana hasil pengeksekusianya akan dilakukan fit untuk membangun random forest yang sudah ditentukan dengan maksimum fitur tersebut sebanyak 50 untuk perpohonnya sesuai dari dataset yang telah dieksekusi.

```
In [26]: imglabels.shape  
Out[26]: (11788, 1)  
In [27]: |
```

Figure 3.64: Random Forest 9/Annisa Fathoroni

```
In [27]: df = imgatt2.join(imglabels)  
...: df = df.sample(frac=1)  
In [28]: |
```

Figure 3.65: Random Forest 10/Annisa Fathoroni

- Code Random Forest yang ketujuhbelas adalah :
 - Penjelasan : Hasil dari codingan ini akan menunjukkan sebuah hasil dari prediksi dataset yang dieksekusi.
- Code Random Forest yang kedelapanbelas :
 - Penjelasan : Hasil dari codingan tersebut akan menunjukkan sebuah score perolehan dari klasifikasi dataset yang telah dieksekusi. Dari hasil besaran akurasi dari code yang telah dieksekusi sesuai dengan data yang digunakan.

5. Program Aplikasi Confusion Matrix dan Penjelasan Keluarannya :

- Code Confusion Matrix Pada Bagian Pertama:
 - Penjelasan : Dari hasil codingan diatas akan menunjukkan sebuah proses dari pembuatan confusion matrix yang dimana melakukan pemetaan dan pemetakan confusion matrix terhadap data yang telah dieksekusi.
- Code Confusion Matrix Pada Bagian Kedua :
 - Penjelasan : Dari hasil codingan tersebut akan menunjukkan hasil dari pembuatan pemetakan data yang telah dilakukan sebelumnya yaitu code confusion mtrix 1. Dan hasilnya akan berupa matrix yang di-dalamnya terdapat angka dari hasil yang eksekusi confusion matrix itu sendiri. Kemudian memunculkan array dan dtype dari hasil pembuatan confusion matrix.
- Code Confusion Matrix Yang Ketiga Adalah :
 - Penjelasan : Dari codingan yang diatas melakukan sebuah plotting confusion matrix yang dimana akan dieksekusi dari beberapa parameter yang disesuaikan dengan data yang telah dieksekusi. Dan code ini

```
In [28]: df_att = df.iloc[:, :312]
...: df_label = df.iloc[:, 312:]
In [29]:
```

Figure 3.66: Random Forest 11/Annisa Fathoroni

```
In [29]: df_att.head()
Out[29]:
   1    2    3    4    5    6    7    ...   306   307   308   309   310   311   312
imgid
5937  0    0    0    0    0    0    1    ...    0    0    0    0    0    0    1
4484  0    0    0    0    0    0    1    ...    0    0    0    1    0    0    0
10023 1    0    0    0    0    0    0    ...    0    0    0    0    0    0    0
9900  0    0    0    0    0    0    1    ...    0    0    0    0    0    0    0
5923  0    0    0    0    0    0    0    ...    0    0    0    0    0    0    1
[5 rows x 312 columns]
```

Figure 3.67: Random Forest 12/Annisa Fathoroni

akan hanya melakukan proses plotting dan tidak akan menunjukkan hasil plotting confusion matrixnya.

- Code Confusion Matrix Yang Keempat Adalah :
 - Penjelasan : Dari codingan tersebut menunjukkan hasil dari sebuah pembacaan file classes.txt yang telah dieksekusi. Kemudian melakukan plotting terhadap sumbu sesuai dengan nama data dan dilakukan penyetingan (set) sehingga akan memberikan hasil yang seperti pada gambar.
- Code Confusion Matrix Yang Keempat Adalah :
 - Penjelasan : Dari codingan ini akan menghasilkan plot dari hasil perubahan label. Yang dimana adalah proses plot yang melakukan akan merubah label dari sebuah classes / data yang telah dieksekusi.

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

- Code SVM adalah:
 - Penjelasan : Dari codingan ini akan menunjukkan klasifikasi dengan decision tree dengan dataset yang digunakan sama. Tentunya hanya pada pemrosesan decision tree yang digunakan dan menghasilkan output dari pengklasifikasi tersebut.
- Code Decision Tree adalah :
 - Penjelasan : Dari Codingan ini akan menunjukkan haasil klasifikasi yang sama dengan SVM dengan dataset yang sama. Tentunya pada pemrosesan yang SVM digunakan dan menghasilkan akan output dari pengklasifikasi tersebut.

```
In [30]: df_label.head()
Out[30]:
   label
  imgid
5937      102
4484       77
10023      171
9900      169
5923      102
```

Figure 3.68: Random Forest 13/Annisa Fathoroni

```
In [31]: df_train_att = df_att[:8000]
...: df_train_label = df_label[:8000]
...: df_test_att = df_att[8000:]
...: df_test_label = df_label[8000:]
...:
...: df_train_label = df_train_label['label']
...: df_test_label = df_test_label['label']
```

Figure 3.69: Random Forest 14/Annisa Fathoroni

7. Program Cross Validation dan Penjelasan Keluarannya :

- Code Cross Validation Yang Pertama Adalah :
 - Penjelasan : Dari codingan tersebut akan menampilkan hasil dari Cross Validation Random Forest. Dari pemrosesan yang dilakukan adalah pengecekan terhadap Cross Validation untuk Random Forest sesuai dengan data yang telah dieksekusi.
- Code Cross Validation Yang Kedua Adalah :
 - Penjelasan : Dari codingan tersebut akan menampilkan hasil dari Cross Validation Decision Tree. Pemrosesan yang dilakukan adalah pengecekan terhadap Cross Validation untuk Decision Tree sesuai dengan data yang telah dieksekusi.
- Code Cross Validation Yang Ketiga Adalah :
 - Penjelasan : Dari codingan tersebut akan menampilkan hasil dari Cross Validation SVM. Pemrosesan yang akan dilakukan adalah pengecekan terhadap Cross Validation untuk SVM sesuai dengan data yang telah dieksekusi.

8. Program Pengamatan Komponen Informasi dan Penjelasan Keluarannya :

- Code Pengamatan Komponen Informasi Yang Petama Adalah :
 - Penjelasan : Pada codingan ini menampilkan hasil dari pengamatan konponen informasi yang dimana pada pengeksekusianya dapat kita ketahui berapa banyak tree yang telah dibuat, dan berapa banyak

```
In [32]: from sklearn.ensemble import RandomForestClassifier  
...: clf = RandomForestClassifier(max_features=50, random_state=0, n_estimators=100)
```

Figure 3.70: Random Forest 15/Annisa Fathoroni

```
In [33]: clf.fit(df_train_att, df_train_label)  
Out[33]:  
RandomForestClassifier(bootstrap=True, class_weight=None, criterion='gini',  
max_depth=None, max_features=50, max_leaf_nodes=None,  
min_impurity_decrease=0.0, min_impurity_split=None,  
min_samples_leaf=1, min_samples_split=2,  
min_weight_fraction_leaf=0.0, n_estimators=100, n_jobs=None,  
oob_score=False, random_state=0, verbose=0, warm_start=False)
```

Figure 3.71: Random Forest 16/Annisa Fathoroni

atribut yang telah dipakai dan informasi yang lainnya terkait dengan kode yang digunakan.

- Code Pengamatan Komponen Informasi Yang Kedua Adalah :
 - Penjelasan : Pada codingan ini akan menampilkan plot dari beberapa komponen informasi agar dapat dibaca yang dimana pada pemrosesan plotting yang dilakukan pada informasi dengan menggunakan kode.

3.12.4 Penanganan Error

Penyelesaian Tugas Harian 6 (Penanganan Error)

- (a) Menyelesaikan dan Membahas Penanganan Yang Error :

- Error 1 :
 - Penjelasan Error Yang Pertama :
Error tersebut dikarena pada bagian codingan ”directory” file yang akan telah dieksekusi atau yang dipanggil tidak dapat.
 - Penyelesaian Error Yang Kedua :
 - i. Yang dilakukan pertama kali adalah memperhatikan file yang akan dieksekusi yaitu ”image_atribute_labels.txt”
 - ii. Kemudian di pastikan file tersebut telah berada dalam satu tempat atau directory dengan file bird-identifier.py yang digunakan sebagai codingan untuk melakukan pengeksekusian file
 - iii. Kemudian menghapus codingan dan sesuai seperti dengan contoh codingan berikut :

```
imgatt = pd.read_csv('image_attribute_labels.txt')
```
 - iv. Maka tidak akan terjadi error seperti itu lagi

```
In [34]: print(clf.predict(df_train_att.head()))
[102 77 171 169 102]
```

Figure 3.72: Random Forest 17/Annisa Fathoroni

```
In [35]: clf.score(df_test_att, df_test_label)
Out[35]: 0.4516895459345301
```

Figure 3.73: Random Forest 18/Annisa Fathoroni

```
In [36]: from sklearn.metrics import confusion_matrix
...: pred_labels = clf.predict(df_test_att)
...: cm = confusion_matrix(df_test_label, pred_labels)
```

Figure 3.74: Confusion Matrix 1/Annisa Fathoroni

```
In [37]: cm
Out[37]:
array([[ 4,  1,  2, ...,  0,  0,  0],
       [ 0, 17,  0, ...,  0,  1,  0],
       [ 1,  1,  5, ...,  0,  0,  0],
       ...,
       [ 0,  0,  0, ...,  3,  0,  0],
       [ 0,  0,  0, ...,  1,  6,  0],
       [ 0,  0,  0, ...,  0,  0, 18]], dtype=int64)
```

Figure 3.75: Confusion Matrix 2/Annisa Fathoroni

```
In [38]: import matplotlib.pyplot as plt
...: import itertools
...: def plot_confusion_matrix(cm, classes,
...:                          normalize=False,
...:                          title='Confusion matrix',
...:                          cmap=plt.cm.Blues):
...:
...:     """
...:     This function prints and plots the confusion matrix.
...:     Normalization can be applied by setting `normalize=True`.
...:
...:     if normalize:
...:         cm = cm.astype('float') / cm.sum(axis=1)[:, np.newaxis]
...:         print("Normalized confusion matrix")
...:     else:
...:         print('Confusion matrix, without normalization')
...:
...:     print(cm)
...:
...:     plt.imshow(cm, interpolation='nearest', cmap=cmap)
...:     plt.title(title)
...:     plt.colorbar()
...:     tick_marks = np.arange(len(classes))
...:     plt.xticks(tick_marks, classes, rotation=90)
...:     plt.yticks(tick_marks, classes)
...:
...:     fmt = '.2f' if normalize else 'd'
...:     thresh = cm.max() / 2.
```

Figure 3.76: Confusion Matrix 3/Annisa Fathoroni

```

In [40]: birds = pd.read_csv("CUB_200_2011/classes.txt",
...:             sep='\s+', header=None, usecols=[1],
...:             names=['birdname'])
...: birds = birds['birdname']
...: birds
Out[40]:
0          001.Black_footed_Albatross
1          002.Laysan_Albatross
2          003.Sooty_Albatross
3          004.Groove_billed_Ani
4          005.Crested_Auklet
5          006.Least_Auklet
6          007.Parakeet_Auklet
7          008.Rhinoceros_Auklet
8          009.Brewer_Blackbird
9          010.Red_winged_Blackbird
10         011.Rusty_Blackbird
11         012.Yellow_headed_Blackbird
12         013.Bobolink
13         014.Indigo_Bunting
14         015.Lazuli_Bunting
15         016.Painted_Bunting
16         017.Cardinal
17         018.Spotted_Catbird
18         019.Gray_Catbird
19         020.Yellow_breasted_Chat
20         021.Eastern_Towhee

```

Figure 3.77: Confusion Matrix 4/Annisa Fathoroni

```

In [41]: 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
[[0.24 0.06 0.12 ... 0. 0. 0. ]
 [0. 0.61 0. ... 0. 0.04 0. ]
 [0.06 0.06 0.28 ... 0. 0. 0. ]
 ...
 [0. 0. 0. ... 0.21 0. 0. ]
 [0. 0. 0. ... 0.05 0.27 0. ]
 [0. 0. 0. ... 0. 0. 0.95]]

```

Figure 3.78: Confusion Matrix 5/Annisa Fathoroni

```

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

```

Figure 3.79: SVM/Annisa Fathoroni

```

In [44]: from sklearn import tree
...: clftree = tree.DecisionTreeClassifier()
...: clftree.fit(df_train_att, df_train_label)
...: clftree.score(df_test_att, df_test_label)
Out[44]: 0.262671594508975

```

Figure 3.80: Decision Tree/Annisa Fathoroni

```

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

```

Figure 3.81: Cross Validation 1/Annisa Fathoroni

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

Figure 3.82: Cross Validation 2/Annisa Fathoroni

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

Figure 3.83: Cross Validation 3/Annisa Fathoroni

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

Figure 3.84: Pengamatan Komponen Informasi 1/Annisa Fathoroni

```
In [50]: import matplotlib.pyplot as plt
...: from mpl_toolkits.mplot3d import Axes3D
...: from matplotlib import cm
...: fig = plt.figure()
...: fig.clf()
...: ax = fig.gca(projection='3d')
...: x = rf_params[:,0]
...: y = rf_params[:,1]
...: z = rf_params[:,2]
...: ax.scatter(x, y, z)
...: ax.set_zlim(0.2, 0.5)
...: ax.set_xlabel('Max features')
...: ax.set_ylabel('Num estimators')
...: ax.set_zlabel('Avg accuracy')
...: plt.show()
```

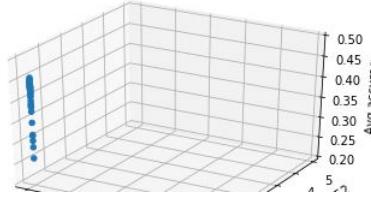


Figure 3.85: Pengamatan Komponen Informasi 2/Annisa Fathoroni

```
parser_f
    ...
    return _read(filepath_or_buffer, kwds)

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

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

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

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

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

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

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

Figure 3.86: Error/Annisa Fathoroni

```
In [2]: import pandas as pd
...:
...: # The file has two many fields (15), so skip bad lines
...: # Skip first line, then read in the rest
...: # The first line contains 'imgatt', 'attid', 'present', 'error', 'bad_lines_value', 'var', 'bad_lines_value',
...: # 'imgatt', 'attid', 'present', 'error', 'bad_lines_value', 'var', 'bad_lines_value', 'imgatt', 'attid', 'present'
...: # Description from dataset creator
...: # The set of attribute labels as perceived by lifelines for each image
...: # imgatt is the image identifier, attid is the attribute identifier, present is the value of the attribute
...: # each line corresponding to one image/attribute/observer triplet
...: # imgatt corresponds to one image/attribute/observer triplet
...: # error corresponds to one image/attribute/observer triplet
...: # bad_lines_value corresponds to one image/attribute/observer triplet
...: # var corresponds to one image/attribute/observer triplet
...: # present: either denotes the raw value by the observer or zero.
```

Figure 3.87: rf

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

Figure 3.88: rf2

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

Figure 3.89: rf3

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

Figure 3.90: rf4

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

Figure 3.91: rf5

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

Figure 3.92: rf6

```
In [8]: imglabels = pd.read_csv("image_label_labels.csv",
...:    sep=',', header=None, names=['imgid', 'label'])
...:
...: imglabels = imglabels.set_index('imgid')
...:
...: # description from dataset README
...: # This file contains the labels for each image. It contains
...: # 11788 rows. The first two columns (first species label) for each image are omitted.
...: # In the file image_label_labels.csv, each line corresponds to one image.
...: # A change in class_id means that
...: # 1 above image_id and classes_id correspond to the 2nd in images_id and classes_id,
...: # and so on respectively.
```

Figure 3.93: rf7

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

Figure 3.94: rf8

```
In [10]: imgatt2.shape
Out[10]: (11788, 312)
```

Figure 3.95: rf9

```
In [11]: df.att = imgatt2.iloc[:, 1:312]
...: df.label = imglabels['label']
```

Figure 3.96: rf10

```
In [12]: df.att = df.att + df.att.isnull() * 312
...: df.att = df.att.fillna(0)
```

Figure 3.97: rf11

```
In [13]: df.att.head()
Out[13]:
   1  2  3  4  5  6  7  ...  306 307 308 309 310 311 312
imgid  ...
5983  0  0  0  0  0  0  0  ...  0  0  0  0  0  0  1
6616  0  0  0  0  0  0  0  ...  0  0  0  1  0  0  0
588  0  0  0  0  0  0  0  ...  0  0  0  0  0  0  1
8979  0  0  0  0  0  0  1  ...  0  0  0  0  0  1  0
1770  0  0  0  0  0  0  0  ...  0  0  0  0  1  0  0
```

[5 rows x 312 columns]

Figure 3.98: rf12

```
In [14]: df_label.head()
Out[14]:
   imgid  label
5983     103
6616     114
588      11
8979     153
1770      32
```

Figure 3.99: rf13

```
In [15]: df_train_att = df.att[:8000]
...: df_train_label = df_label[:8000]
...: df_test_att = df.att[8000:]
...: df_test_label = df_label[8000:]
...: df_train_label = df_train_label['label']
...: df_test_label = df_test_label['label']
```

Figure 3.100: rf14

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

Figure 3.101: rf15

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

Figure 3.102: rf16

```
In [18]: print(clf.predict(df_train_att.head()))
[103 114 11 153 32]
```

Figure 3.103: rf17

```
In [19]: clf.score(df_test_att, df_test_label)
Out[19]: 0.4442977824709669
```

Figure 3.104: rf18

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

Figure 3.105: cm1

```
In [21]: cm
Out[21]:
array([[ 0,  0,  0, ...,  0,  0,  0],
       [ 0,  0,  0, ...,  0,  0,  0],
       [ 0,  0,  0, ...,  0,  0,  0],
       ...,
       [ 0,  0,  0, ...,  0,  0,  0],
       [ 0,  0,  0, ...,  0,  0,  0],
       [ 0,  0,  0, ...,  0,  0,  0]], dtype=int64)
```

Figure 3.106: cm2

```
In [22]: import numpy as np
...: from sklearn import metrics
...: # Fit the classifier
...: # Compute the confusion matrix
...: # Normalization can be applied by setting normalize=True.
...: # If normalize is set to True, the returned confusion matrix
...: # will be a float array where each element i,j is the ratio:
...: # precision(i,j) / sum_j(precision(i,j)), pp=precision(i,i)
...: # print("Normalized confusion matrix")
...: # print(cm_normalized)
...: # print("Unnormalized confusion matrix, without normalization")
...: # print(cm)
...: print(cm)
...: print("Unnormalized, normalization='unweighted', unweighted")
...: print("Normalized, normalization='true', true")
...: print("Normalized, normalization='all', all")
...: print("Normalized, normalization='macro', macro")
...: print("Normalized, normalization='micro', micro")
...: print("Normalized, normalization='weighted', weighted")
...: for i in range(len(cm)):
...:     for j in range(len(cm[i])):
...:         if normalize == 'true' or normalize == 'all' or normalize == 'weighted':
...:             cm_normalized[i][j] = cm[i][j] / np.sum(cm[i])
...:         else:
...:             cm_normalized[i][j] = cm[i][j]
...: print("Normalized confusion matrix")
...: print(cm_normalized)
```

Figure 3.107: cm3

```
In [23]: birds = pd.read_csv("classes.csv",
...:                      header=None, names=[1, 0])
...: birds = birds[0].values
Out[23]:
array([ 0,  1,  2,  3,  4,  5,  6,  7,  8,  9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29], dtype=int64)
```

Figure 3.108: cm4

```
In [24]: birds = pd.read_csv("classes.csv",
...:                      header=None, names=[1, 0])
...: birds = birds[0].values
Out[24]:
array([ 0,  1,  2,  3,  4,  5,  6,  7,  8,  9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29], dtype=int64)
```

Figure 3.109: cm5

```
In [25]: from sklearn import svm
...: # Create a SVC classifier
...: # Fit the classifier
...: # Compute the confusion matrix
...: # Normalization can be applied by setting normalize=True.
...: # If normalize is set to True, the returned confusion matrix
...: # will be a float array where each element i,j is the ratio:
...: # precision(i,j) / sum_j(precision(i,j)), pp=precision(i,i)
...: # print("Normalized confusion matrix")
...: # print(cm_normalized)
...: # print("Unnormalized confusion matrix, without normalization")
...: # print(cm)
...: print(cm)
...: print("Unnormalized, normalization='unweighted', unweighted")
...: print("Normalized, normalization='true', true")
...: print("Normalized, normalization='all', all")
...: print("Normalized, normalization='macro', macro")
...: print("Normalized, normalization='micro', micro")
...: print("Normalized, normalization='weighted', weighted")
...: for i in range(len(cm)):
...:     for j in range(len(cm[i])):
...:         if normalize == 'true' or normalize == 'all' or normalize == 'weighted':
...:             cm_normalized[i][j] = cm[i][j] / np.sum(cm[i])
...:         else:
...:             cm_normalized[i][j] = cm[i][j]
...: print("Normalized confusion matrix")
...: print(cm_normalized)
```

Figure 3.110: s1

```
In [49]: from sklearn import tree
.....
clfTree = tree.DecisionTreeClassifier()
.....
clfTree.score(dt['test_attr'], dt['test_label'])
Out[49]: 0.4733333333333333
```

Figure 3.111: dt

```
In [49]: from sklearn.model_selection import cross_val_score
.....
scores = cross_val_score(clfTree, dt['train_attr'], dt['train_label'], cv=5)
.....
print("Accuracy: %0.2f (+/- %0.2f)" % (scores.mean(), scores.std() * 2))
accuracy: 0.48 (+/- 0.01)
```

Figure 3.112: cv1

```
In [49]: crossVal = cross_val_score(clfTree, dt['train_attr'], dt['train_label'], cv=5)
.....
print("Accuracy: %0.2f (+/- %0.2f)" % (crossVal.mean(), crossVal.std()))
accuracy: 0.26 (+/- 0.02)
```

Figure 3.113: cv2

```
C:\ProgramData\Anaconda\lib\site-packages\sklearn\tree\__init__.py:196: FutureWarning:
The default value of gamma will change from 'auto' to 'scale' in version 0.22 to avoid this warning.
  "gamma": "auto", "min_weight_fraction_leaf": 0.01}, "max_depth": 5,
.....
C:\ProgramData\Anaconda\lib\site-packages\sklearn\tree\__init__.py:196: FutureWarning:
The default value of gamma will change from 'auto' to 'scale' in version 0.22 to avoid this warning.
  "gamma": "auto", "min_weight_fraction_leaf": 0.01}, "max_depth": 5,
.....
C:\ProgramData\Anaconda\lib\site-packages\sklearn\tree\__init__.py:196: FutureWarning:
The default value of gamma will change from 'auto' to 'scale' in version 0.22 to avoid this warning.
  "gamma": "auto", "min_weight_fraction_leaf": 0.01}, "max_depth": 5,
.....
C:\ProgramData\Anaconda\lib\site-packages\sklearn\tree\__init__.py:196: FutureWarning:
The default value of gamma will change from 'auto' to 'scale' in version 0.22 to avoid this warning.
  "gamma": "auto", "min_weight_fraction_leaf": 0.01}, "max_depth": 5,
.....
C:\ProgramData\Anaconda\lib\site-packages\sklearn\tree\__init__.py:196: FutureWarning:
The default value of gamma will change from 'auto' to 'scale' in version 0.22 to avoid this warning.
  "gamma": "auto", "min_weight_fraction_leaf": 0.01}, "max_depth": 5,
.....
accuracy: 0.31 (+/- 0.01)
```

Figure 3.114: cv3

```
In [49]: max_features_opts = np.arange(50, 100, 1)
.....
for n_estimators, max_features in zip(max_features_opts, max_features_opts):
    print("n_estimators: %d, max_features: %d, accuracy: %0.2f (+/- %0.2f)" % (n_estimators, max_features, np.mean(cross_val_score(dt['train_attr'], dt['train_label'], cv=5, n_estimators=n_estimators, max_features=max_features)), np.std(cross_val_score(dt['train_attr'], dt['train_label'], cv=5, n_estimators=n_estimators, max_features=max_features)) * 2))
.....
n_estimators: 5, max_features: 50, accuracy: 0.48 (+/- 0.01)
n_estimators: 5, max_features: 51, accuracy: 0.48 (+/- 0.01)
n_estimators: 5, max_features: 52, accuracy: 0.48 (+/- 0.01)
n_estimators: 5, max_features: 53, accuracy: 0.48 (+/- 0.01)
n_estimators: 5, max_features: 54, accuracy: 0.48 (+/- 0.01)
n_estimators: 5, max_features: 55, accuracy: 0.48 (+/- 0.01)
n_estimators: 5, max_features: 56, accuracy: 0.48 (+/- 0.01)
n_estimators: 5, max_features: 57, accuracy: 0.48 (+/- 0.01)
n_estimators: 5, max_features: 58, accuracy: 0.48 (+/- 0.01)
n_estimators: 5, max_features: 59, accuracy: 0.48 (+/- 0.01)
n_estimators: 5, max_features: 60, accuracy: 0.48 (+/- 0.01)
n_estimators: 5, max_features: 61, accuracy: 0.48 (+/- 0.01)
n_estimators: 5, max_features: 62, accuracy: 0.48 (+/- 0.01)
n_estimators: 5, max_features: 63, accuracy: 0.48 (+/- 0.01)
n_estimators: 5, max_features: 64, accuracy: 0.48 (+/- 0.01)
n_estimators: 5, max_features: 65, accuracy: 0.48 (+/- 0.01)
n_estimators: 5, max_features: 66, accuracy: 0.48 (+/- 0.01)
n_estimators: 5, max_features: 67, accuracy: 0.48 (+/- 0.01)
n_estimators: 5, max_features: 68, accuracy: 0.48 (+/- 0.01)
n_estimators: 5, max_features: 69, accuracy: 0.48 (+/- 0.01)
n_estimators: 5, max_features: 70, accuracy: 0.48 (+/- 0.01)
n_estimators: 5, max_features: 71, accuracy: 0.48 (+/- 0.01)
n_estimators: 5, max_features: 72, accuracy: 0.48 (+/- 0.01)
n_estimators: 5, max_features: 73, accuracy: 0.48 (+/- 0.01)
n_estimators: 5, max_features: 74, accuracy: 0.48 (+/- 0.01)
n_estimators: 5, max_features: 75, accuracy: 0.48 (+/- 0.01)
n_estimators: 5, max_features: 76, accuracy: 0.48 (+/- 0.01)
n_estimators: 5, max_features: 77, accuracy: 0.48 (+/- 0.01)
n_estimators: 5, max_features: 78, accuracy: 0.48 (+/- 0.01)
n_estimators: 5, max_features: 79, accuracy: 0.48 (+/- 0.01)
n_estimators: 5, max_features: 80, accuracy: 0.48 (+/- 0.01)
n_estimators: 5, max_features: 81, accuracy: 0.48 (+/- 0.01)
n_estimators: 5, max_features: 82, accuracy: 0.48 (+/- 0.01)
n_estimators: 5, max_features: 83, accuracy: 0.48 (+/- 0.01)
n_estimators: 5, max_features: 84, accuracy: 0.48 (+/- 0.01)
n_estimators: 5, max_features: 85, accuracy: 0.48 (+/- 0.01)
n_estimators: 5, max_features: 86, accuracy: 0.48 (+/- 0.01)
n_estimators: 5, max_features: 87, accuracy: 0.48 (+/- 0.01)
n_estimators: 5, max_features: 88, accuracy: 0.48 (+/- 0.01)
n_estimators: 5, max_features: 89, accuracy: 0.48 (+/- 0.01)
n_estimators: 5, max_features: 90, accuracy: 0.48 (+/- 0.01)
n_estimators: 5, max_features: 91, accuracy: 0.48 (+/- 0.01)
n_estimators: 5, max_features: 92, accuracy: 0.48 (+/- 0.01)
n_estimators: 5, max_features: 93, accuracy: 0.48 (+/- 0.01)
n_estimators: 5, max_features: 94, accuracy: 0.48 (+/- 0.01)
n_estimators: 5, max_features: 95, accuracy: 0.48 (+/- 0.01)
n_estimators: 5, max_features: 96, accuracy: 0.48 (+/- 0.01)
n_estimators: 5, max_features: 97, accuracy: 0.48 (+/- 0.01)
n_estimators: 5, max_features: 98, accuracy: 0.48 (+/- 0.01)
n_estimators: 5, max_features: 99, accuracy: 0.48 (+/- 0.01)
n_estimators: 5, max_features: 100, accuracy: 0.48 (+/- 0.01)
```

Figure 3.115: pki1

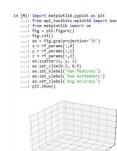


Figure 3.116: pki2

```
File "<ipython-input-36-db9347377f69>", line 8, in <module>
    usecols=[0,1,2], names=['imgid', 'attid', 'present'])
File "C:\ProgramData\Anaconda\lib\site-packages\pandas\io\parsers.py", line 678, in parser_f
    return _read(filepath_or_buffer, kwds)
File "C:\ProgramData\Anaconda\lib\site-packages\pandas\io\parsers.py", line 440, in _read
    parser = TextFileReader(filepath_or_buffer, **kwds)
File "C:\ProgramData\Anaconda\lib\site-packages\pandas\io\parsers.py", line 787, in __init__
    self._make_engine(self.f, **self.options)
File "C:\ProgramData\Anaconda\lib\site-packages\pandas\io\parsers.py", line 1014, in _make_engine
    self._engine = CParserWrapper(self.f, **self.options)
File "C:\ProgramData\Anaconda\lib\site-packages\pandas\io\parsers.py", line 1708, in __init__
    self._reader = parsers.TextReader(src, **kwds)
File "pandas\_libs\parsers.pyx", line 304, in pandas._libs.parsers.TextReader.__cinit__
File "pandas\_libs\parsers.pyx", line 695, in pandas._libs.parsers.TextReader._setup_parser_source
FileNotFoundError: File b'data/CUB_200_2011/attributes/image_attribute_labels.txt' does not exist
```

Figure 3.117: error1

Chapter 4

Experiment and Result

brief of experiment and result. Chapter 4

4.1 Teori

4.2 Tasya Wiendhyra/1164086

Hari Pertama Minggu Keempat

4.2.1 Klasifikasi Teks

4.2.1.1 Pengertian Dan Ilustrasi

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

Figure 4.1: Klasifikasi Teks Tasya

4.2.2 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 warna 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.



Figure 4.2: Klasifikasi Bunga Berwana Ungu Tasya

4.2.3 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 terbut, apakah masuk kedalam spam atau tidak. dan contoh lainnya yaitu pada Caption. di-

mana 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
345	z13th1q4yzihf1bl23qzxjpjeuljryd	2014-11-14T13:27:52	How can this have 2 billion views when there's...	0
346	z13fcn1wfpb5e51xe04chdxakpgzhyaxzo0k	2014-11-14T13:28:08	I don't know why I'm watching this in 2014	0
347	z130zd5b3ltudkoe04cbceohojxuzppvbg	2015-05-23T13:04:32	subscribe to me for call of duty vids and give...	1
348	z12he50arvkviv5u04ctawgzkjfsjc4	2015-06-05T14:14:48	hi guys please my android photo editor download	1
349	z13vhvu54u3ewpp5h04ccb4zuoardrmjlyk0k	2015-06-05T18:05:16	The first billion viewed this because they tho...	0

Figure 4.3: Klasifikasi Comment Spam Di Youtube Tasya

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

Itu merupakan presentase keakurasi prediksi yang dilakukan pada saat testing menggunakan label pada dataset yang digunakan. Score merupakan mendefinisikan aturan evaluasi model. Maka pada saat dijalankan akan muncuk persentase tersebut yang menunjukan keakurasi 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% .

4.2.5 Bag of Words

4.2.5.1 Pengertian Dan Ilustrasi

Merupakan representasi teks yang menggambarkan kemunculan kata-kata dalam dokumen. ePngelompokan kata kata kedalam perhitunga, berapakah 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

Figure 4.4: Bag of Words Tasya

4.2.6 TF-IDF

4.2.6.1 Pengertian Dan Ilustrasi

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

Figure 4.5: Contoh TF-IDF Tasya

4.3 Annisa Fathoroni/1164067

4.3.1 Teori

Penjelasan Tugas Harian 7 (No 1-6)

1. Pengertian Klasifikasi Teks Dan Ilustrasi Gambar

- Pengertian

Klasifikasi teks digunakan untuk mengelompokkan teks ke dalam grup yang terorganisir. Teks dianalisis oleh model dan kemudian tag yang sesuai diterapkan berdasarkan konten. Model pembelajaran mesin yang dapat secara otomatis menerapkan tag untuk klasifikasi dikenal sebagai pengklasifikasi.

- Ilustrasi Gambar

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



Figure 4.6: Klasifikasi Text - Annisa Fathoroni

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

- Penjelasan

Untuk klasifikasi bunga tidak dapat menggunakan machine learning dikarenakan memiliki masalah input yang serupa namun output yang berbeda, biasanya output / error ini disebut dengan istilah 'noise'. Noise sendiri merupakan output yang disimpan/ditangkap ataupun direkam bukan seperti seharusnya (keluaran yang diinginkan). Apabila diberikan contoh, maka mari kita misalkan sebuah contoh. Contohnya yaitu kita berasumsi secara implisit bahwa klarifikasi bunga kita sudah tepat seperti kita ahli tanaman. Namun terjadi masih terjadi kesalahan. Selain itu, selalu ada peluang untuk memperkenalkan kesalahan saat merekam data. Maka harus dilakukan penelitian yang lebih rinci sehingga tidak menimbulkan 'noise' itu sendiri.

- Ilustrasi Gambar

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

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

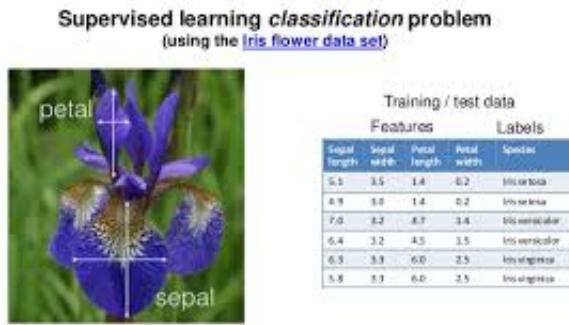


Figure 4.7: Klasifikasi Bunga Machine Learning - Annisa Fathoroni

- Penjelasan

Kita ambil sebuah kasus yang semua orang telah ketahui dan juga pahami. Kasus tersebut yaitu perekendasian video dari pencarian menggunakan "text / kata" di Youtube. Pada saat menggunakan Youtube terdapat Machine Learning yang bekerja dan memproses perintah ataupun aktivitas tersebut, dimana akan memfilter secara otomatis video yang disesuaikan dengan "keyword" yang kita masukkan sehingga memberikan keluaran video dengan keyword yang benar. Adapula pada saat kita sedang menonton video di YouTube, pada bagian sebelah kanan (tampilan Youtube) terdapat 'Up Next' yang menampilkan beberapa video serupa yang sedang ditonton. Dan ketika mengklik salah satu video dari baris tersebut, maka YouTube akan mengingatnya dan menggunakan kata yang tertera sebagai referensi kembali sehingga akan memberikan kemudahan pada pencarian yang lannya, Dan disitulah mesin belajar sendiri dan menyimpan data secara berkala sehingga berkembang.

- Ilustrasi Gambar

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

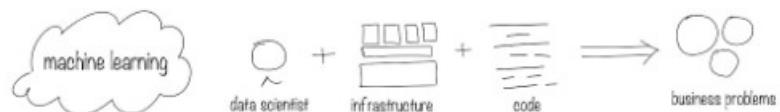


Figure 4.8: Teknik Pembelajaran Mesin - Annisa Fathoroni

4. Vektorisasi Data

- Penjelasan

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

Array angka floating point.

Mewakili data:

- teks
- audio
- gambar

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

5. Pengertian Bag Of Words Dan Ilustrasi Gambar

- Penjelasan

Bag-of-words ialah sebuah gambaran sederhana digunakan dalam pengolahan bahasa alami dan pencarian informasi. Dikenal sebagai model ruang vektor. Pada model ini, tiap kalimat dalam dokumen digambarkan sebagai token, mengabaikan tata bahasa dan bahkan urutan kata namun menghitung frekuensi kejadian atau kemunculan kata dari dokumen.

- Ilustrasi Gambar

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

6. Pengertian TF-IDF Dan Ilustrasi Gambar

- Penjelasan

TF-IDF merupakan metode untuk menghitung bobot setiap kata yang paling umum digunakan pada information retrieval. Metode ini juga terkenal efisien, mudah dan memiliki hasil yang akurat. Metode ini akan menghitung nilai Term Frequency (TF) dan Inverse Document Frequency (IDF) pada setiap token (kata) di setiap dokumen dalam korpus

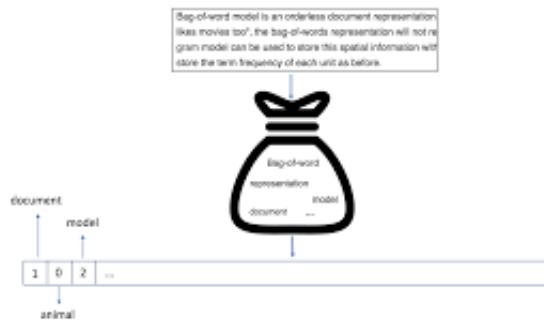


Figure 4.9: Bag of Words - Annisa Fathoroni

- Ilustrasi Gambar

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

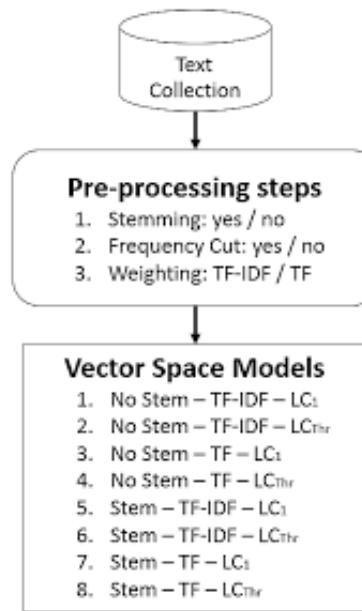


Figure 4.10: TF IDF - Annisa Fathoroni

4.4 Annisa Cahyani-1164066

4.4.1 Teori

Penjelasan Tugas Harian ke- 7 (No 1-6)

1. Pengertian Klasifikasi Teks Dan Ilustrasi Gambar

- Pengertian Klasifikasi Teks

Klasifikasi teks atau kategorisasi teks merupakan suatu proses yang secara otomatis menempatkan dokumen teks tersebut ke dalam suatu kategori yang berdasarkan dengan isi dari teks tersebut proses pemberian tag atau kategori pada teks harus sesuai dengan isinya.

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

- Mengapa Klasifikasi Bunga Tidak Bisa Menggunakan Machine Learning

Mengapa untuk pengklasifikasi pada bunga tidak dapat menggunakan machine learning karena memiliki masalah input yang serupa tetapi menggunakan output yang berbeda, biasanya output atau error ini biasa disebut dengan istilah 'noise'. Noise adalah output yang disimpan atau yang ditangkap ataupun yang direkam bukan seperti seharusnya (keluaran yang diinginkan).

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

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

Yaitu ambil sebuah kasus yang dimana semua orang telah mengetahui dan juga memahaminya. Kasus yang dimaksud yaitu perekondasian video dari sebuah pencarian dengan menggunakan "text atau kata" pada Youtube. Yang dimana pada saat kita menggunakan Youtube terdapat Machine Learning yang akan bekerja dan mulai memproses perintah ataupun aktivitas tersebut, yang dimana akan memfilter secara otomatis video-video yang telah disesuaikan dengan "keyword" yang telah kita masukkan sehingga dapat memberikan keluaran video dengan keyword yang sesuai.

4. Vektorisasi Data

- Maksud Dari Vektorisasi Data

Pada pembagian atau pemecahan data, yang dilakukan adalah perhitungan. Vektorisasi juga dapat diartikan dengan setiap data yang telah mungkin dipetakan ke integer yang tertentu. Apabila memiliki array yang cukup besar maka setiap kata atau data yang ada dan cocok dengan slot unik

dalam array maka nilai yang ada pada indeks tersebut adalah nomor satu kali kata itu muncul.

5. Pengertian Bag Of Words Dan Ilustrasi Gambar

- Pengertian Bag Of Words

Bag of words adalah merupakan representasi penyederhanaan yang digunakan dalam suatu pemrosesan bahasa yang alami dan pengambilan suatu informasi. Dalamnya yaitu sebuah teks yang seperti kalimat atau dokumen yang mewakili sebagai tas (multiset) dari kata-katanya, mengabaikan tata bahasa dan bahkan urutan kata tetapi tetap menjaga multiplisitas.

6. Pengertian TF-IDF Dan Ilustrasi Gambar

- Pengertian TF-IDF

TF-IDF merupakan salah satu dari metode yang biasa digunakan untuk melakukan pembobotan kata sebelum melakukan suatu proses selanjutnya. Biasanya pembobotan kata ini akan diperlukan jika kita ingin melakukan sebuah pemrosesan Bahasa alami atau pembuatan system temu kembali informasi.

4.5 BAGIAN PRAKTEK

4.6 Tasya Wiendhyra /1164086

PRAKTEK HARI KEDUA MIGGU KEEMPAT

4.6.1 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', 'parental level of ed'])
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.read_csv untuk membuat datafarme di pandas dari file CSV yang tadi.
4. Mendefinisikan variabel dummy untuk mengubah data categorical menjadi integer. dibawah merupakan data sebelum di Dummy.

Index	gender	race/ethnicity	ntal level of educ:	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

Figure 4.11: 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.

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

Figure 4.12: Dataset Dummy Tasya

4.6.2 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 :

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

Figure 4.13: Split DataFrame Tasya

4.6.3 Vektorisasi Dan Klasifikasi Dari Data Youtube Eminem Dengan Decision Tree

1. Ini hasil dari impor dataset
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

```
In [43]: import pandas as pd
...: d = pd.read_csv("Youtube04-Eminem.csv")
In [44]: from sklearn.feature_extraction.text import CountVectorizer
...: vectorizer = CountVectorizer()
```

Figure 4.14: Dataset Youtube Eminem Tasya

```
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, 0, 0, 1, 0, 1, 0, 0, 1, 0, 1, 0, 1, 0, 0, 0,
       0, 1, 0, 0, 1, 0, 1, 1, 1, 1, 1, 0, 1, 0, 1, 1, 1, 0, 0, 0,
       1, 1, 0, 1, 1, 0, 0, 1, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1,
       0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 1, 0, 1, 0, 0, 1, 0, 0, 0, 1, 0,
       0, 0, 1, 0, 0, 0, 0, 0, 1, 1, 1, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 1, 1,
       0, 1, 0, 0, 1, 0, 0, 1, 0, 0, 1, 0, 0, 0, 1, 0, 0, 1, 0, 0, 1, 0, 0,
       0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 1, 0, 0, 1, 0, 0, 1, 0, 0,
```

```
In [54]: clf.score(d_test_att, d_test_label)
Out[54]: 0.9324324324324324
```

Figure 4.15: Dataset Youtube Eminem Tasya

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.

4.6.4 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
```

Figure 4.16: 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

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

Maksud dari codingan diatas yaitu, mengklasifikasikan Dataset Youtube Eminem dengan Decision Tree dengan melakukan prediksi menggunakan function test pada d test

```
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
```

Figure 4.17: Dataset Youtube Eminem Tasya

att, dan memberikan akurasi prediksi menggunakan prediksi score.

4.6.6 Plotting Confusion Matrix

Berikut adalah skrip 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]]

```

Figure 4.18: Confusion Matrix Tasya

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

4.6.7 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)
...:
...: skorrrata2=scores.mean()
...: skoresd=scores.std()

```

Figure 4.19: 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. Hssilnya 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, 0, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 1, 0, 0, 0,
       0, 1, 0, 0, 1, 0, 1, 1, 1, 1, 0, 1, 0, 1, 1, 1, 0, 0, 0, 0,
       1, 1, 0, 1, 0, 1, 1, 1, 0, 0, 0, 1, 0, 1, 0, 1, 0, 1, 1, 0, 0, 0, 1,
       0, 0, 1, 0, 0, 1, 1, 0, 0, 0, 1, 0, 1, 0, 1, 0, 0, 0, 1, 0, 1, 0, 1,
       0, 0, 1, 0, 0, 0, 0, 0, 1, 1, 1, 0, 0, 0, 1, 1, 0, 1, 1, 0, 1, 1,
       0, 1, 0, 0, 1, 0, 0, 1, 1, 0, 1, 0, 0, 1, 0, 1, 1, 1, 0, 1, 0, 1, 1,
       0, 1, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 1, 0, 1, 0, 0, 0, 1, 1, 0, 1, 1,
       0, 1, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 1, 0, 1, 0, 0, 0, 1, 1, 0, 1, 1,
       0, 1, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 1, 0, 1, 1, 1, 0, 1, 0, 1, 1,
       0, 1, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 1, 0, 1, 1, 1, 0, 1, 0, 1, 1,
       0, 1, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 1, 0, 1, 1, 1, 0, 1, 0, 1, 1], dtype=int64)

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

Figure 4.20: Hasil Cross Validation Tasya

4.6.8 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
```

Figure 4.21: 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

4.7 Penanganan Error

HARI KEDUA TASYA WIENDHYRA 1164086

4.7.1 Error Index

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

```
KeyError: 'student'
```

Figure 4.22: Error Key Tasya

2. Pada gambar diatas kode erornya adalah KeyEror. Eror ini terjadi karena keyword yang dimasukan tidak ada.
3. Solusi yang bisa dilakukan untuk mengatasi eror tersebut adalah sebagai berikut :

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

Figure 4.23: 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()
```

Figure 4.24: Error Key Tasya

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

4.7.2 Annisa Fathoroni/1164067

Pengerjaanya Tugas Harian 8 (No 1-8)

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

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

Figure 4.25: Error Key Tasya

- Penjelasan Code:

```
import pandas as pd
d = pd.read_csv('tictactoe.csv', sep=',')
df = pd.DataFrame(d, columns = ['top-left-square', 'top-middle-square',
                                 'top-right-square', 'middle-left-square',
                                 'middle-middle-square', 'middle-right-square',
                                 'bottom-left-square', 'bottom-middle-square',
                                 'bottom-right-square', 'Class'])

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

- (a) Baris 1 : Mengimport Library Pandas sebagai pd
- (b) Baris 2 : Membaca dataset ‘tictactoe’.
- (c) Baris 3 : Mengambil dataframe dari library pandas.
- (d) Baris 4 : Mengubah categorical menjadi integer.
- (e) Baris 5 : Menampilkan 5 data teratas.

- Hasil Dari Aplikasi Sederhana Code Pandas

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

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

Index	top-left-square	op-middle-square	top-right-square	middle-left-square	idle-middle-squ	middle-right-squa	ottom-left-squa	ttom-middle-squ	ottom-right-squa	Class
0	x	x	x	x	o	o	o	x	o	positive
1	x	x	x	x	o	o	o	b	b	positive
2	x	x	x	x	o	o	b	o	b	positive
3	x	x	x	x	o	o	b	o	b	positive
4	x	x	x	x	o	o	b	b	o	positive
5	x	x	x	x	o	b	o	o	b	positive
6	x	x	x	x	o	b	o	b	o	positive
7	x	x	x	x	o	b	b	o	o	positive
8	x	x	x	x	b	o	o	o	b	positive
9	x	x	x	x	b	o	o	b	o	positive
10	x	x	x	x	b	o	b	o	o	positive
11	x	x	x	o	x	o	x	o	o	positive
12	x	x	x	o	x	o	o	x	o	positive
13	x	x	x	o	x	o	o	o	x	positive
14	x	x	x	o	x	o	o	b	b	positive
15	x	x	x	o	x	o	b	o	b	positive
16	x	x	x	o	x	o	b	b	o	positive
17	x	x	x	o	x	b	o	o	b	positive
18	x	x	x	o	x	b	o	b	o	positive
19	x	x	x	o	x	b	o	o	o	positive
20	x	x	x	o	o	x	x	o	o	positive

Figure 4.26: Aplikasi Pandas-Annisa Fathoroni

```
In [61]: dummy = pd.get_dummies (df['Class'])
...: dummy.head()
Out[61]:
   positive
0         1
1         1
2         1
3         1
4         1
```

Figure 4.27: Aplikasi Pandas-Annisa Fathoroni

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

- (a) Baris 1 : Membuat variabel dengan 450 data awal untuk dijadikan data training
- (b) Baris 2 : Membuat variabel dengan 50 data sisanya untuk data testing (dimulai dari data ke 451 sampai akhir data)

- Hasil Dari Pemecahan Dataframe

d_test	DataFrame	(51, 10)	Column names: top-left-square, top-middle-square, top-right-square, mi ...
d_train	DataFrame	(450, 10)	Column names: top-left-square, top-middle-square, top-right-square, mi ...

Figure 4.28: Pemecahan Data Frame-Annisa Fathoroni

3. Vektorisasi dan klasifikasi dari Youtube Shakira dengan Decision Tree

```
import pandas as pd
d = pd.read_csv("Youtube05-Shakira.csv")
```

Penjelasan:

Index	top-left-square	op-middle-square	top-right-square	middle-left-square	idle-middle-squ	iddle-right-squa	ottom-left-squa	ttom-middle-squa	ttom-right-squa	Class
450	o	b	x	b	x	b	x	b	o	positive
451	o	b	x	b	o	x	x	o	x	positive
452	o	b	x	b	o	x	o	x	x	positive
453	o	b	x	b	o	x	b	b	x	positive
454	o	b	x	b	o	o	x	x	x	positive
455	o	b	x	b	b	x	o	b	x	positive
456	o	b	x	b	b	x	b	o	x	positive
457	o	b	o	x	x	x	x	o	b	positive
458	o	b	o	x	x	x	x	b	o	positive
459	o	b	o	x	x	x	o	x	b	positive
460	o	b	o	x	x	x	o	b	x	positive
461	o	b	o	x	x	x	b	x	o	positive
462	o	b	o	x	x	x	b	o	x	positive
463	o	b	o	x	x	x	b	b	b	positive
464	o	b	o	x	o	b	x	x	x	positive
465	o	b	o	x	b	o	x	x	x	positive
466	o	b	o	o	x	b	x	x	x	positive
467	o	b	o	o	b	x	x	x	x	positive
468	o	b	o	b	x	o	x	x	x	positive
469	o	b	o	b	o	x	x	x	x	positive
470	o	b	o	b	b	b	x	x	x	positive

Figure 4.29: Pemecahan Data Frame-Annisa Fathoroni

Index	top-left-square	op-middle-square	top-right-square	middle-left-square	idle-middle-squa	iddle-right-squa	ottom-left-squa	ttom-middle-squa	ttom-right-squa	Class
0	x	x	x	x	o	o	o	x	o	positive
1	x	x	x	x	o	o	o	o	x	positive
2	x	x	x	x	o	o	o	b	b	positive
3	x	x	x	x	o	o	b	o	b	positive
4	x	x	x	x	o	o	b	b	o	positive
5	x	x	x	x	o	b	o	o	b	positive
6	x	x	x	x	o	b	o	b	o	positive
7	x	x	x	x	o	b	b	o	o	positive
8	x	x	x	x	b	o	o	o	b	positive
9	x	x	x	x	b	o	o	b	o	positive
10	x	x	x	x	b	o	b	o	o	positive
11	x	x	x	o	x	o	x	o	o	positive
12	x	x	o	x	x	o	x	o	o	positive
13	x	x	o	x	x	o	o	o	x	positive
14	x	x	o	x	x	o	o	b	b	positive
15	x	x	o	x	x	o	b	o	b	positive
16	x	x	o	x	x	o	b	b	o	positive
17	x	x	o	x	b	o	o	b	b	positive
18	x	x	o	x	b	o	b	o	o	positive
19	x	x	o	x	b	b	b	o	o	positive
20	x	x	o	x	o	x	x	o	o	positive

Figure 4.30: Pemecahan Data Frame-Annisa Fathoroni

Mendefinisikan variabel untuk memanggil Decision Tree dan melakukan pengujian lalu digunakan untuk memprediksi dan hasilnya muncul dalam bentuk array. df score memunculkan akurasi prediksi yang dilakukan terhadap df.

- Hasil

```
In [71]: import pandas as pd  
...: d = pd.read_csv("Youtube05-Shakira.csv")
```

Figure 4.31: Vektorisasi dan Klasifikasi Decision Tree-Annisa Fathoroni

```
In [82]: clf.predict(d_test_att)  
Out[82]:  
array([1, 0, 1, 0, 0, 1, 1, 0, 1, 0, 1, 1, 0, 1, 0, 1, 0, 0, 0, 1,  
     1, 1, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 1, 1, 1, 0, 0, 1, 1, 1, 0,  
     1, 0, 1, 1, 1, 0, 1, 1, 0, 0, 0, 1, 1, 0, 0, 1, 0, 0, 1, 0, 0, 1,  
     1, 1, 0, 0], dtype=int64)  
  
In [83]: clf.score(d_test_att,d_test_label)  
Out[83]: 0.8285714285714286
```

Figure 4.32: Vektorisasi dan Klasifikasi Decision Tree-Annisa Fathoroni

4. Vektorisasi dan klasifikasi dari Youtube Shakira dengan SVM

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

Penjelasan:

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

- Hasil Dari Codingan Klasifikasi SVM

```
In [84]: from sklearn import svm  
...: clfsvm = svm.SVC()  
...: clfsvm.fit(d_train_att, d_train_label)  
...:  
...: clfsvm.score(d_test_att, d_test_label)  
C:\ProgramData\Anaconda3\lib\site-packages\sklearn\svm\base.py:196: FutureWarning:  
The default value of gamma will change from 'auto' to 'scale' in version 0.22 to  
account better for unscaled features. Set gamma explicitly to 'auto' or 'scale' to  
avoid this warning.  
    "avoid this warning.", FutureWarning)  
Out[84]: 0.5428571428571428
```

Figure 4.33: Vektorisasi Klasifikasi SVM-Annisa Fathoroni

5. Klasifikasi dari data vektorisasi dari nomor sebelumnya dari decision tree.

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

Penjelasan:

Mendefinisikan variabel untuk memanggil Decision Tree dan melakukan pengujian lalu digunakan untuk memprediksi dan hasilnya muncul dalam bentuk array. df score memunculkan akurasi prediksi yang dilakukan terhadap df.

- Hasil:

```
In [85]: from sklearn import tree
....: clf = tree.DecisionTreeClassifier()
....: clf = clf.fit(d_train_att, d_train_label)
....: clf.predict(d_test_att)
....: clf.score(d_test_att,d_test_label)
Out[85]: 0.9
```

Figure 4.34: Klasifikasi Data Vektorisasi dari sebelumnya-Annisa Fathoroni

6. Plot Confusion Matrix menggunakan Matplotlib

```
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')

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

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

```

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()

```

Penjelasan :

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

- Hasil Program Cross Validation

7. Program Cross Validation

```

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

```

Figure 4.35: Plot Confusion Matrix-Annisa Fathoroni

```

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()

```

- (a) Baris 1 : Mengimport cross validation dari sklearn
- (b) Baris 2 : Variabel scores akan melakukan cross validation pada variabel tersebut
- (c) Baris 3 : Variabel skorrata2 akan menghitung nilai rata-rata dari variabel scores menggunakan function mean
- (d) Baris 4 : Skoresd menghitung standar dari data yang diberikan

- Hasil Program Cross Validation

```

In [64]: from sklearn.model_selection import cross_val_score
...: scores = cross_val_score(clf,d_train_att,d_train_label,cv=5)
...:
...: skorrata2=scores.mean()
...: skoresd=scores.std()

```

Figure 4.36: Cross Validation-Annisa Fathoroni

8. Program Pengamatan Komponen Informasi

Penjelasan :

scores	float64	(5,)	[0.91803279 0.98333333 0.9 0.93333333 0.93220339]
skoresd	float64	1	0.027736198186314388
skorrata2	float64	1	0.9333805686764842

Figure 4.37: Cross Validation-Annisa Fathoroni

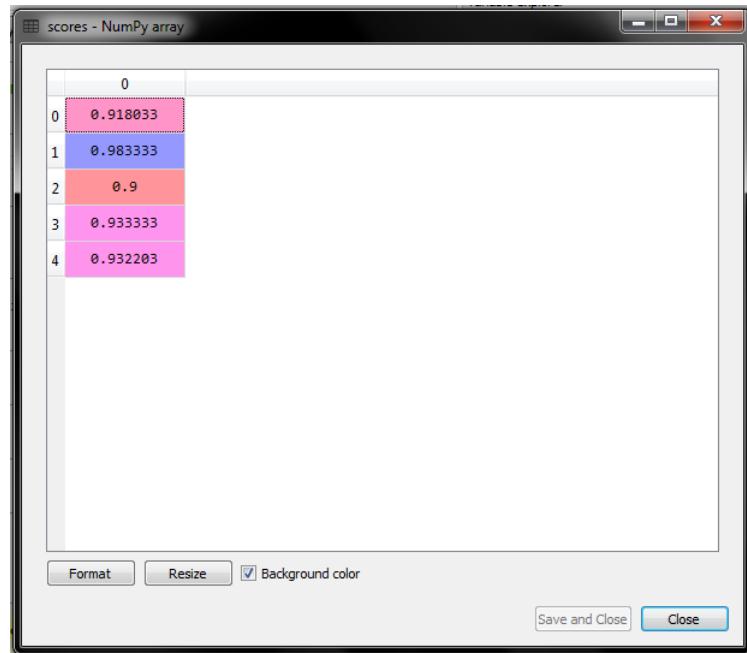


Figure 4.38: Cross Validation-Annisa Fathoroni

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

- Hasil:

```
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
```

Figure 4.39: Program Pengamatan-Annisa Fathoroni

9. PENANGANAN ERROR.

- Kode Error serta jenis error:

FileNotFoundException: File b'dataset/Youtube05-Shakira.csv' does not exist

- Solusi penanganan error:

Hapus dataset pada code dan pastikan satu folder.

- Screenshoot Error :

4.7.3 Praktek Annisa Cahyani 1164066

Pengerjaan Tugas Harian 8 (Nomor 1-8)

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

- Penjelasan Code Pada Pandas Sederhana

- (a) Pada baris yang pertama menjelaskan tentang mengimport Library Pandas sebagai pd

```

\parsers.py", line 1014, in _make_engine
    self._engine = CParserWrapper(self.f, **self.options)

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

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

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

FileNotFoundException: File b'dataset/Youtube05-Shakira.csv' does not
exist

```

Figure 4.40: Error/Annisa Fathoroni

- (b) Pada baris yang kedua itu Membuat variabel cahya yang dimana membaca dataset berupa format csv dengan separator.
- (c) Pada baris yang ketiga yaitu untuk menampilkan hasil dari variabel cahya (berupa jumlah/angka karena len)
- (d) Pada baris yang keempat yaitu untuk membuat variabel cahya yang dimana mengambil atau memanggil DataFrame dari library pd (pandas) dengan parameter variabel.
- Hasil Dari Aplikasi Sederhana Code Pandas Yaitu

Name	Type	Size	Value
cahya	Dataframe	(501, 9)	Column names: parents, has_nurs, form, children, housing, #finance, soc ...
dummy	Dataframe	(501, 4)	Column names: not_recom, priority, recommend, very_recom

Figure 4.41: cahyani1

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

- Pada dataframe yang pertama yaitu 450 row dari data pertama
- Pada dataframe yang kedua yaitu sebanyak 50 row dari data sisanya.
- Codingan Dan Penjelasannya yaitu :
 - (a) Pada baris yang pertama yaitu untuk membuat suatu variabel cahyatrain dari variabel cahyatrain dengan 450 data awal untuk dijadikan data training
 - (b) Pada baris yang kedua yaitu untuk membuat variabel cahyatest dari variabel cahyatrain dengan 50 data yang sisanya untuk data pada bagian testing yang dimana dimulai dari data ke 451 sampai akhir data

- Hasil Dari Pemecahan Dataframe

```
cahya_test DataFrame (50, 9) Column names: parents, has_nurs, form, children, housing,
finance, soc ...
cahya_train DataFrame (450, 9) Column names: parents, has_nurs, form, children, housing,
finance, soc ...
```

Figure 4.42: cahyani2

3. Vektorisasi Klasifikasi Data

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

Penjelasan tersebut sesuai dengan hasil.

- yang dimana hasil dari impor tree dari sklearn. Dan untuk mendefinisikan variabel clf berfungsi untuk memanggil pada bagian Decission Tree Classifier serta melakukan fit atau suatu pengujian.
- yang dimana akan menggunakan prediksi padabagian clf dengan menggunakan function predict untuk memprediksi suatu test. Dan hasilnya akan muncul dalam bentuk array.
- pada bagian clf score akan memunculkan sebuah akurasi suatu prediksi

- Hasil Program Yang Menggunakan Decision Tree

```
: 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)
: [[ 0, 0, 1, 0, 0, 1, 1, 1, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 1,
1, 0, 0, 1, 0, 1, 0, 1, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 1, 0, 0, 1, 1,
0, 0, 0, 0, 1, 1, 0, 1, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 1, 0, 0, 1, 0, 0,
0, 1, 0, 0, 0, 0], dtype=int64]
: clf.score(d_test_att, d_test_label)
: 0.92
```

Figure 4.43: cahyani3

4. Vektorisasi Dengan Mengguakan Klasifikasi SVM

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

Penjelasan adalah :

- yang pertama kita lakukan pengimportan module SVM dari sklearn

- (b) Kemudian melakukan fit dari d_train_att dan d_train_label atau disebut dengan sebuah pengujian

- Hasil Dari Klasifikasi SVM

```
from sklearn import svm
clfsvm = svm.SVC()
clfsvm.fit(d_train_att, d_train_label)
clfsvm.score(d_test_att, d_test_label)
: 0.96
```

Figure 4.44: Cahyani4

5. Vektorisasi Yang Menggunakan Klasifikasi Decision Tree

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

Penjelasannya adalah :

- Pada hasil impor tree dari sklearn tersebut. Dan melauakan pendefinisikan variabel clf untuk memanggil dari Decission Tree Classifier dan melakukan sebuah fit atau suatu pengujian.
- yang di dalamnya akan menjelaskan bahwa yang digunakan prediksi untuk clf dengan menggunakan function predict untuk memprediksi suatu test.
- kemudian untuk yang clf score akan memunculkan akurasi prediksi yang dilakukan terhadap clf

- Hasil Program Cross Validation

```
: from sklearn import tree
: clf = tree.DecisionTreeClassifier()
: clf = clf.fit(d_train_att, d_train_label)
: clf.predict(d_test_att)
: clf.score(d_test_att, d_test_label)
: 1.0
```

Figure 4.45: cahyani5

6. Plot Confusion Matrix

- Plot Confusion Matrix yaitu fungsi dari code tersebut untuk mencetak dan memplot Confussion Matrix. Yang dimana normalisasi dapat diterapkan dengan mengurnya seperti ‘normalize = True’.
- Hasil Plot Dari Confusion Matrix

```

1: import numpy as np
2: import matplotlib.pyplot as plt
3: import itertools
4: def plot_confusion_matrix(cm, classes,
5:                          normalize=False,
6:                          title='Confusion matrix',
7:                          cmap=plt.cm.Blues):
8:
9:     """
10:     This function prints and plots the confusion matrix.
11:     Normalization can be applied by setting `normalize=True`.
12:     """
13:     if normalize:
14:         cm = np.around(cm/cm.sum(axis=1)[:, np.newaxis]
15:    else:
16:        print('Confusion matrix, without normalization')
17:
18:    print(cm)
19:
20:    plt.imshow(cm, interpolation='nearest', cmap=cmap)
21:    plt.title(title)
22:    plt.colorbar()
23:    tick_marks = np.arange(len(classes))
24:    plt.xticks(tick_marks, classes, rotation=45)
25:    plt.yticks(tick_marks, classes)
26:
27:    thresh = cm.max() / 2.
28:
29:    for i, j in itertools.product(range(cm.shape[0]),
30:                                range(cm.shape[1])):
31:
32:        plt.text(j, i, cm[i, j],
33:                 horizontalalignment="center",
34:                 verticalalignment="center",
35:                 color="white" if cm[i, j] > thresh else "black")
36:
37:    plt.tight_layout()
38:    plt.ylabel('True label')
39:    plt.xlabel('Predicted label')
40:
```

Figure 4.46: cahyani6

7. Program Cross Validation

Pada bagian codingan ataupun hasil dari cross validation berikut, akan menjelaskan dimana variabel score tersebut akan melakukan sebuah cross validation pada variabel clf, d train att, dan train label.

- Hasil Program Cross Validation

```

1: from sklearn.model_selection import cross_val_score
2: scores = cross_val_score(clf,d_train_att,d_train_label, cv=5)
3: skorrate2=scores.mean()
4: skoredsd=scores.std()

```

Figure 4.47: cahyani7

8. Program Pengamatan Komponen Informasi

Penjelasannya adalah :

- Hasil Dari Program Cross Validation
- Pada bagian max featuresnya dari range 4 yaitu sampai 50
- Sedangkan Untuk n estimators dengan range 10 yaitu sampai 70
- Kemudian pada variable bagian rf params yang berisikan function np empty yang dimana akan membuat array baru yang berisikan sebuah tipe yang telah didefinisikan dengan random value
- Untuk mendefinisikan pada nilai i dimulai dari angka 0 yang dimana max features dan n estimators menggunakan sebuah klasifikasi randomforest-classifier yang menggunakan sebuah data prediksi.

```
In [15]: max_features_opts = range(4, 50, 1)
.... n_estimators_opts = range(6, 70, 4)
.... rf_params = np.empty((len(max_features_opts)*len(n_estimators_opts),4), float)
....
.... for max_features in max_features_opts:
....     for n_estimators in n_estimators_opts:
....         for i in range(len(max_features)):
....             rf_params[i] = np.append(max_features,n_estimators)
....             for j in range(4):
....                 rf_params[i,j] = np.append(rf_params[i,j],cross_val_score(c1t, fadila_train_att, fadila_train_label, cv=5))
....             scores = cross_val_score(c1t, fadila_train_att, fadila_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
....             print("Max Features: %d, num estimators: %d, accuracy: %0.2f (%0.2f)" % (max_features,
n_estimators, scores.mean(), scores.std() * 2))
```

Figure 4.48: cahyani8

4.7.4 Penanganan Yang Error Cahya

Untuk Menangani dan Mengatasi Error Pada Praktek yaitu

1. Error yang pertama :

- Code Yang Error yaitu :

```
import pandas as pd
d = pd.read_csv("dataset/Youtube04-Eminem.csv")
```

- Peringatan Yang Error :

```
FileNotFoundException: File b'dataset/Youtube04-Eminem.csv' does no
```

- Gambar Error yang pertama adalah :

```
File "pandas\_libs/parsers.pyx", line 384, in pandas._libs.parsers.TextReader.__init__
    File "pandas\_libs/parsers.pyx", line 699, in pandas._libs.parsers.TextReader._setup_parser_source
FileNotFoundException: File b'dataset/Youtube04-Eminem.csv' does not exist
```

Figure 4.49: errorcahya

- Cara Penanganan error adalah :

- (a) Pada bagian dari codingan yang akan dieksekusi sebenarnya itu untuk membaca dataset dari Youtube04-Eminem.csv
- (b) Namun, telah terdapat yang error dan hal tersebut disebabkan karena file codingan yang telah dieksekusi tidak berada pada folder yang sama dengan dataset eminem.
- (c) Sehingga dengan menganti codingan pada bagian tersebut, maka tidak akan terdapat error lagi.

Chapter 5

Conclusion

brief of conclusion

5.1 Annisa Fathoroni/1164067

5.1.1 Teori

Penjelasan Tugas Harian 9 (No 1-6)

1. Mengapa Kata-Kata Harus Di Lakukan Vektorisasi Dan Ilustrasi Gambar.

- Penjelasan:

Karenakan mesin hanya mampu membaca data dengan bentuk angka. Berdasarkan hal tersebut maka tentunya diperlukan vektorisasi kata atau bisa disebut dengan mengubah kata menjadi bentuk vektor agar mesin seolah-olah paham apa yang kita maksudkan dan dapat memproses aktifitas/perintah dengan benar. Selain alasan diatas, kata harus di vektorisasi untuk mengetahui persentase kata yang sering muncul dalam setiap kalimatnya, yang berguna untuk menentukan kata kunci.

- Ilustrasi Gambar

2. Mengapa Dimensi Dari Vektor Dataset Google Bisa Mencapai 300 Dan Ilustrasi Gambar.

- Penjelasan:

Karena pada masing-masing objek yang terdapat pada dataset akan memiliki identitasnya tersendiri. Apabila dicontohkan dengan penjelasan yang lebih rinci maka dilakukan perumpamaan sederhana. Misalnya untuk sebuah dataset google yang memiliki 3 buah objek yaitu berat, lebar, dan

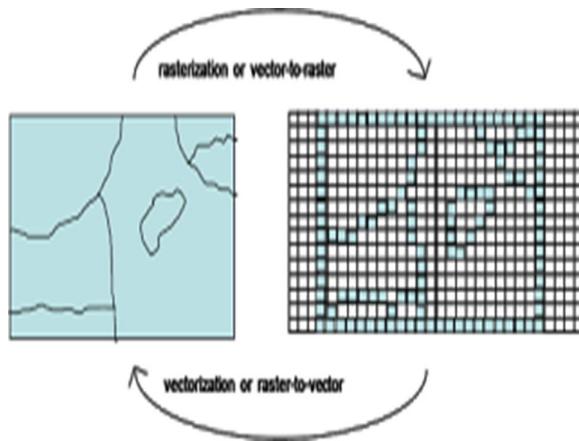


Figure 5.1: Vektorisasi - Annisa Fathoroni

tinggi. Kemudian dari masing-masing objek tersebut dilakukan perbandingan antara berat dan lebar beserta berat dan tinggi. Hasil yang didapatkan akan memiliki presentasi yang berbeda sehingga dapat diartikan bahwa mesin dapat membedakan objek yang hampir serupa namun tak sama.

- Ilustrasi Gambar

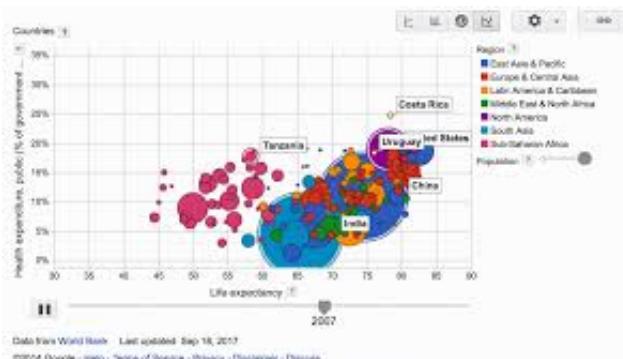


Figure 5.2: Dimensi Vektor Dataset - Annisa Fathoroni

3. Konsep Vektorisasi Untuk Kata Dan Ilustrasi Gambar.

- Penjelasan:

Konsep untuk vektorisasi kata sebenarnya sama dengan ketika dilakukan input suatu kata pada mesin pencarian. Kemudian untuk hasilnya akan mengeluarkan (berupa) referensi mengenai kata tersebut. Jadi data kata tersebut didapatkan dari hasil pengolahan pada kalimat-kalimat sebelumnya yang telah diolah. Contoh sederhananya pada kalimat berikut (Please

click the alarm icon for more notifications about my channel), pada kalimat tersebut terdapat konteks yakni channel, kata tersebut akan dijadikan data latih untuk mesin yang akan dipelajari dan diproses. Jadi ketika kita inputkan kta channel, maka mesin akan menampilkan keterkaitannya dengan kata tersebut sehingga akan lebih efisien dan lebih mudah.

- Ilustrasi Gambar

1. I enjoy flying.
2. I like NLP.
3. I like deep learning.

The resulting counts matrix will then be:

$$X = \begin{matrix} & \begin{matrix} I & like & enjoy & deep & learning & NLP & flying & . \end{matrix} \\ \begin{matrix} I \\ like \\ enjoy \\ deep \\ learning \\ NLP \\ flying \\ . \end{matrix} & \left[\begin{matrix} 0 & 2 & 1 & 0 & 0 & 0 & 0 & 0 \\ 2 & 0 & 0 & 1 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 \end{matrix} \right] \end{matrix}$$

Figure 5.3: Vektorisasi Untuk Kata - Annisa Fathoroni

4. Konsep Vektorisasi Untuk Dokumen Dan Ilustrasi Gambar.

- Penjelasan:

Untuk vektorisasi dokumen sebenarnya terbilang sama dengan konsep vektorisasi kata, yang membedakan hanya pada proses awalnya (pada eksekusi awal). Untuk vektorisasi dokumen ini, mesin akan membaca semua kalimat yang terdapat pada dokumen tersebut, kemudian kalimat yang terdapat pada dokumen tersebut akan di pecah menjadi kata-kata. Seperti itulah konsep vektorisasi dokumen.

- Ilustrasi Gambar

5. Pengertian Mean Dan Standar Deviasi Beserta Ilustrasi Gambar.

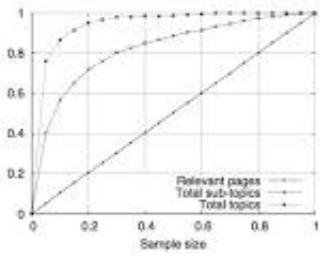


Figure 1: Impact of collection size on the fraction of relevant pages and subtopics with relevance.

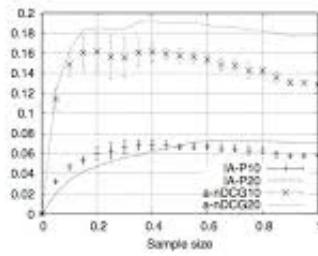


Figure 2: Impact of collection size on result diversity.

Figure 5.4: Vektorisasi Untuk Dokumen - Annisa Fathoroni

- Pengertian Mean:

Mean adalah nilai rata-rata dari beberapa buah data. Nilai mean dapat ditentukan dengan membagi jumlah data dengan banyaknya data. Mean (rata-rata) merupakan suatu ukuran pemusatan data. Mean suatu data juga merupakan statistik karena mampu menggambarkan bahwa data tersebut berada pada kisaran mean data tersebut. Mean tidak dapat digunakan sebagai ukuran pemusatan untuk jenis data nominal dan ordinal.

- Pengertian Standar Deviasi:

Standar Deviasi dan Varians Salah satu teknik statistik yg digunakan untuk menjelaskan homogenitas kelompok. Varians merupakan jumlah kuadrat semua deviasi nilai-nilai individual thd rata-rata kelompok. Sedangkan akar dari varians disebut dengan standar deviasi atau simpangan baku. Standar Deviasi dan Varians Simpangan baku merupakan variasi sebaran data. Semakin kecil nilai sebarannya berarti variasi nilai data makin sama Jika sebarannya bernilai 0, maka nilai semua datanya adalah sama. Semakin besar nilai sebarannya berarti data semakin bervariasi.

- Ilustrasi Gambar

6. Penjelasan Skip-gram Dan Ilustrasi Gambar

- Penjelasan:

Skip-Gram adalah kebalikannya, yaitu mencoba memprediksi vektor kata-kata yang ada di konteks diberikan vektor kata tertentu. Skip-Gram membuat sepasang kata target dan konteks sebagai sebuah instance sehingga Skip-Gram cenderung lebih baik ketika ukuran corpus sangat besar.

- Ilustrasi Gambar

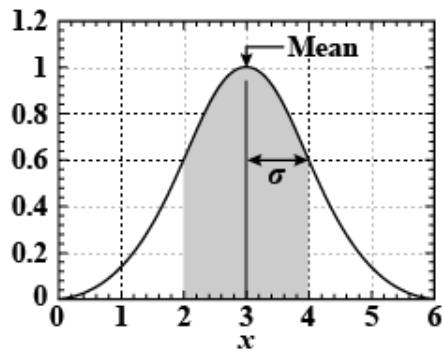


Figure 5.5: Mean - Annisa Fathoroni

$$s = \sqrt{\frac{\sum_{i=1}^n (x_i - \bar{x})^2}{n-1}}$$

$$s = \sqrt{\frac{n \sum_{i=1}^n x_i^2 - (\sum_{i=1}^n x_i)^2}{n(n-1)}}$$

Figure 5.6: Standar Deviasi - Annisa Fathoroni

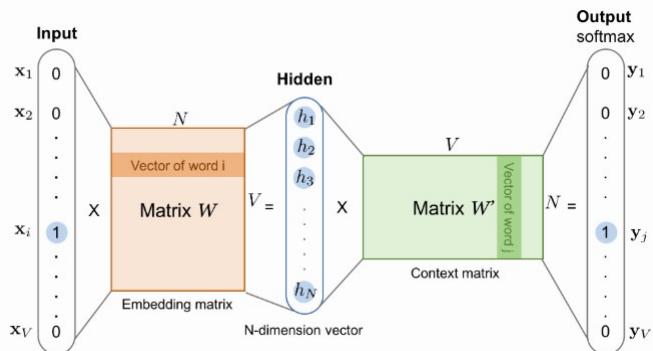


Figure 5.7: Skip Gram - Annisa Fathoroni

5.2 Tasya Wiendhyra /1164086

HARI PERTAMA MINGGU KELIMA

5.2.1 Kenapa Kata-Kata Harus Di Lakukan Vektorisasi Beserta Ilustrasi

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 mengonversi 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) .

5.2.2 Mengapa Dimensi Dari Vektor Dataset Google Bisa Sampai 300 Beserta Ilustrasi

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.

5.2.3 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 mengelompokan kalimat menjadi token atau membagi kata kata menjadi potongan kecil. Baru setelah itu dilakukan normalisasi untuk mengubah datanya menjadi angka.

Ilustrasinya, misalkan ada beberapa kalimat seperti berikut :

- There used to be Iron Age.

Kemudian didapatkan token seperti berikut "There", "was", "to", "be", "used", "Stone", "Bronze"

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]

5.2.4 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.

5.2.5 Mean Dan Standar Deviasi

5.2.5.1 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.

5.2.5.2 Contoh

1. Misalkan kita sudah menghitung tinggi anjing.

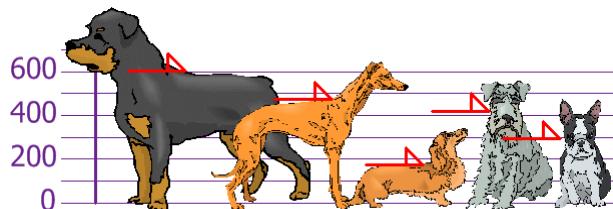


Figure 5.8: Contoh Mean dan Standar Deviasi

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.

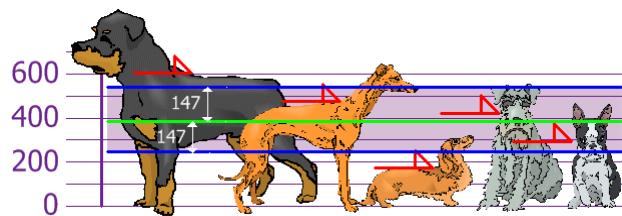


Figure 5.9: Contoh Mean dan Standar Deviasi

5.2.6 Skip-gram

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

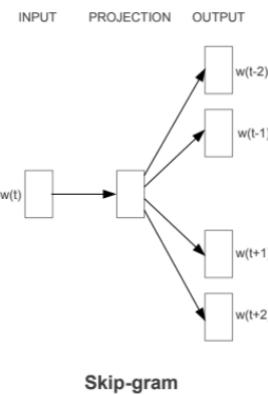


Figure 5.10: Contoh Skipgram

5.3 Annisa Cahyani-1164066

5.3.1 Teori

Penjelasan Tugas Harian 9 (No 1-6).

1. Mengapa Kata Harus Di Lakukan Vektorisasi Dan Ilustrasi Gambar
 - Mengapa kata ini harus dilakukan vektorisasi terlebih dahulu karena mesin tersebut hanya mampu atau dapat membaca data yang berupa angka. Sehingga diperlukannya suatu vektorisasi kata atau yang bisa disebut dengan mengubah suatu kata menjadi bentuk vektor agar mesin seolah paham apa yang telah kita maksud dan dapat memproses aktifitas atau perintah dengan betul.
2. Mengapa Dimensi Dari Vektor Dataset Google Bisa Mencapai 300 Dan Ilustrasi Serta Contoh Gambar
 - Konsep vektorisasi untuk kata yaitu sebuah dimensi dari suatu Vektor Dataset Google yang dapat mencapai 300 dikarenakan pada masing-masing objek yang telah terdapat pada dataset maka akan memiliki suatu identitasnya tersendiri.
3. Konsep Vektorisasi Untuk Kata Dan Ilustrasi Gambar
 - Penjelasan tetang vektorisasi untuk kata
Vektorisasi Kata adalah suatu hal yang sama dengan pada saat kita meng-inputkan suatu kata pada mesin pencari. Kemudian akan mengeluarkan hasil yang berupa suatu referensi mengenai kata tersebut. Sehingga hasil dari data kata tersebut akan didapatkan dari hasil pengolahan pada kalimat yang telah diolah sebelumnya.
4. Konsep Vektorisasi Untuk Dokumen Dan Ilustrasi Gambar
 - Penjelasan tentang vektorisasi untuk dokumen
Vektorisasi dokumen yaitu suatu data yang terstruktur karena jauh dari bentuk table baris dan kolom, sehingga Perlu pembentukan data yang terstruktur untuk mewakili dokumen. Maka dari itu kita harus menentukan features yang mewakilki seluruh kumpulan dokumen.
5. Pengertian Mean Dan Standar Deviasi Beserta Ilustrasi Gambar

- Penjelasan Mean :

Mean yaitu merupakan suatu nilai rata-rata dari suatu data. Sedangkan mean sendiri juga dapat dicari dengan menggunakan cara membagi jumlah data dengan banyaknya data, maka daritulah diperoleh nilai rata-rata dari suatu data yang dicari.

- Penjelasan Standar Deviasi :

standar deviasi merupakan sebuah teknik statistik yang dimanfaatkan digunakan untuk menjelaskan homogenitas kelompok ataupun yang dapat diartikan sebagai nilai statistic. Yang dimanfaatkan untuk menentukan bagaimana sebaran data pada dalam sampel, serta seberapa dekat dengan titik data individu ke mean atau ke rata-rata nilai sampel yang telah ada.

6. Penjelasan Skip-gram Dan Ilustrasi Gambar

- Penjelasan skip-gram

Skip-Gram adalah dibutuhkan setiap kata yang ada dalam fokus dan juga mengambil satu-per-satu kata yang ditentukan untuk diberikan setelah pelatihan akan memprediksi probabilitas untuk setiap kata untuk benar-benar muncul di jendela di sekitar kata fokus.

5.4 PRAKTEK PROGRAM

5.5 Tasya Wiendhyra/1164086

Praktek Hari Kedua Minggu Kelima

5.5.1 Mencoba Dataset

5.5.1.1 Vektor

- 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 ,
       0.00117100,
```

Figure 5.11: 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,
```

Figure 5.12: 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
```

Figure 5.13: 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,
```

Figure 5.14: Vektor Sick Tasya

- Vektor clear memiliki nilai identitas -2,44 dan tidak mendekati nilai dari vektor fall sehingga tidak dapat dijadikan dalam satu kategori
- Untuk vektor shine -0.12 tidak mendekati vektor manapun.
- 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.
- Vektor car nilainya 0.13 mendekati vektor love dan faith sehingga mungkin dapat dikategorikan dalam satu kategori.
- Vektor wash memiliki nilai 9.46 jauh dari vektor vektor lainnya.
- 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 [18]: gmodel['clear']
Out[18]:
array([-2.44140625e-04, -1.02050781e-01, -1.49414062e-01,
       -4.24804688e-02,
```

Figure 5.15: Vektor Clear Tasya

```
In [19]: gmodel['shine']
Out[19]:
array([-0.12402344,  0.25976562, -0.15917969, -0.27734375,
       0.38273435,  0.09960938,  0.39257812, -0.22949219, -0.18359375,  0.3671875,
```

Figure 5.16: Vektor Shine Tasya

5.5.1.2 Similariti

1. Lihat gambar berikut yang merupakan hasil prediksi similariti

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 kategorui yang sama misalnya dalam kategori percintaan. Mesin sudah mengetahui bahwa keduanya dapat dikategorikan sebagai percintaan.

5.5.2 Extract Words dan PermuteSentences

5.5.2.1 Extract Words

ExtractWords merupakan function untuk menambahkan, menghilangkan atau menghapuskan, hal hal yang tidak penting atau tidak perlu 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.

5.5.2.2 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 lenght. Yang kemudian mendefinisikan variabel req untuk lenght dan melakukan random choice yaitu pengocokan acak untuk kata car.

```
In [20]: gmodel['bag']
Out[20]:
array([-0.03515625,  0.15234375, -0.12492344,  0.13378906,
-0.11528125, -0.0133667 , -0.16113281,  0.14648438, -0.06839398,  0.140625
,
-0.06805059, -0.3946875 ,  0.20998094, -0.04545763, -0.2189375
```

Figure 5.17: Vektor Bag Tasya

```
In [21]: gmodel['car']
Out[21]:
array([ 0.13085958,  0.09842285,  0.03344727, -0.05853789,
0.04000396,
 1.4257812,  0.04931641, -0.16894531,  0.20898438,
0.11962691,
```

Figure 5.18: Vektor Car Tasya

```
In [22]: gmodel['wash']
Out[22]:
array([ 9.46044922e-03,  1.41681562e-01, -5.46875900e-02,
1.34765125e-01, -2.38281250e-01,  3.24218750e-01, -8.44726562e-02,
-1.29882812e-01,
```

Figure 5.19: Vektor Wash Tasya

```
In [23]: gmodel['motor']
Out[23]:
array([ 5.73730469e-02,  1.50398025e-01, -4.61425781e-02,
-1.32812500e-01, -2.39763025e-01, -1.77734375e-01,  3.68652344e-02,
```

Figure 5.20: Vektor Motor Tasya

```
In [19]: gmodel.similarity('love', 'faith')
Out[19]: 0.37953479345872814
In [20]: gmodel.similarity('love', 'fall')
Out[20]: 0.11425473111714524
In [21]: gmodel.similarity('love', 'sick')
Out[21]: 0.2665636147352396
In [22]: gmodel.similarity('love', 'clear')
Out[22]: 0.0658447631132513
In [23]: gmodel.similarity('love', 'shine')
Out[23]: 0.2098529863089267
In [24]: gmodel.similarity('love', 'reg')
Out[24]: 0.0753609968892862
In [25]: gmodel.similarity('love', 'car')
Out[25]: 0.0042139915245212
In [26]: gmodel.similarity('love', 'wash')
Out[26]: 0.0042139915245212
In [27]: gmodel.similarity('love', 'motor')
Out[27]: 0.08877587654154623
In [28]: gmodel.similarity('love', 'cycle')
Out[28]: 0.0566458891759813
```

Figure 5.21: Similariti Tasya

```
In [33]: import re
...: def extract_words(cantik):
...:     cantik = cantik.lower()
...:     cantik = re.sub("\$|^", "", cantik) #hpus python style
...:     for i in range(1, len(cantik)):
...:         cantik = re.findall("(\w+)", "", cantik)
...:     cantik = re.split("\s+", "", cantik)
...:     return cantik.split()
```

Figure 5.22: Extract Words Tasya

```
import random
class Permutedsentences(object):
    def __init__(self, lenght):
        self.lenght = lenght
    def __iter__(self):
        eq = list(self.lenght)
        random.shuffle(eq)
```

Figure 5.23: PermuteSentencesi Tasya

Chapter 6

Discussion

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

Chapter 7

Discussion

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

Chapter 8

Discussion

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

Chapter 9

Discussion

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

Chapter 10

Discussion

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

Chapter 11

Discussion

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

Chapter 12

Discussion

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

Chapter 13

Discussion

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

Chapter 14

Discussion

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

Appendix A

Form Penilaian Jurnal

gambar ?? dan ?? merupakan contoh bagaimana reviewer menilai jurnal kita.

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

Figure A.1: Form nilai bagian 1.

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

Figure A.2: form nilai bagian 2.

Appendix B

FAQ

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

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

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

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

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

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

M : Pa saya bingung D : Coba lihat standar alasan

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

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

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

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

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