LAPORAN PRAKTIKUM MACHINE LEARNING

VALIDATION MODEL OF CLASSIFICATION

Dosen Pengampu : Entin Martiana Kusumaningtyas S.Kom, M.Kom.

Bayu Kurniawan / 3322600019

```python
# assignment 1
from sklearn.model_selection import train_test_split
import pandas as pd
import numpy as np

dataset = pd.read_csv("C:/Users/bayuk/OneDrive/Documents/AI/pens/smtr3/Machine Learning/Data/milk.csv")
dataset
```

✓ 0.0s

	pH	Temprature	Taste	Odor	Fat	Turbidity	Colour	Grade
0	6.6	35	1	0	1	0	254	high
1	6.6	36	0	1	0	1	253	high
2	8.5	70	1	1	1	1	246	low
3	9.5	34	1	1	0	1	255	low
4	6.6	37	0	0	0	0	255	medium
...	...	...	...	...	...	...	...	...
1054	6.7	45	1	1	0	0	247	medium
1055	6.7	38	1	0	1	0	255	high
1056	3.0	40	1	1	1	1	255	low
1057	6.8	43	1	0	1	0	250	high
1058	8.6	55	0	1	1	1	255	low

1059 rows × 8 columns

```python
# assignment 2
# a. Hold-out Method (70%-30%)
datalabel = dataset.loc[:,['Grade']]
xtrain, xtest, ytrain, ytest = train_test_split(dataset, datalabel, test_size = 0.30, random_state = 100)

# a 3
train_data = np.array(xtrain)[:,:-1]
newmin = 0
newmax = 1
mindata = train_data.min()
maxdata = train_data.max()
train_data = ((train_data-mindata)*(newmax-newmin)/(maxdata-mindata))+newmin
print("Train data : ", train_data)

# a 4
test_data = np.array(xtest)[:,:-1]
newmin = 0
newmax = 1
mindata = test_data.min()
maxdata = test_data.max()
test_data = ((test_data-mindata)*(newmax-newmin)/(maxdata-mindata))+newmin
print("Test data : ", test_data)

# a 5
from sklearn.neighbors import KNeighborsClassifier
kNN = KNeighborsClassifier(n_neighbors=3, weights='distance')
kNN.fit(train_data, ytrain)
hasil = kNN.predict(test_data)
print("Hasil k-NN : ", hasil)

precision_ratio = kNN.score(test_data, ytest)
error_ratio = 1 - precision_ratio
print("Error ratio : ", error_ratio)
```

✓ 0.0s

```
Train data :  [[0.026666666666666665 0.1568627450980392 0.00392156862745098 ...
  0.00392156862745098 0.0 0.9607843137254902]
 [0.01843137254901961 0.14901960784313725 0.00392156862745098 ...
  0.00392156862745098 0.0 1.0]
 [0.02588235294117647 0.16862745098039217 0.0 ... 0.00392156862745098 0.0
  0.9803921568627451]
 ...
 [0.025490196078431372 0.14901960784313725 0.00392156862745098 ... 0.0
  0.0 1.0]
 [0.026666666666666665 0.13333333333333333 0.0 ... 0.0
  0.00392156862745098 0.9411764705882353]
 [0.011764705882352941 0.1568627450980392 0.00392156862745098 ...
  0.00392156862745098 0.00392156862745098 1.0]]
Test data :  [[0.02588235294117647 0.17647058823529413 0.0 ... 0.0 0.00392156862745098
  0.9803921568627451]
 [0.021960784313725487 0.19607843137254902 0.0 ... 0.00392156862745098
  0.00392156862745098 1.0]
 [0.026666666666666665 0.17647058823529413 0.0 ... 0.00392156862745098
  0.00392156862745098 1.0]
 ...
 [0.026666666666666665 0.17647058823529413 0.0 ... 0.0
  0.00392156862745098 1.0]
 [0.03529411764705882 0.16862745098039217 0.00392156862745098 ...
  0.00392156862745098 0.00392156862745098 0.9803921568627451]
 [0.03176470588235294 0.25882352941176473 0.00392156862745098 ...
...
 'low' 'high' 'high' 'low' 'low' 'low' 'low' 'medium' 'medium' 'medium'
 'medium' 'low' 'low' 'medium' 'medium' 'low' 'low' 'high' 'medium'
 'medium' 'high' 'low' 'high' 'medium' 'low' 'low']
Error ratio :  0.0062893081761006275
```

Output is truncated. View as a scrollable element or open in a text editor. Adjust cell output settings...

```
C:\Users\bayuk\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.10_qbz5n2kfra8p0\LocalCache\local-packages\Pytho
  return self._fit(X, y)
```

```python
# b. K-Fold (k=10)
from sklearn.model_selection import KFold

kf = KFold(n_splits = 10, random_state = 0, shuffle = True)
p = 0
for train_index, test_index in kf.split(dataset):
    p = p+1
    xtrain = dataset.loc[train_index]
    xtest = dataset.loc[test_index]
    ytrain = xtrain.loc[:,["Grade"]]
    ytest = xtest.loc[:,["Grade"]]

# b 3
train_data = np.array(xtrain)[:,:-1]
newmin = 0
newmax = 1
mindata = train_data.min()
maxdata = train_data.max()
train_data = ((train_data-mindata)*(newmax-newmin)/(maxdata-mindata))+newmin
print("Train data : ", train_data)

# b 4
test_data = np.array(xtest)[:,:-1]
newmin = 0
newmax = 1
mindata = test_data.min()
maxdata = test_data.max()
test_data = ((test_data-mindata)*(newmax-newmin)/(maxdata-mindata))+newmin
print("Test data : ", test_data)

# b 5
from sklearn.neighbors import KNeighborsClassifier
kNN = KNeighborsClassifier(n_neighbors=3, weights='distance')
kNN.fit(train_data, ytrain)
hasil = kNN.predict(test_data)
print("Hasil k-NN : ", hasil)

precision_ratio = kNN.score(test_data, ytest)
error_ratio = 1 - precision_ratio
print("Error ratio : ", error_ratio)
```

✓ 0.0s

```
Train data :  [[0.02588235294117647 0.13725490196078433 0.00392156862745098 ...
  0.00392156862745098 0.0 0.996078431372549]
 [0.02588235294117647 0.1411764705882353 0.0 ... 0.0 0.00392156862745098
  0.9921568627450981]
 [0.03333333333333333 0.27450980392156865 0.00392156862745098 ...
  0.00392156862745098 0.00392156862745098 0.9647058823529412]
 ...
 [0.011764705882352941 0.1568627450980392 0.00392156862745098 ...
  0.00392156862745098 0.00392156862745098 1.0]
 [0.026666666666666665 0.16862745098039217 0.00392156862745098 ...
  0.00392156862745098 0.0 0.9803921568627451]
 [0.03372549019607843 0.21568627450980393 0.0 ... 0.00392156862745098
  0.00392156862745098 1.0]]
Test data :  [[0.025490196078431372 0.1450980392156863 0.0 0.0 0.0 0.0
  0.9607843137254902]
 [0.03529411764705882 0.16862745098039217 0.00392156862745098
  0.00392156862745098 0.00392156862745098 0.00392156862745098
  0.9725490196078431]
 [0.026666666666666665 0.17647058823529413 0.00392156862745098
  0.00392156862745098 0.00392156862745098 0.0 0.9607843137254902]
 [0.03176470588235294 0.25882352941176473 0.00392156862745098 0.0
  0.00392156862745098 0.00392156862745098 1.0]
 [0.03372549019607843 0.21568627450980393 0.0 0.00392156862745098
  0.00392156862745098 0.00392156862745098 1.0]
 [0.02588235294117647 0.17647058823529413 0.0 0.00392156862745098
...
 'medium' 'medium' 'low' 'medium' 'low' 'high' 'low' 'low' 'medium'
 'medium' 'medium' 'high' 'low' 'low' 'low' 'medium' 'high' 'low' 'high'
 'medium' 'low' 'high' 'medium' 'high' 'medium' 'low' 'medium']
Error ratio :  0.0
```

Output is truncated. View as a scrollable element or open in a text editor. Adjust cell output settings...

```
C:\Users\bayuk\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.10_qbz5n2kfra8p0\LocalCache\local-pa
  return self._fit(X, y)
```

```python
# c. LOO
from sklearn.metrics import accuracy_score
from sklearn.model_selection import LeaveOneOut as LeaveOneOut

loo = LeaveOneOut()
error_ratios = []

p = 0
mindata_LOO = dataset.iloc[:, 0:-1].min()
maxdata_LOO = dataset.iloc[:, 0:-1].max()
dataset_normalized = ((dataset.iloc[:, 0:-1] - mindata_LOO) * (newmax - newmin) / (maxdata_LOO - mindata_LOO)) + newmin

for train_index, test_index in loo.split(dataset_normalized):
    xtrain = dataset_normalized.iloc[train_index]
    xtest = dataset_normalized.iloc[test_index]
    ytrain = datalabel.iloc[train_index]
    ytest = datalabel.iloc[test_index]
    p=p+1

    kNN_LOO = KNeighborsClassifier(n_neighbors=3, weights='distance')
    kNN_LOO.fit(xtrain, ytrain.values.ravel())

    y_pred_LOO = kNN_LOO.predict(xtest)

    accuracy_LOO = accuracy_score(ytest, y_pred_LOO)
    error_ratio_LOO = 1 - accuracy_LOO
error_ratios.append(error_ratio_LOO)

print(f"LOO {p}:")
print("Train data:")
print(xtrain)
print("Test data:")
print(xtest)
print("Akurasi:")
print(accuracy_LOO)
print("Error:")
print(error_ratio_LOO)
```

✓ 5.7s

```
LOO 1059:
Train data:
            pH  Temprature  Taste  Odor  Fat   Turbidity    Colour
0     0.553846    0.017857    1.0   0.0   1.0         0.0  0.933333
1     0.553846    0.035714    0.0   1.0   0.0         1.0  0.866667
2     0.846154    0.642857    1.0   1.0   1.0         1.0  0.400000
3     1.000000    0.000000    1.0   1.0   0.0         1.0  1.000000
4     0.553846    0.053571    0.0   0.0   0.0         0.0  1.000000
...        ...         ...    ...   ...   ...         ...       ...
1053  0.784615    0.571429    1.0   0.0   1.0         1.0  1.000000
1054  0.569231    0.196429    1.0   1.0   0.0         0.0  0.466667
1055  0.569231    0.071429    1.0   0.0   1.0         0.0  1.000000
1056  0.000000    0.107143    1.0   1.0   1.0         1.0  1.000000
1057  0.584615    0.160714    1.0   0.0   1.0         0.0  0.666667

[1058 rows x 7 columns]
Test data:
            pH  Temprature  Taste  Odor  Fat   Turbidity  Colour
1058  0.861538       0.375    0.0   1.0   1.0         1.0     1.0
Akurasi:
1.0
Error:
0.0
```