

Pembelajaran Mesin

(Teori)

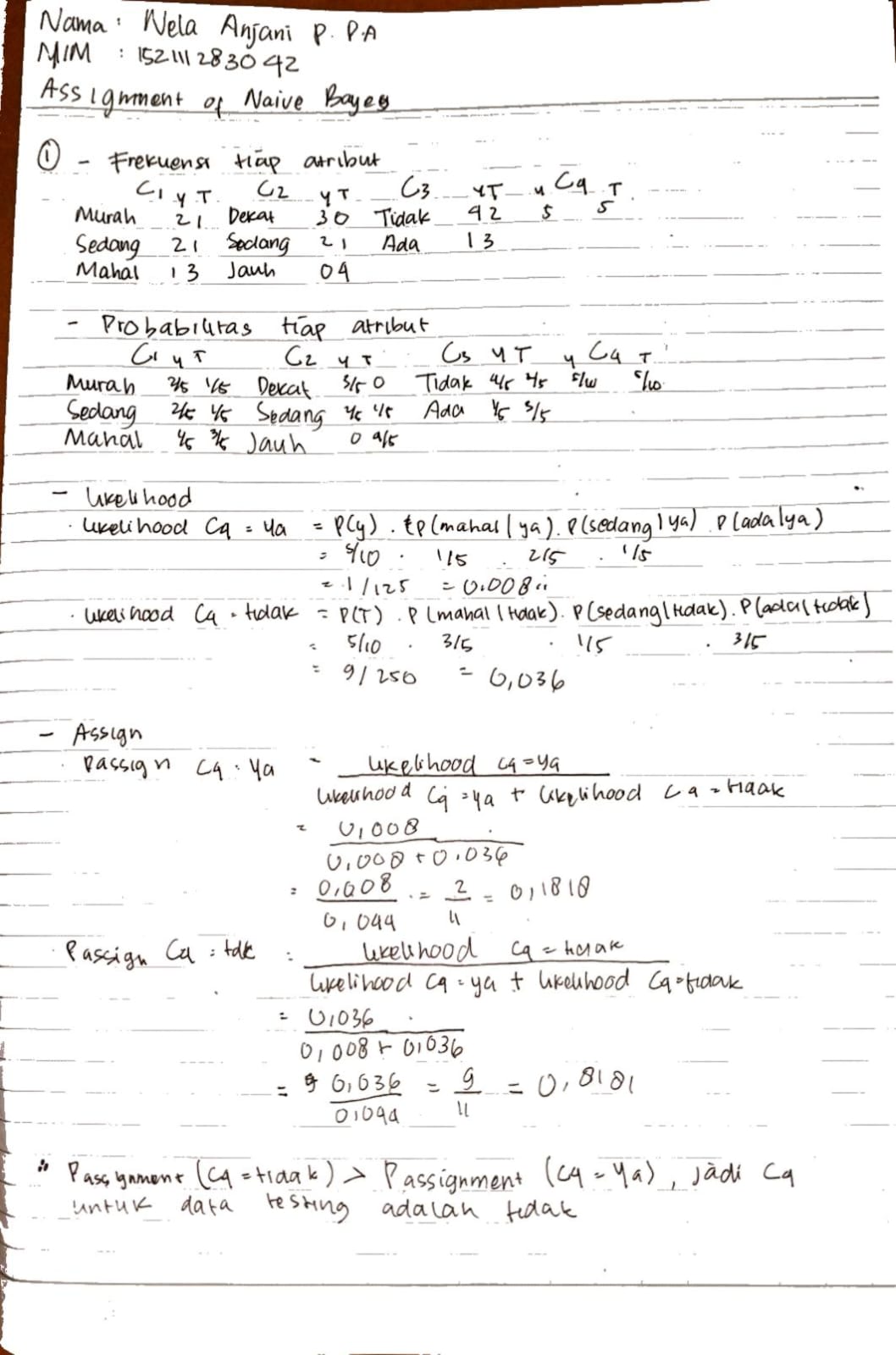
TI –C

Fakulttas Vokasi

Universitas Airlangga

[152111283042] | [Nela Anjani] | [6 November 2023]

# NO 1



# NO 2

# NO 1

import numpy as np

import pandas as pd

import seaborn as sns

import scipy.stats as stats

from statistics import mean

from sklearn.metrics import classification\_report, confusion\_matrix, accuracy\_score, precision\_score, recall\_score, f1\_score

from sklearn.naive\_bayes import GaussianNB

from sklearn.model\_selection import train\_test\_split

import matplotlib.pyplot as plt

from sklearn.metrics import ConfusionMatrixDisplay

from sklearn.impute import SimpleImputer

pd.options.mode.chained\_assignment = None

# Membaca data

dataframe = pd.read\_excel("PerumahanML.xlsx")

# Pilih kolom data

data = dataframe[

    ["Dipilih\_untuk\_Perumahan\_C4", "Harga\_Tanah\_C1", "Jarak\_dari\_Pusat\_Kota\_C2", "Ada\_Angkutan\_Umum\_C3"]

]

print("Data Awal".center(75, "="))

print(data)

print("============================================================")

# Pengecekan missing value

print("Pengecekan Missing Value".center(75, "="))

print(data.isnull().sum())

print("============================================================")

#Penanganan missing value

#menghapus data

print("Penanganan Missing Value")

Missing\_C1 = data['Harga\_Tanah\_C1'].isna().dropna()

Missing\_C2 = data['Jarak\_dari\_Pusat\_Kota\_C2'].isna().dropna()

Missing\_C3 = data['Ada\_Angkutan\_Umum\_C3'].isna().dropna()

print("Missing data pada Gender = ", Missing\_C1.isna().sum())

print("Missing data pada Age at diagnosis = ", Missing\_C2.isna().sum())

print("Missing data pada Race = ", Missing\_C3.isna().sum())

data\_bersih = data.dropna()

print("Data Bersih")

#Mendeteksi Outlier

print("Deteksi Outlier")

outliers = []

def detect\_outlier(data):

    threshold=2

    mean = np.mean(data)

    std = np.std(data)

    for x in data:

        z\_score = (x-mean)/std

        if np.abs(z\_score)>threshold:

            outliers.append(x)

    return outliers

#Mencetak outlier

outlier1 = detect\_outlier(data\_bersih['Harga\_Tanah\_C1'])

print("outlier kolom C1 = ",outlier1)

print("banyak outlier C1 = ",len(outlier1))

print()

outlier2 = detect\_outlier(data\_bersih['Jarak\_dari\_Pusat\_Kota\_C2'])

print("outlier kolom C2 = ",outlier2)

print("banyak outlier C2 = ",len(outlier2))

print()

outlier3 = detect\_outlier(data\_bersih['Ada\_Angkutan\_Umum\_C3'])

print("outlier kolom C3 = ",outlier3)

print("banyak outlier C3 = ",len(outlier3))

print()

#penanganan outlier

variabel = ["Harga\_Tanah\_C1", "Jarak\_dari\_Pusat\_Kota\_C2", "Ada\_Angkutan\_Umum\_C3"]

for var in variabel:

    outlier\_datapoints = detect\_outlier(data\_bersih[var])

    print("outlier ", var, " = ", outlier\_datapoints)

    rata = mean(data\_bersih[var])

    print("outlier ", var, " telah diganti menjadi mean ")

    data[var] = data\_bersih[var].replace(outlier\_datapoints, rata)

    print(data)

#Normalisasi

#Z-Score

zscores = stats.zscore(data\_bersih, axis=1)

print("Hasil Z Score = ")

print(zscores)

# Grouping variabel

print("GROUPING VARIABEL".center(75, "="))

X = data.iloc[:, 1:

].values  # Mengambil kolom kedua hingga terakhir sebagai variabel independen

y = data.iloc[:, 0].values  # Mengambil kolom pertama (Grade) sebagai variabel dependen

print("Data Variabel".center(75, "="))

print(X)

print("Data Kelas".center(75, "="))

print(y)

print("============================================================")

# Pembagian training dan testing

print("SPLITTING DATA 80-20".center(75, "="))

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=0)

imputer = SimpleImputer(strategy='most\_frequent')

# Melakukan imputasi pada data train

X\_train = imputer.fit\_transform(X\_train)

X\_test = imputer.fit\_transform(X\_test)

print("Instance Variabel Data Training".center(75, "="))

print(X\_train)

print("Instance Kelas Data Training".center(75, "="))

print(y\_train)

print("Instance Variabel Data Testing".center(75, "="))

print(X\_test)

print("Instance Kelas Data Testing".center(75, "="))

print(y\_test)

print("============================================================")

print()

# pemodelan naive bayes

print("PEMODELAN DENGAN NAIVE BAYES".center(75, "="))

gaussian = GaussianNB()

gaussian.fit(X\_train, y\_train)

Y\_pred = gaussian.predict(X\_test)

accuracy\_nb = round(accuracy\_score(y\_test, Y\_pred) \* 100, 2)

acc\_gaussian = round(gaussian.score(X\_train, y\_train) \* 100, 2)

print("instance prediksi naive bayes:")

print(Y\_pred)

# perhitungan confusion matrix

cm = confusion\_matrix(y\_test, Y\_pred)

print("CLASSIFICATION REPORT NAIVE BAYES".center(75, "="))

# Mendapat Akurasi

accuracy = accuracy\_score(y\_test, Y\_pred)

# Mendapat presisi

precision = precision\_score(y\_test, Y\_pred)

# Menghitung recall

recall = recall\_score(y\_test, Y\_pred)

# Menghitung F1 score

f1 = f1\_score(y\_test, Y\_pred)

# Menampilkan precision recall  f1-score   support

print(classification\_report(y\_test, Y\_pred))

cm = confusion\_matrix(y\_test, Y\_pred)

TN = cm[1][1] \* 1.0

FN = cm[1][0] \* 1.0

TP = cm[0][0] \* 1.0

FP = cm[0][1] \* 1.0

total = TN + FN + TP + FP

sens = TN / (TN + FP) \* 100

spec = TP / (TP + FN) \* 100

print("Akurasi : ", accuracy \* 100, "%")

print("Sensitivity : " + str(sens))

print("Specificity : " + str(spec))

print("Precision : " + str(precision))

print("Recall : " + str(recall))

print("F1 Score : " + str(f1))

print("============================================================")

print()

# Menampilkan Confusion Matrix

cm\_display = ConfusionMatrixDisplay(confusion\_matrix=cm)

print("Confusion matrix for Naive Bayes\n", cm)

f, ax = plt.subplots(figsize=(8, 5))

sns.heatmap(confusion\_matrix(y\_test, Y\_pred), annot=True, fmt=".0f", ax=ax)

plt.xlabel("Predicted")

plt.ylabel("Actual")

# plt.show()

print("============================================================")

print()

# COBA INPUT

print("Masukkan data untuk prediksi Kelayakan Area:")

Harga\_Tanah\_C1 = int(input("Harga Tanah (1 for Murah, 2 for Sedang, 3 for Mahal): "))

Jarak\_dari\_Pusat\_Kota\_C2= int(input("Jarak dari Pusat Kota (1 for Dekat, 2 for Sedang, 3 for Jauh): "))

Ada\_Angkutan\_Umum\_C3 = int(input("Angkutan Umum (0 for Tidak, 1 for Ada): "))

Train = [Harga\_Tanah\_C1, Jarak\_dari\_Pusat\_Kota\_C2, Ada\_Angkutan\_Umum\_C3]

print(Train)

input = pd.DataFrame(Train).T

# Melakukan prediksi

predtest = gaussian.predict(input)

if predtest == 1:

    print("Ya")

else:

    print("Tidak")

# HASIL

=================================Data Awal=================================

Dipilih\_untuk\_Perumahan\_C4 Harga\_Tanah\_C1 Jarak\_dari\_Pusat\_Kota\_C2 Ada\_Angkutan\_Umum\_C3

0 1 1 1 0

1 1 2 1 0

2 1 3 1 0

3 0 3 3 0

4 0 3 2 0

5 0 2 3 1

6 0 1 3 1

7 1 1 2 0

8 0 3 3 1

9 1 2 2 1

============================================================

==========================Pengecekan Missing Value=========================

Dipilih\_untuk\_Perumahan\_C4 0

Harga\_Tanah\_C1 0

Jarak\_dari\_Pusat\_Kota\_C2 0

Ada\_Angkutan\_Umum\_C3 0

dtype: int64

============================================================

Penanganan Missing Value

Missing data pada Gender = 0

Missing data pada Age at diagnosis = 0

Missing data pada Race = 0

Data Bersih

Deteksi Outlier

outlier kolom C1 = []

banyak outlier C1 = 0

outlier kolom C2 = []

banyak outlier C2 = 0

outlier kolom C3 = []

banyak outlier C3 = 0

outlier Harga\_Tanah\_C1 = []

outlier Harga\_Tanah\_C1 telah diganti menjadi mean

Dipilih\_untuk\_Perumahan\_C4 Harga\_Tanah\_C1 Jarak\_dari\_Pusat\_Kota\_C2 Ada\_Angkutan\_Umum\_C3

0 1 1 1 0

1 1 2 1 0

2 1 3 1 0

3 0 3 3 0

4 0 3 2 0

5 0 2 3 1

6 0 1 3 1

7 1 1 2 0

8 0 3 3 1

9 1 2 2 1

outlier Jarak\_dari\_Pusat\_Kota\_C2 = []

outlier Jarak\_dari\_Pusat\_Kota\_C2 telah diganti menjadi mean

Dipilih\_untuk\_Perumahan\_C4 Harga\_Tanah\_C1 Jarak\_dari\_Pusat\_Kota\_C2 Ada\_Angkutan\_Umum\_C3

0 1 1 1 0

1 1 2 1 0

2 1 3 1 0

3 0 3 3 0

4 0 3 2 0

5 0 2 3 1

6 0 1 3 1

7 1 1 2 0

8 0 3 3 1

9 1 2 2 1

outlier Ada\_Angkutan\_Umum\_C3 = []

outlier Ada\_Angkutan\_Umum\_C3 telah diganti menjadi mean

Dipilih\_untuk\_Perumahan\_C4 Harga\_Tanah\_C1 Jarak\_dari\_Pusat\_Kota\_C2 Ada\_Angkutan\_Umum\_C3

0 1 1 1 0

1 1 2 1 0

2 1 3 1 0

3 0 3 3 0

4 0 3 2 0

5 0 2 3 1

6 0 1 3 1

7 1 1 2 0

8 0 3 3 1

9 1 2 2 1

Hasil Z Score =

Dipilih\_untuk\_Perumahan\_C4 Harga\_Tanah\_C1 Jarak\_dari\_Pusat\_Kota\_C2 Ada\_Angkutan\_Umum\_C3

0 0.577350 0.577350 0.577350 -1.732051

1 0.000000 1.414214 0.000000 -1.414214

2 -0.229416 1.605910 -0.229416 -1.147079

3 -1.000000 1.000000 1.000000 -1.000000

4 -0.962250 1.347151 0.577350 -0.962250

5 -1.341641 0.447214 1.341641 -0.447214

6 -1.147079 -0.229416 1.605910 -0.229416

7 0.000000 0.000000 1.414214 -1.414214

8 -1.347151 0.962250 0.962250 -0.577350

9 -1.000000 1.000000 1.000000 -1.000000

=============================GROUPING VARIABEL=============================

===============================Data Variabel===============================

[[1 1 0]

[2 1 0]

[3 1 0]

[3 3 0]

[3 2 0]

[2 3 1]

[1 3 1]

[1 2 0]

[3 3 1]

[2 2 1]]

=================================Data Kelas================================

[1 1 1 0 0 0 0 1 0 1]

============================================================

============================SPLITTING DATA 80-20===========================

======================Instance Variabel Data Training======================

[[3 2 0]

[2 2 1]

[2 1 0]

[1 3 1]

[1 2 0]

[3 3 0]

[1 1 0]

[2 3 1]]

========================Instance Kelas Data Training=======================

[0 1 1 0 1 0 1 0]

=======================Instance Variabel Data Testing======================

[[3 1 0]

[3 3 1]]

========================Instance Kelas Data Testing========================

[1 0]

============================================================

========================PEMODELAN DENGAN NAIVE BAYES=======================

instance prediksi naive bayes:

[1 0]

=====================CLASSIFICATION REPORT NAIVE BAYES=====================

precision recall f1-score support

0 1.00 1.00 1.00 1

1 1.00 1.00 1.00 1

accuracy 1.00 2

macro avg 1.00 1.00 1.00 2

weighted avg 1.00 1.00 1.00 2

Akurasi : 100.0 %

Sensitivity : 100.0

Specificity : 100.0

Precision : 1.0

Recall : 1.0

F1 Score : 1.0

============================================================

Confusion matrix for Naive Bayes

[[1 0]

[0 1]]

============================================================

Masukkan data untuk prediksi Kelayakan Area:

Harga Tanah (1 for Murah, 2 for Sedang, 3 for Mahal): 3

Jarak dari Pusat Kota (1 for Dekat, 2 for Sedang, 3 for Jauh): 2

Angkutan Umum (0 for Tidak, 1 for Ada): 1

[3, 2, 1]

Tidak

# NO 2

import numpy as np

import pandas as pd

from sklearn.metrics import accuracy\_score ,precision\_score, confusion\_matrix, classification\_report

from sklearn.naive\_bayes import GaussianNB

from sklearn.preprocessing import LabelEncoder

from sklearn.model\_selection import train\_test\_split

pd.options.mode.chained\_assignment = None

#membaca data

dataframe = pd.read\_excel("RekeningML.xlsx")

data=dataframe[['Usia', 'Pekerjaan', 'Pendapatan\_dalam\_Juta', 'Membuka\_Rekening']]

print("data awal".center(75,"="))

print(data)

print("============================================================")

ubahvalue=LabelEncoder()

data['Pekerjaan']=ubahvalue.fit\_transform(data['Pekerjaan'])

data['Membuka\_Rekening']=ubahvalue.fit\_transform(data['Membuka\_Rekening'])

data\_usia = np.log2(data['Usia'])

data\_pendapatan = np.log2(data['Pendapatan\_dalam\_Juta'])

data['Usia'] = np.floor(data\_usia)

# <32 = 4, 32 - 63 = 5, >63 = 6

data['Pendapatan\_dalam\_Juta'] = np.floor(data\_pendapatan)

# <4 = 1, 4 - 7 = 2, >7 = 3

print(data)

#grouping yang dibagi menjadi dua

print("GROUPING VARIABEL".center(75,"="))

X=data.iloc[:,0:-1].values

y=data.iloc[:,3].values

print("data variabel".center(75,"="))

print(X)

print("data kelas".center(75,"="))

print(y)

print("============================================================")

#pembagian training dan testing

print("SPLITTING DATA 20-80".center(75,"="))

X\_train,X\_test,y\_train,y\_test=train\_test\_split(X,y,test\_size=0.2,random\_state=0)

print("instance variabel data training".center(75,"="))

print(X\_train)

print("instance kelas data training".center(75,"="))

print(y\_train)

print("instance variabel data testing".center(75,"="))

print(X\_test)

print("instance kelas data testing".center(75,"="))

print(y\_test)

print("============================================================")

print()

#pemodelan Naive Bayes

print("PEMODELAN DENGAN Naive Bayes".center(75,"="))

gaussian = GaussianNB()

gaussian.fit(X\_train, y\_train)

Y\_pred = gaussian.predict(X\_test)

print("instance prediksi Naive Bayes:")

print(Y\_pred)

print('CLASSIFICATION REPORT Naive Bayes'.center(75,'='))

#Mendapat Akurasi

accuracy = accuracy\_score(y\_test, Y\_pred)

cm = confusion\_matrix(y\_test, Y\_pred)

print('Confusion matrix for Naive Bayes\n',cm)

print("============================================================")

print()

print('Akurasi : ', accuracy)

print("============================================================")

print()

#COBA INPUT

print("Inputkan Data")

print("4 for Usia < 32 , 5 for Usia 32 - 63, 6 for Usia > 63")

A = int(input("Usia = "))

print("0 for Karyawan, 1 for Mahasiswa, 2 for Pelajar, 3 for Pensiunan, 4 for Wiraswasta")

B = int(input("Pekerjaan = "))

print("1 for Pendapatan < 4, 2 for pendapatan 4 - 7, 3 for pendapatan > 7")

C = int(input("Pendapatan (dalam juta) = "))

Train = [A,B,C]

print(Train)

test = pd.DataFrame(Train).T

predtest = gaussian.predict(test)

print(predtest)

if(predtest == 1):

    print('Ya')

else:

    print('Tidak')

# HASIL

=================================data awal=================================

Usia Pekerjaan Pendapatan\_dalam\_Juta Membuka\_Rekening

0 22 Pelajar 2 Ya

1 25 Mahasiswa 3 Ya

2 30 Karyawan 4 Tidak

3 35 Karyawan 7 Ya

4 40 Wiraswasta 10 Tidak

5 45 Wiraswasta 12 Tidak

6 50 Wiraswasta 15 Tidak

7 55 Karyawan 8 Ya

8 60 Pensiunan 6 Tidak

9 65 Pensiunan 5 Tidak

============================================================

Usia Pekerjaan Pendapatan\_dalam\_Juta Membuka\_Rekening

0 4.0 2 1.0 1

1 4.0 1 1.0 1

2 4.0 0 2.0 0

3 5.0 0 2.0 1

4 5.0 4 3.0 0

5 5.0 4 3.0 0

6 5.0 4 3.0 0

7 5.0 0 3.0 1

8 5.0 3 2.0 0

9 6.0 3 2.0 0

=============================GROUPING VARIABEL=============================

===============================data variabel===============================

[[4. 2. 1.]

[4. 1. 1.]

[4. 0. 2.]

[5. 0. 2.]

[5. 4. 3.]

[5. 4. 3.]

[5. 4. 3.]

[5. 0. 3.]

[5. 3. 2.]

[6. 3. 2.]]

=================================data kelas================================

[1 1 0 1 0 0 0 1 0 0]

============================================================

============================SPLITTING DATA 20-80===========================

======================instance variabel data training======================

[[5. 4. 3.]

[6. 3. 2.]

[4. 1. 1.]

[5. 4. 3.]

[5. 0. 3.]

[5. 0. 2.]

[4. 2. 1.]

[5. 4. 3.]]

========================instance kelas data training=======================

[0 0 1 0 1 1 1 0]

=======================instance variabel data testing======================

[[4. 0. 2.]

[5. 3. 2.]]

========================instance kelas data testing========================

[0 0]

============================================================

========================PEMODELAN DENGAN Naive Bayes=======================

instance prediksi Naive Bayes:

[1 0]

=====================CLASSIFICATION REPORT Naive Bayes=====================

Confusion matrix for Naive Bayes

[[1 1]

[0 0]]

============================================================

Akurasi : 0.5

============================================================

Inputkan Data

4 for Usia < 32 , 5 for Usia 32 - 63, 6 for Usia > 63

Usia = 4

0 for Karyawan, 1 for Mahasiswa, 2 for Pelajar, 3 for Pensiunan, 4 for Wiraswasta

Pekerjaan = 1

1 for Pendapatan < 4, 2 for pendapatan 4 - 7, 3 for pendapatan > 7

Pendapatan (dalam juta) = 2

[4, 1, 2]

[1]

Ya