UAS Pemodelan Data dengan Python Kelompok 10

Kelompok 10 :

1. **Claresta Tirta S. G (6161901025)**
2. **Christian Jauhari (6161901028)**
3. **Fransiska Nadya C. P (6161901051)**
4. **Tiara Alamanda (6161901116)**

Preprocessing Data

Load Data

In [ ]:

```
import pandas as pd
url = 'https://raw.githubusercontent.com/blackhespy/heart-disease-data/main/Heart%20Disea
se.csv'
df = pd.read_csv(url)
df
```

Out[ ]:

	male	age	education	currentSmoker	cigsPerDay	BPMeds	prevalentStroke	prevalentHyp	diabetes	totChol	sysBP	d
0	1	39	4.0	0	0.0	0.0	0	0	0	195.0	106.0	
1	0	46	2.0	0	0.0	0.0	0	0	0	250.0	121.0	
2	1	48	1.0	1	20.0	0.0	0	0	0	245.0	127.5	
3	0	61	3.0	1	30.0	0.0	0	1	0	225.0	150.0	
4	0	46	3.0	1	23.0	0.0	0	0	0	285.0	130.0	
...	...	...	...	...	...	...	...	...	...	...	...	
4235	0	48	2.0	1	20.0	NaN	0	0	0	248.0	131.0	
4236	0	44	1.0	1	15.0	0.0	0	0	0	210.0	126.5	
4237	0	52	2.0	0	0.0	0.0	0	0	0	269.0	133.5	
4238	1	40	3.0	0	0.0	0.0	0	1	0	185.0	141.0	
4239	0	39	3.0	1	30.0	0.0	0	0	0	196.0	133.0	

4240 rows × 16 columns

Sebelumnya sudah di load data csv mengenai sejumlah data terkait penyebab penyakit jantung dan sudah dimasukkan ke dalam variable df. Data ini berjumlah 4240 baris dan 16 kolom.

Langkah-langkah Persiapan Data

In [ ]:

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 4240 entries, 0 to 4239
Data columns (total 16 columns):
 #   Column           Non-Null Count  Dtype
---  ------           --------------  -----
 0   male             4240 non-null   int64
 1   age              4240 non-null   int64
 2   education        4135 non-null   float64
 3   currentSmoker    4240 non-null   int64
 4   cigsPerDay       4211 non-null   float64
 5   BPMeds           4187 non-null   float64
 6   prevalentStroke  4240 non-null   int64
 7   prevalentHyp     4240 non-null   int64
 8   diabetes         4240 non-null   int64
 9   totChol          4190 non-null   float64
 10  sysBP            4240 non-null   float64
 11  diaBP            4240 non-null   float64
 12  BMI              4221 non-null   float64
 13  heartRate        4239 non-null   float64
 14  glucose          3852 non-null   float64
 15  TenYearCHD       4240 non-null   int64
dtypes: float64(9), int64(7)
memory usage: 530.1 KB
```

Lalu dengan menggunakan df.info(), dapat dilihat untuk kolom seperti Education, cigsPerDay, BPMeds, totChol, BMI, heartRate, dan glucose terdapat data yang bernilai Null (data kosong) karena jumlah baris yang ditunjukkan tidak sama dengan jumlah baris pada data awal, yaitu 4240 baris. Dapat diperhatikan pula tipe data dari masing-masing kolom.

In [ ]:

```
df.describe()
```

Out[ ]:

	male	age	education	currentSmoker	cigsPerDay	BPMeds	prevalentStroke	prevalentHyp	di
count	4240.000000	4240.000000	4135.000000	4240.000000	4211.000000	4187.000000	4240.000000	4240.000000	4240.
mean	0.429245	49.580189	1.979444	0.494104	9.005937	0.029615	0.005896	0.310613	0.
std	0.495027	8.572942	1.019791	0.500024	11.922462	0.169544	0.076569	0.462799	0.
min	0.000000	32.000000	1.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.
25%	0.000000	42.000000	1.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.
50%	0.000000	49.000000	2.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.
75%	1.000000	56.000000	3.000000	1.000000	20.000000	0.000000	0.000000	1.000000	0.
max	1.000000	70.000000	4.000000	1.000000	70.000000	1.000000	1.000000	1.000000	1.

In [ ]:

```
df = df.drop(columns=['education', 'cigsPerDay','BPMeds','diabetes','sysBP','diaBP','hea
rtRate'])
df
```

Out[ ]:

	male	age	currentSmoker	prevalentStroke	prevalentHyp	totChol	BMI	glucose	TenYearCHD
0	1	39	0	0	0	195.0	26.97	77.0	0
1	0	46	0	0	0	250.0	28.73	76.0	0
2	1	48	1	0	0	245.0	25.34	70.0	0
3	0	61	1	0	1	225.0	28.58	103.0	1
4	0	46	1	0	0	285.0	23.10	85.0	0
...	...	...	...	...	...	...	...	...	...
4235	0	48	1	0	0	248.0	22.00	86.0	0
4236	0	44	1	0	0	210.0	19.16	NaN	0
4237	0	52	0	0	0	269.0	21.47	107.0	0
4238	1	40	0	0	1	185.0	25.60	72.0	0

	male	age	currentSmoker	prevalentStroke	prevalentHyp	totChol	BMI	glucose	TenYearCHD
4239	0	39	1	0	0	196.0	20.91	80.0	0

4240 rows × 9 columns

Kelompok kami memutuskan untuk menghapus kolom education, cigsPerDay, BPMeds, diabetes, sysBP, diaBP, dan heartRate. Kolom education dihapus karena menurut kami, pendidikan tidak ada hubungannya dengan penyakit jantung. Lalu, kami menghapus kolom cigsPerDay, BPMeds, dan diabetes karena bisa diketahui dari kolom yang lain. Kemudian kolom sysBP, diaBP, dan heartRate dapat dijelaskan oleh kolom prevalentHyp.

Cek Nilai NA pada tiap baris

In [ ]:

```
persentase_data_kosong= df.isna().sum()*100/len(df)
nilaikosong_df= pd.DataFrame({'Persentase Data Kosong': persentase_data_kosong})
nilaikosong_df
```

Out[ ]:

	Persentase Data Kosong
male	0.000000
age	0.000000
currentSmoker	0.000000
prevalentStroke	0.000000
prevalentHyp	0.000000
totChol	1.179245
BMI	0.448113
glucose	9.150943
TenYearCHD	0.000000

In [ ]:

```
df=df.dropna()
df
```

Out[ ]:

	male	age	currentSmoker	prevalentStroke	prevalentHyp	totChol	BMI	glucose	TenYearCHD
0	1	39	0	0	0	195.0	26.97	77.0	0
1	0	46	0	0	0	250.0	28.73	76.0	0
2	1	48	1	0	0	245.0	25.34	70.0	0
3	0	61	1	0	1	225.0	28.58	103.0	1
4	0	46	1	0	0	285.0	23.10	85.0	0
...	...	...	...	...	...	...	...	...	...
4234	1	51	1	0	0	207.0	19.71	68.0	0
4235	0	48	1	0	0	248.0	22.00	86.0	0
4237	0	52	0	0	0	269.0	21.47	107.0	0
4238	1	40	0	0	1	185.0	25.60	72.0	0
4239	0	39	1	0	0	196.0	20.91	80.0	0

3828 rows × 9 columns

Pada bagian ini, kami memutuskan untuk menghapus baris yang terdapat nilai Null. Sehingga diperoleh data yang baru dengan jumlah baris adalah 3828 dan jumlah kolom adalah 9.

Desicion Tree

In [ ]:

```
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import classification_report, confusion_matrix
```

Menentukan data X (response) dan y (class)

In [ ]:

```
X = df.drop(columns='TenYearCHD')
y = df["TenYearCHD"]
```

In [ ]:

```
X.head
```

Out[ ]:

```
<bound method NDFrame.head of       male  age  currentSmoker  prevalentStroke  prevalentH
yp  totChol    BMI  \
0        1   39             0                0             0    195.0  26.97
1        0   46             0                0             0    250.0  28.73
2        1   48             1                0             0    245.0  25.34
3        0   61             1                0             1    225.0  28.58
4        0   46             1                0             0    285.0  23.10
...    ...  ...            ...              ...           ...      ...    ...
4234     1   51             1                0             0    207.0  19.71
4235     0   48             1                0             0    248.0  22.00
4237     0   52             0                0             0    269.0  21.47
4238     1   40             0                0             1    185.0  25.60
4239     0   39             1                0             0    196.0  20.91

      glucose
0        77.0
1        76.0
2        70.0
3       103.0
4        85.0
...       ...
4234     68.0
4235     86.0
4237    107.0
4238     72.0
4239     80.0

[3828 rows x 8 columns]>
```

Splitting data dengan Scikit-Learn (Training 80%, testing 20%)

In [ ]:

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.2)
```

Model Training

In [ ]:

```
classifier = DecisionTreeClassifier(max_depth=5)
classifier.fit(X_train,y_train)
```

Out[ ]:

```
DecisionTreeClassifier(max_depth=5)
```

In [ ]:

In [ ]:

```
import graphviz
from sklearn import tree
# DOT data
dot_data = tree.export_graphviz(classifier, out_file=None,
filled=True)
# Draw graph
graph = graphviz.Source(dot_data, format="png")
graph
```

Out[ ]:

prediction

In [ ]:

```
y_pred = classifier.predict(X_test)
y_pred
```

Out[ ]:

```
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, 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, 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, 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, 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, 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, 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, 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, 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, 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, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 1, 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, 1, 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, 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, 0, 0, 0, 0, 0, 0, 0, 0, 1, 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, 1, 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, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 1, 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, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 1, 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, 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, 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, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0])
```

evaluation

In [ ]:

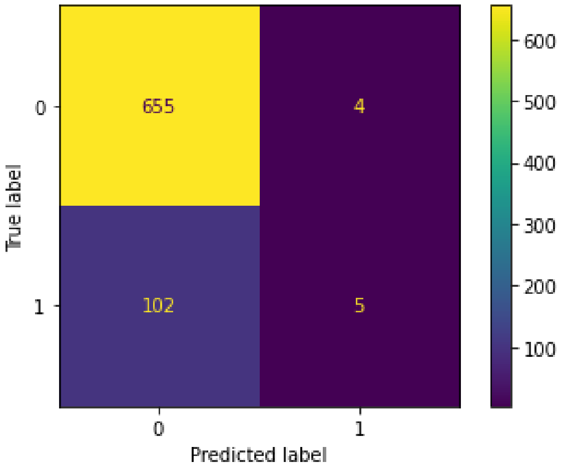
```
from sklearn.metrics import plot_confusion_matrix
plot_confusion_matrix(classifier, X_test, y_test)
```

```
/usr/local/lib/python3.7/dist-packages/sklearn/utils/deprecation.py:87: FutureWarning: Fu
nction plot_confusion_matrix is deprecated; Function `plot_confusion_matrix` is deprecate
d in 1.0 and will be removed in 1.2. Use one of the class methods: ConfusionMatrixDisplay
.from_predictions or ConfusionMatrixDisplay.from_estimator.
```

```
  warnings.warn(msg, category=FutureWarning)
```

Out[ ]:

```
<sklearn.metrics._plot.confusion_matrix.ConfusionMatrixDisplay at 0x7f1e4b1b11d0>
```



In [ ]:

```
print(classification_report(y_test,y_pred))
```

```
              precision    recall  f1-score   support

           0       0.87      0.99      0.93       659
           1       0.56      0.05      0.09       107

    accuracy                           0.86       766
   macro avg       0.71      0.52      0.51       766
weighted avg       0.82      0.86      0.81       766
```