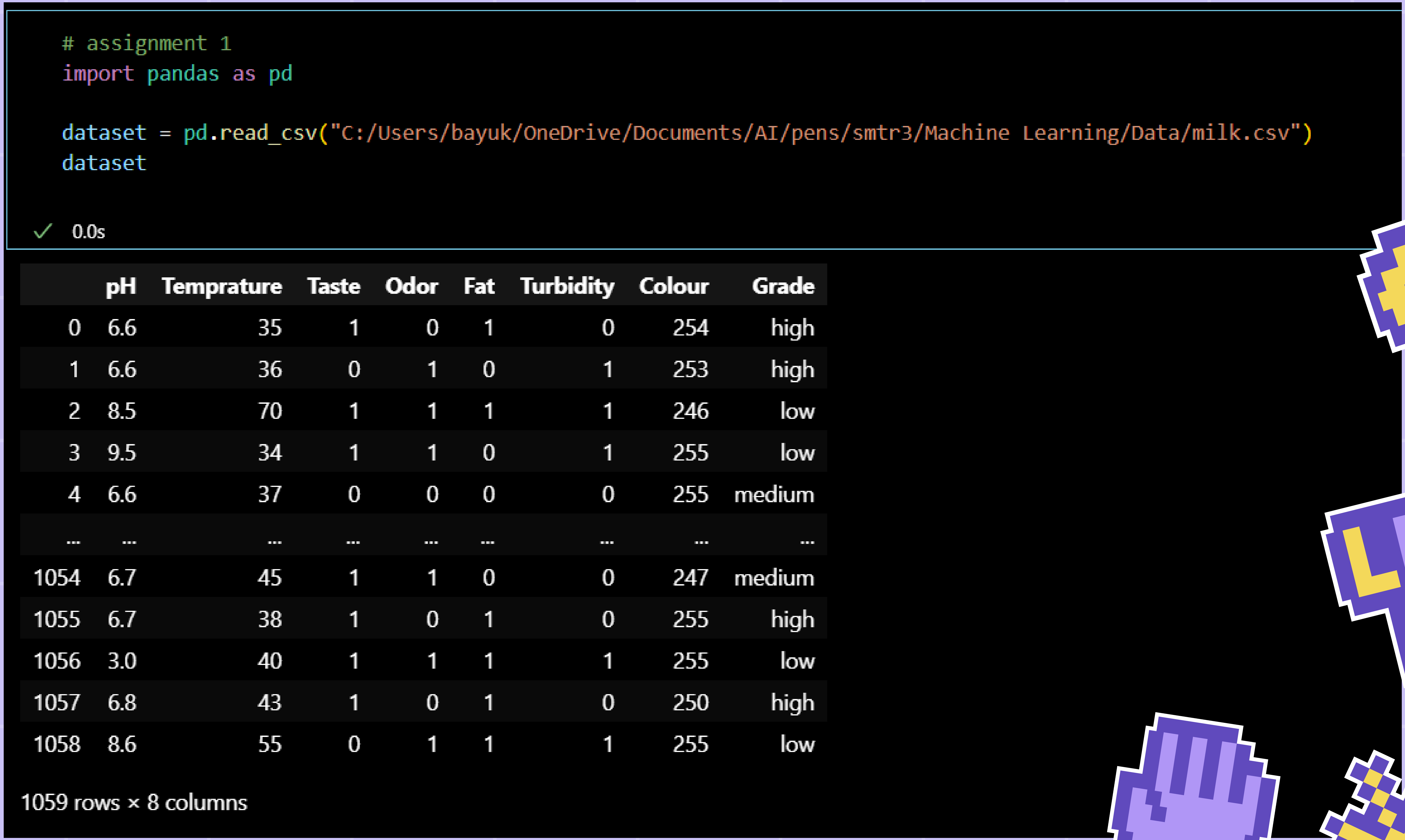
LAPORAN PRAKTIKUM MACHINE LEARNING

Praktikum Naïve Bayesian

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

Bayu Kurniawan / 3322600019

Start

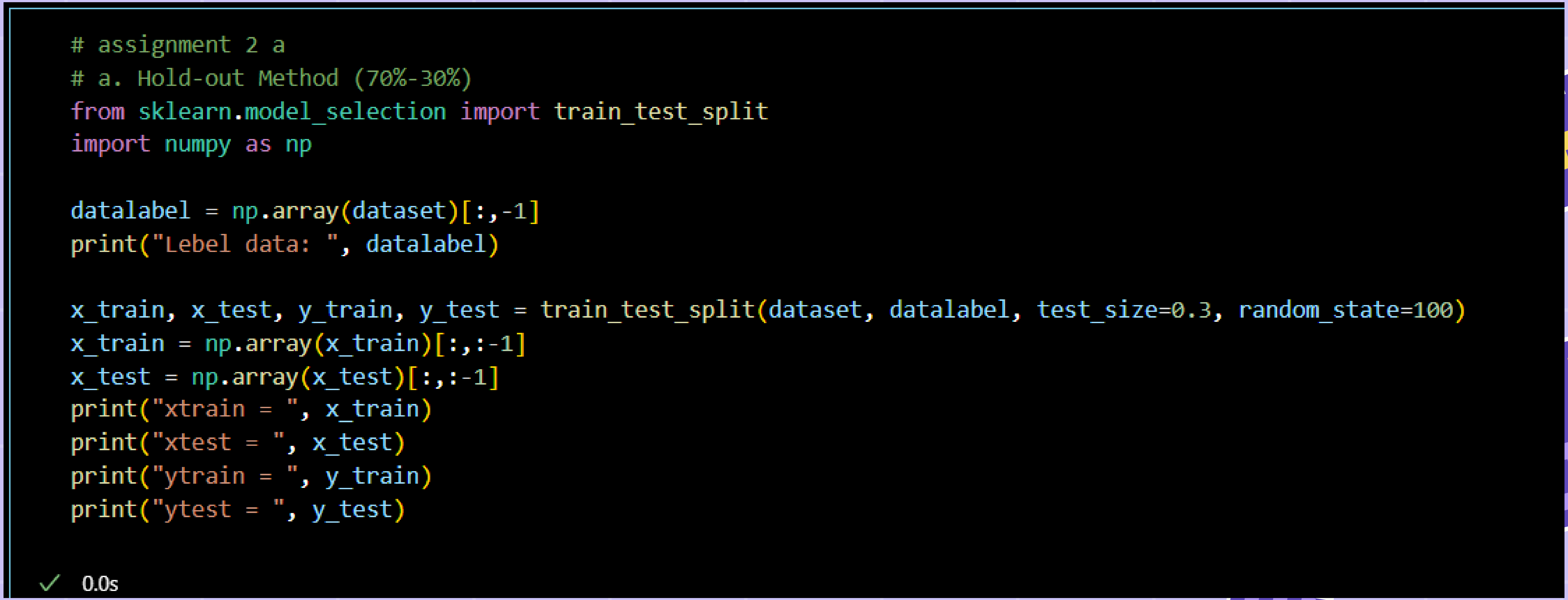


```
# assignment 1
import pandas as pd

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

✓ 0.0s

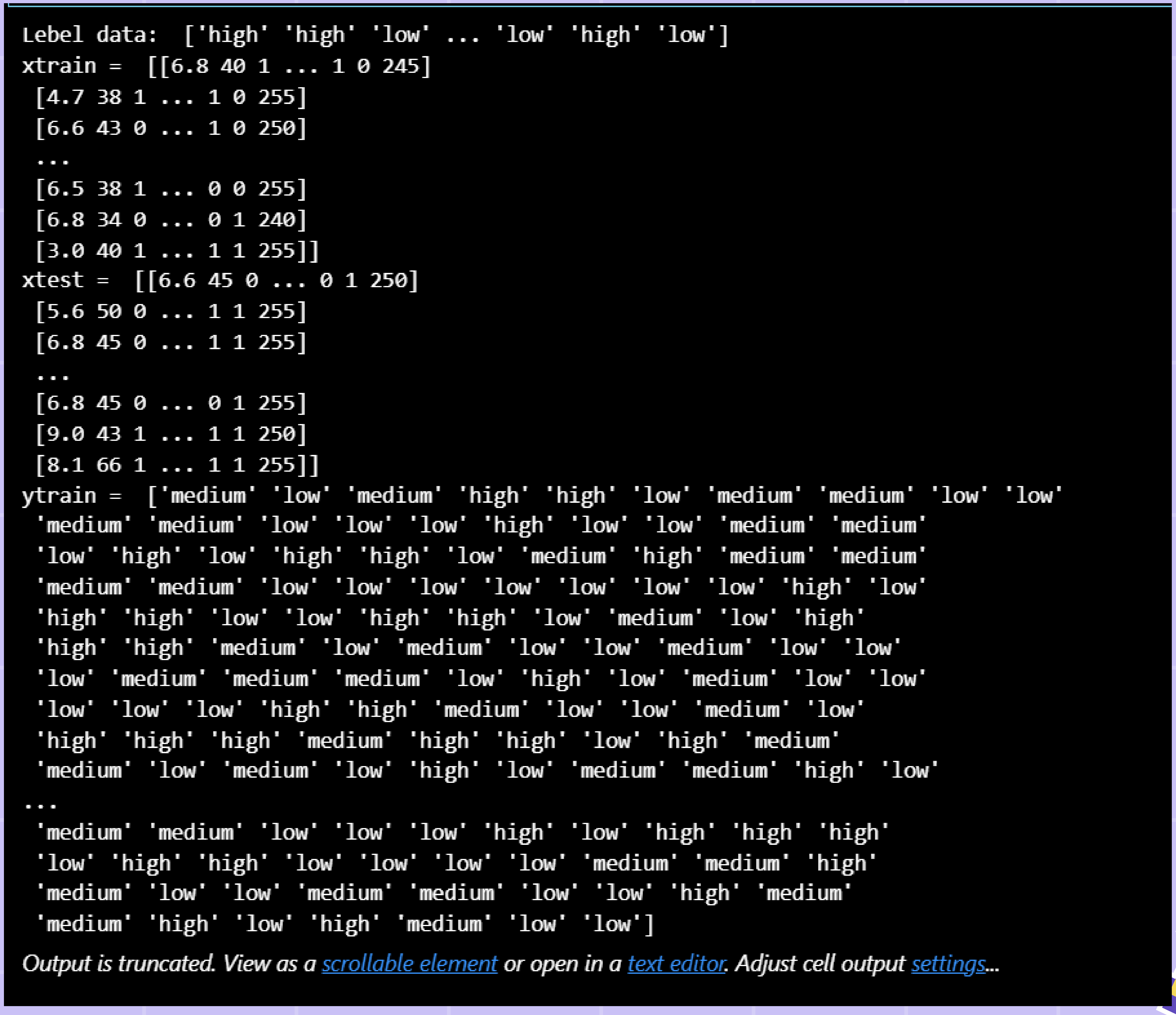
	pH	Temprature	Taste	Odor	Fat	Turbidity	Colour	Grade
0	6.6	35	1	0	1	0	254	high
1	6.6	36	0	1	0	1	253	high
2	8.5	70	1	1	1	1	246	low
3	9.5	34	1	1	0	1	255	low
4	6.6	37	0	0	0	0	255	medium
...	...	...	...	...	...	...	...	...
1054	6.7	45	1	1	0	0	247	medium
1055	6.7	38	1	0	1	0	255	high
1056	3.0	40	1	1	1	1	255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



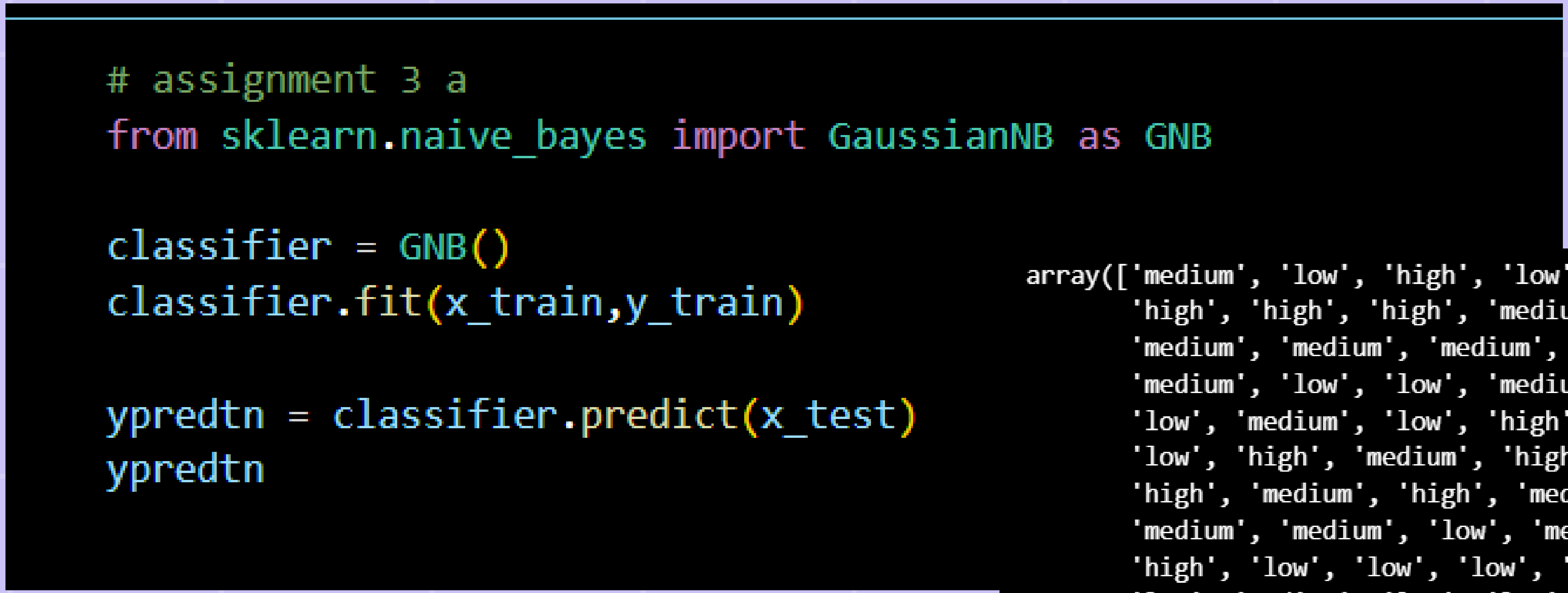
```python
# assignment 2 a
# a. Hold-out Method (70%-30%)
from sklearn.model_selection import train_test_split
import numpy as np

datalabel = np.array(dataset)[:,-1]
print("Lebel data: ", datalabel)

x_train, x_test, y_train, y_test = train_test_split(dataset, datalabel, test_size=0.3, random_state=100)
x_train = np.array(x_train)[:,:-1]
x_test = np.array(x_test)[:,:-1]
print("xtrain = ", x_train)
print("xtest = ", x_test)
print("ytrain = ", y_train)
print("ytest = ", y_test)
```

✓ 0.0s

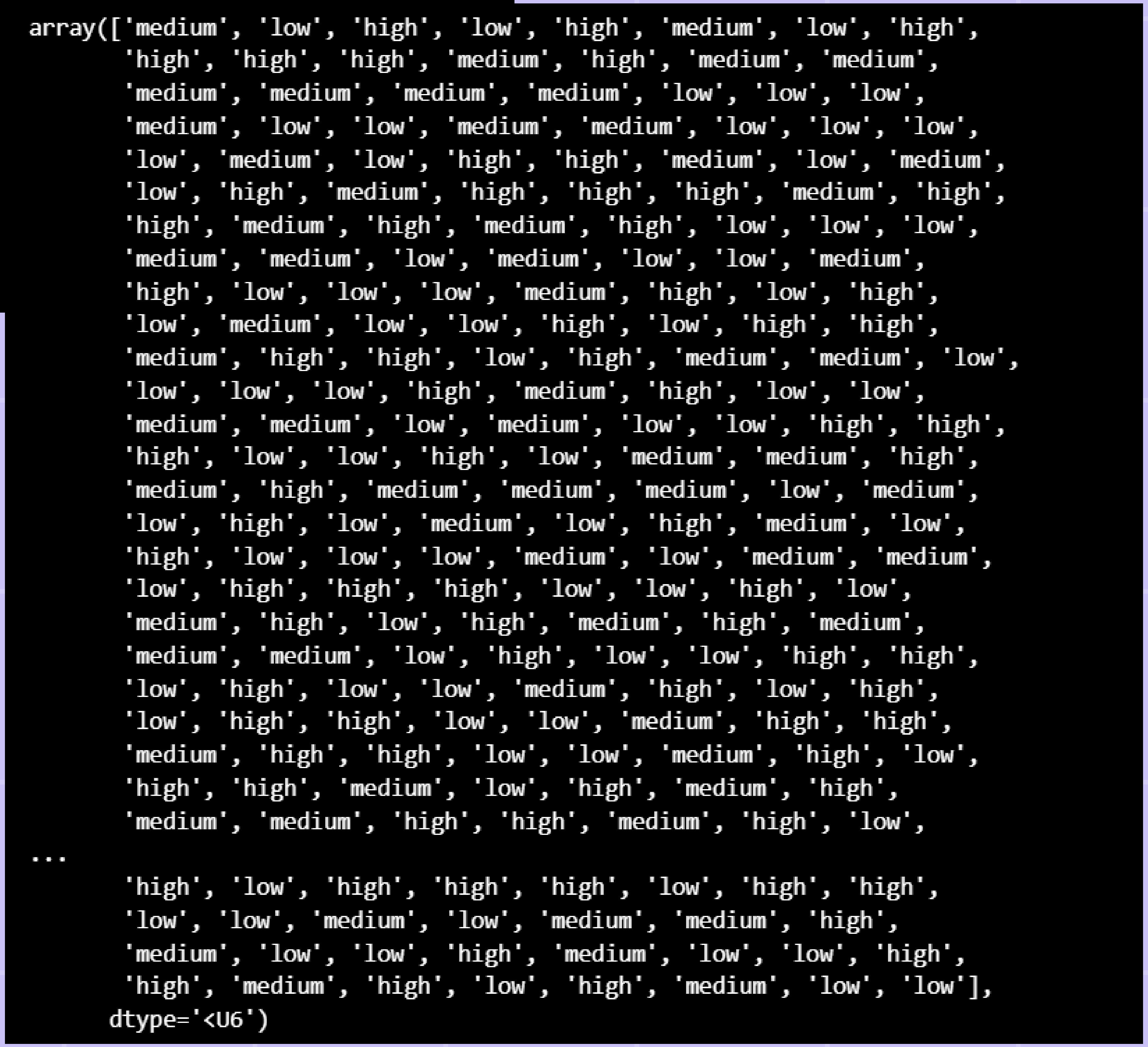


```
Lebel data:  ['high' 'high' 'low' ... 'low' 'high' 'low']
xtrain =  [[6.8 40 1 ... 1 0 245]
 [4.7 38 1 ... 1 0 255]
 [6.6 43 0 ... 1 0 250]
 ...
 [6.5 38 1 ... 0 0 255]
 [6.8 34 0 ... 0 1 240]
 [3.0 40 1 ... 1 1 255]]
xtest =  [[6.6 45 0 ... 0 1 250]
 [5.6 50 0 ... 1 1 255]
 [6.8 45 0 ... 1 1 255]
 ...
 [6.8 45 0 ... 0 1 255]
 [9.0 43 1 ... 1 1 250]
 [8.1 66 1 ... 1 1 255]]
ytrain =  ['medium' 'low' 'medium' 'high' 'high' 'low' 'medium' 'medium' 'low' 'low'
 'medium' 'medium' 'low' 'low' 'low' 'high' 'low' 'low' 'medium' 'medium'
 'low' 'high' 'low' 'high' 'high' 'low' 'medium' 'high' 'medium' 'medium'
 'medium' 'medium' 'low' 'low' 'low' 'low' 'low' 'low' 'low' 'high' 'low'
 'high' 'high' 'low' 'low' 'high' 'high' 'low' 'medium' 'low' 'high'
 'high' 'high' 'medium' 'low' 'medium' 'low' 'low' 'medium' 'low' 'low'
 'low' 'medium' 'medium' 'medium' 'low' 'high' 'low' 'medium' 'low' 'low'
 'low' 'low' 'low' 'high' 'high' 'medium' 'low' 'low' 'medium' 'low'
 'high' 'high' 'high' 'medium' 'high' 'high' 'low' 'high' 'medium'
 'medium' 'low' 'medium' 'low' 'high' 'low' 'medium' 'medium' 'high' 'low'
...
 'medium' 'medium' 'low' 'low' 'low' 'high' 'low' 'high' 'high' 'high'
 'low' 'high' 'high' 'low' 'low' 'low' 'low' 'medium' 'medium' 'high'
 'medium' 'low' 'low' 'medium' 'medium' 'low' 'low' 'high' 'medium'
 'medium' 'high' 'low' 'high' 'medium' 'low' 'low']
```

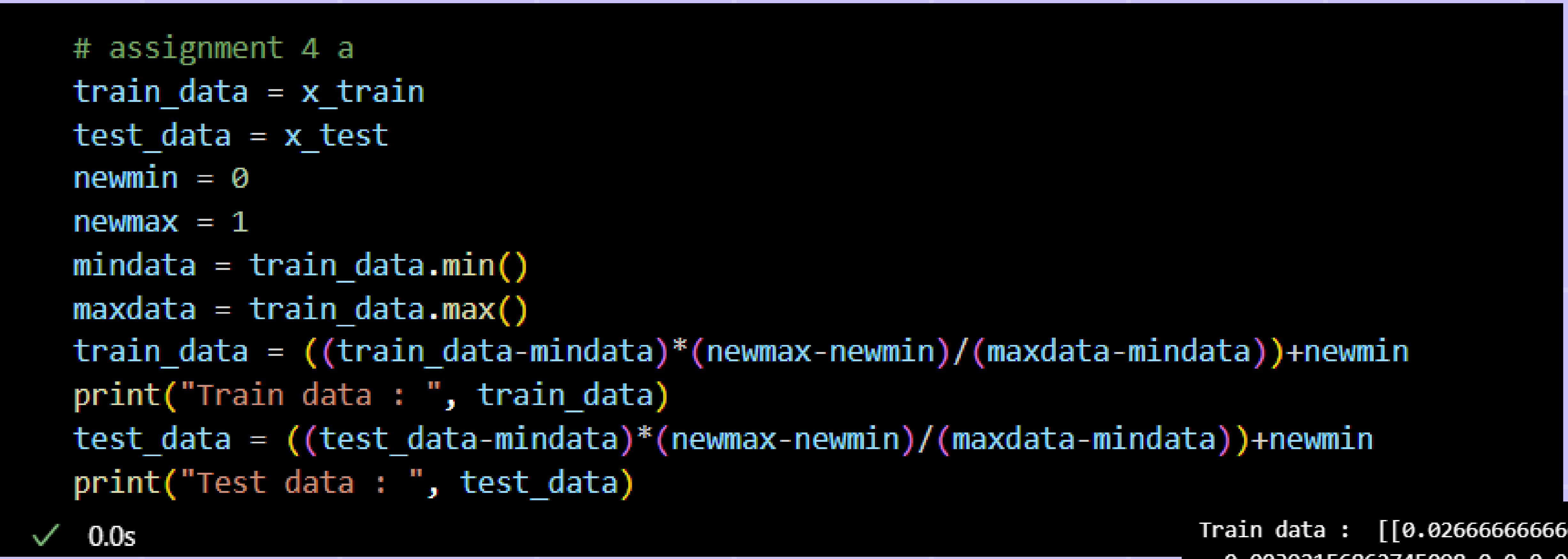
Output is truncated. View as a scrollable element or open in a text editor. Adjust cell output settings...



```python
# assignment 3 a
from sklearn.naive_bayes import GaussianNB as GNB

classifier = GNB()
classifier.fit(x_train,y_train)

ypredtn = classifier.predict(x_test)
ypredtn
```

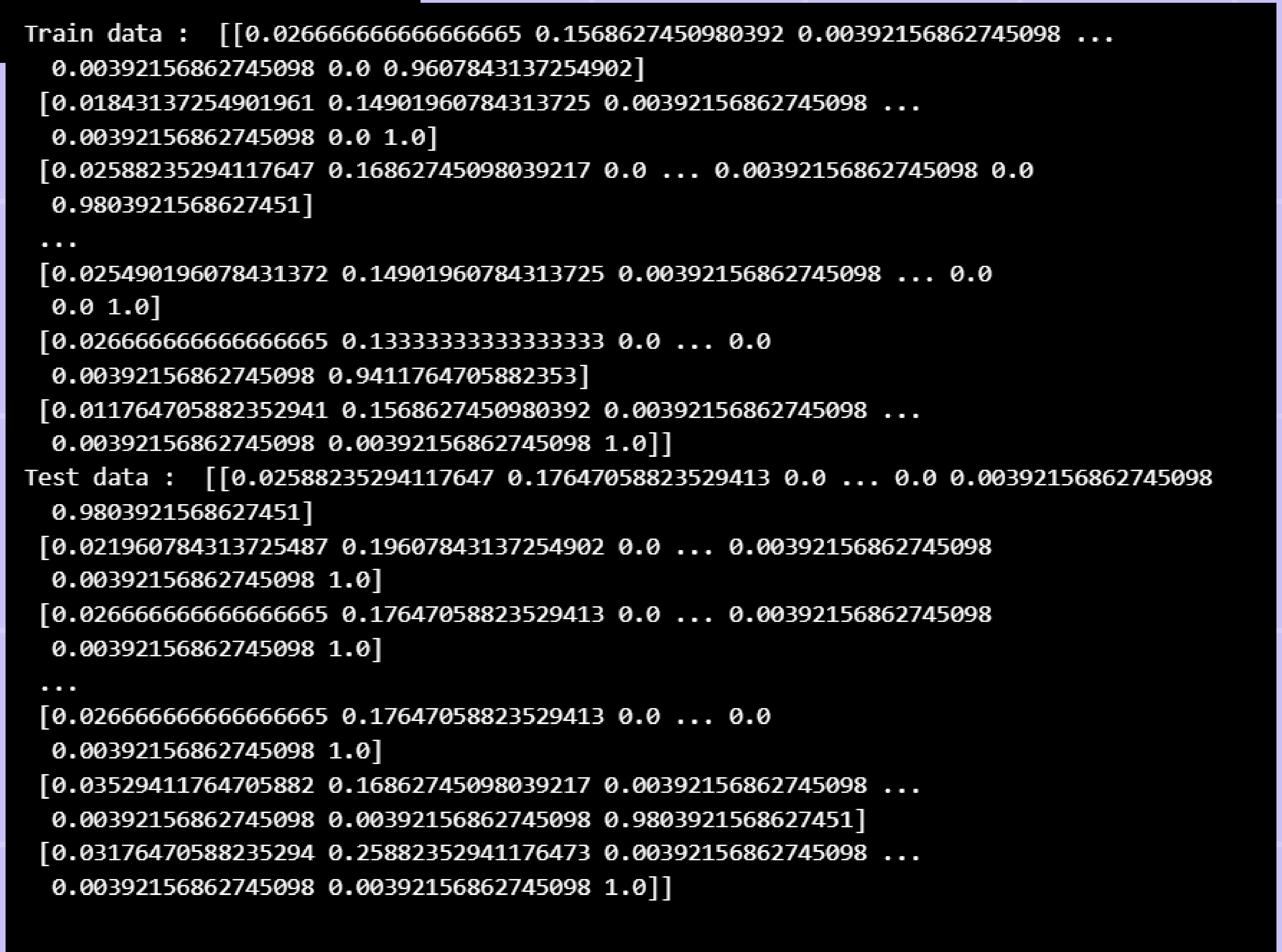


```
array(['medium', 'low', 'high', 'low', 'high', 'medium', 'low', 'high',
       'high', 'high', 'high', 'medium', 'high', 'medium', 'medium',
       'medium', 'medium', 'medium', 'medium', 'low', 'low', 'low',
       'medium', 'low', 'low', 'medium', 'medium', 'low', 'low', 'low',
       'low', 'medium', 'low', 'high', 'high', 'medium', 'low', 'medium',
       'low', 'high', 'medium', 'high', 'high', 'high', 'medium', 'high',
       'high', 'medium', 'high', 'medium', 'high', 'low', 'low', 'low',
       'medium', 'medium', 'low', 'medium', 'low', 'low', 'medium',
       'high', 'low', 'low', 'low', 'medium', 'high', 'low', 'high',
       'low', 'medium', 'low', 'low', 'high', 'low', 'high', 'high',
       'medium', 'high', 'high', 'low', 'high', 'medium', 'medium', 'low',
       'low', 'low', 'low', 'high', 'medium', 'high', 'low', 'low',
       'medium', 'medium', 'low', 'medium', 'low', 'low', 'high', 'high',
       'high', 'low', 'low', 'high', 'low', 'medium', 'medium', 'high',
       'medium', 'high', 'medium', 'medium', 'medium', 'low', 'medium',
       'low', 'high', 'low', 'medium', 'low', 'high', 'medium', 'low',
       'high', 'low', 'low', 'low', 'medium', 'low', 'medium', 'medium',
       'low', 'high', 'high', 'high', 'low', 'low', 'high', 'low',
       'medium', 'high', 'low', 'high', 'medium', 'high', 'medium',
       'medium', 'medium', 'low', 'high', 'low', 'low', 'high', 'high',
       'low', 'high', 'low', 'low', 'medium', 'high', 'low', 'high',
       'low', 'high', 'high', 'low', 'low', 'medium', 'high', 'high',
       'medium', 'high', 'high', 'low', 'low', 'medium', 'high', 'low',
       'high', 'high', 'medium', 'low', 'high', 'medium', 'high',
       'medium', 'medium', 'high', 'high', 'medium', 'high', 'low',
...
       'high', 'low', 'high', 'high', 'high', 'low', 'high', 'high',
       'low', 'low', 'medium', 'low', 'medium', 'medium', 'high',
       'medium', 'low', 'low', 'high', 'medium', 'low', 'low', 'high',
       'high', 'medium', 'high', 'low', 'high', 'medium', 'low', 'low'],
      dtype='<U6')
```

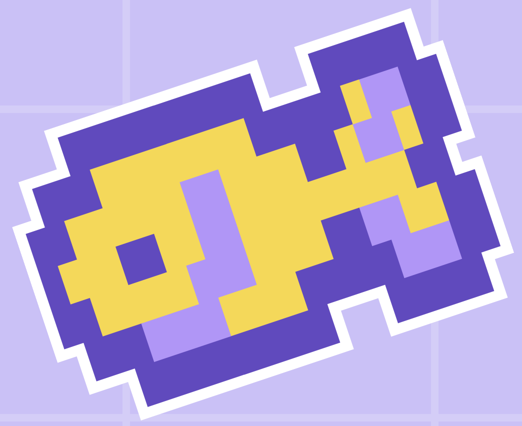


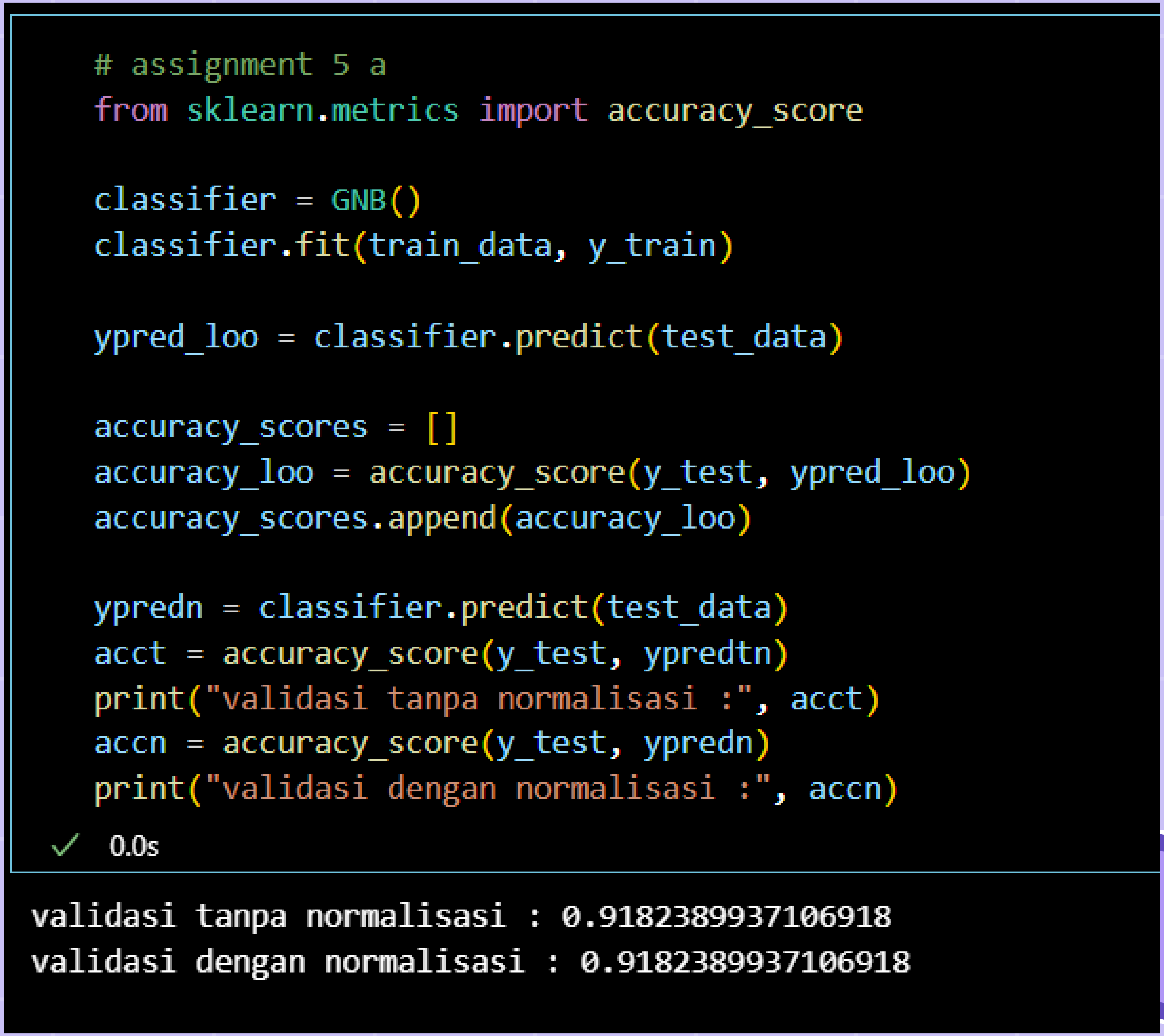
```
# assignment 4 a
train_data = x_train
test_data = x_test
newmin = 0
newmax = 1
mindata = train_data.min()
maxdata = train_data.max()
train_data = ((train_data-mindata)*(newmax-newmin)/(maxdata-mindata))+newmin
print("Train data : ", train_data)
test_data = ((test_data-mindata)*(newmax-newmin)/(maxdata-mindata))+newmin
print("Test data : ", test_data)
```

✓ 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 ...
  0.00392156862745098 0.00392156862745098 1.0]]
```

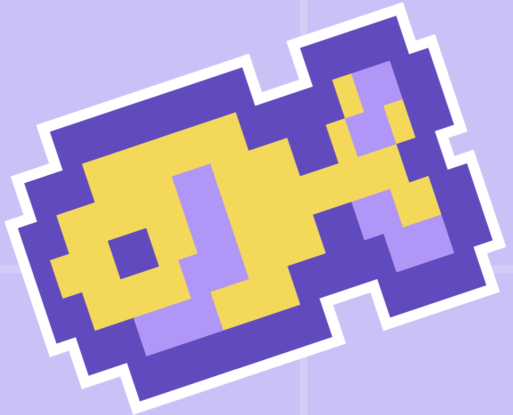


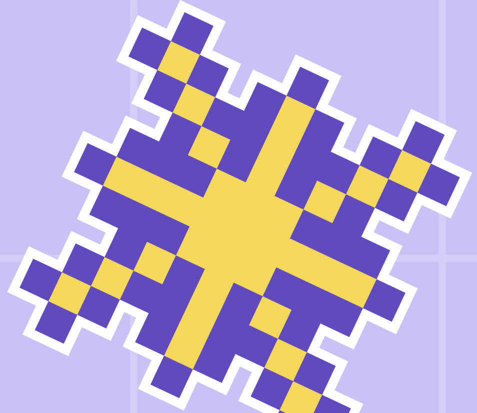


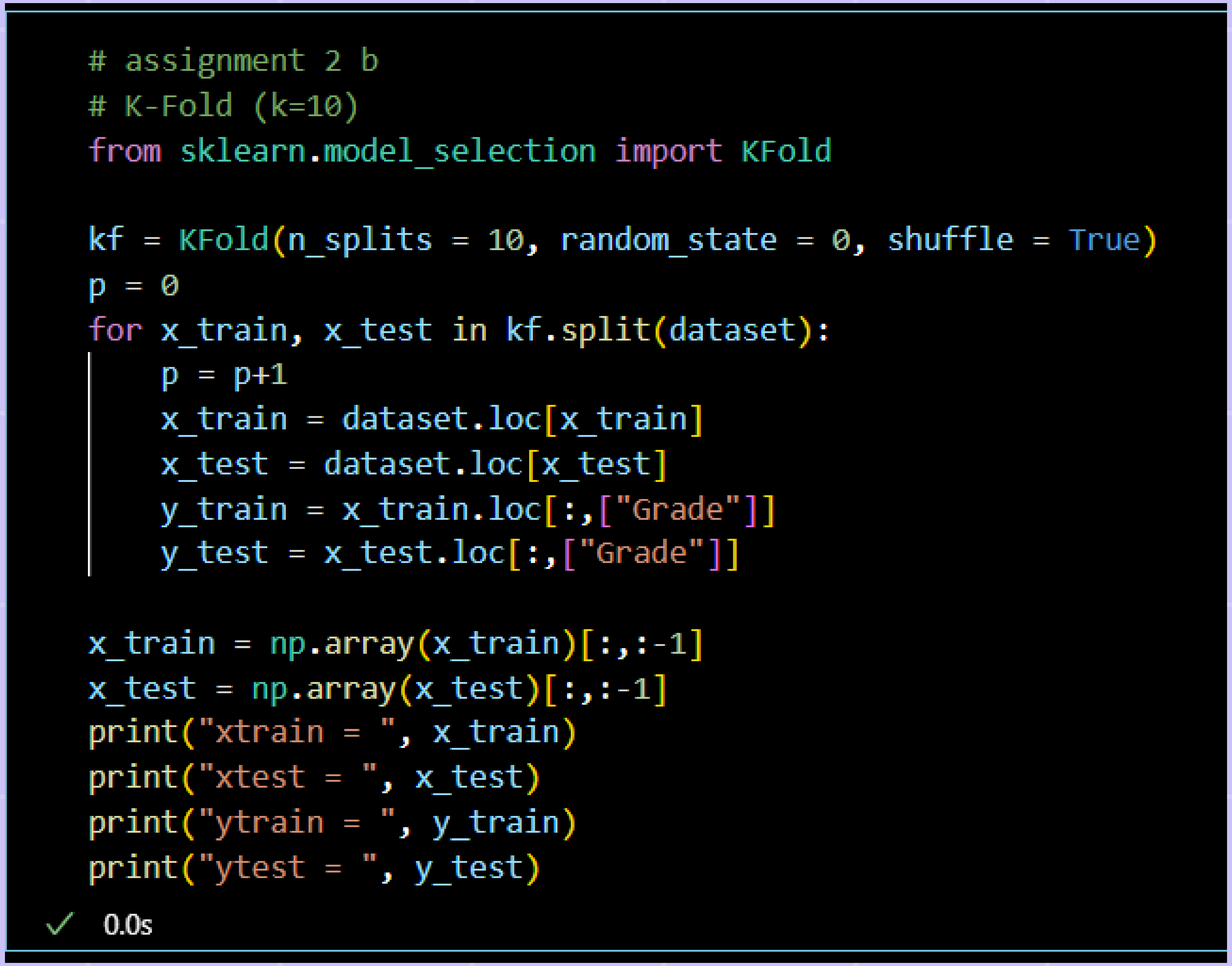
```
# assignment 5 a
from sklearn.metrics import accuracy_score

classifier = GNB()
classifier.fit(train_data, y_train)

ypred_loo = classifier.predict(test_data)

accuracy_scores = []
accuracy_loo = accuracy_score(y_test, ypred_loo)
accuracy_scores.append(accuracy_loo)

ypredn = classifier.predict(test_data)
acct = accuracy_score(y_test, ypredtn)
print("validasi tanpa normalisasi :", acct)
accn = accuracy_score(y_test, ypredn)
print("validasi dengan normalisasi :", accn)
```

✓ 0.0s

```
validasi tanpa normalisasi : 0.9182389937106918
validasi dengan normalisasi : 0.9182389937106918
```

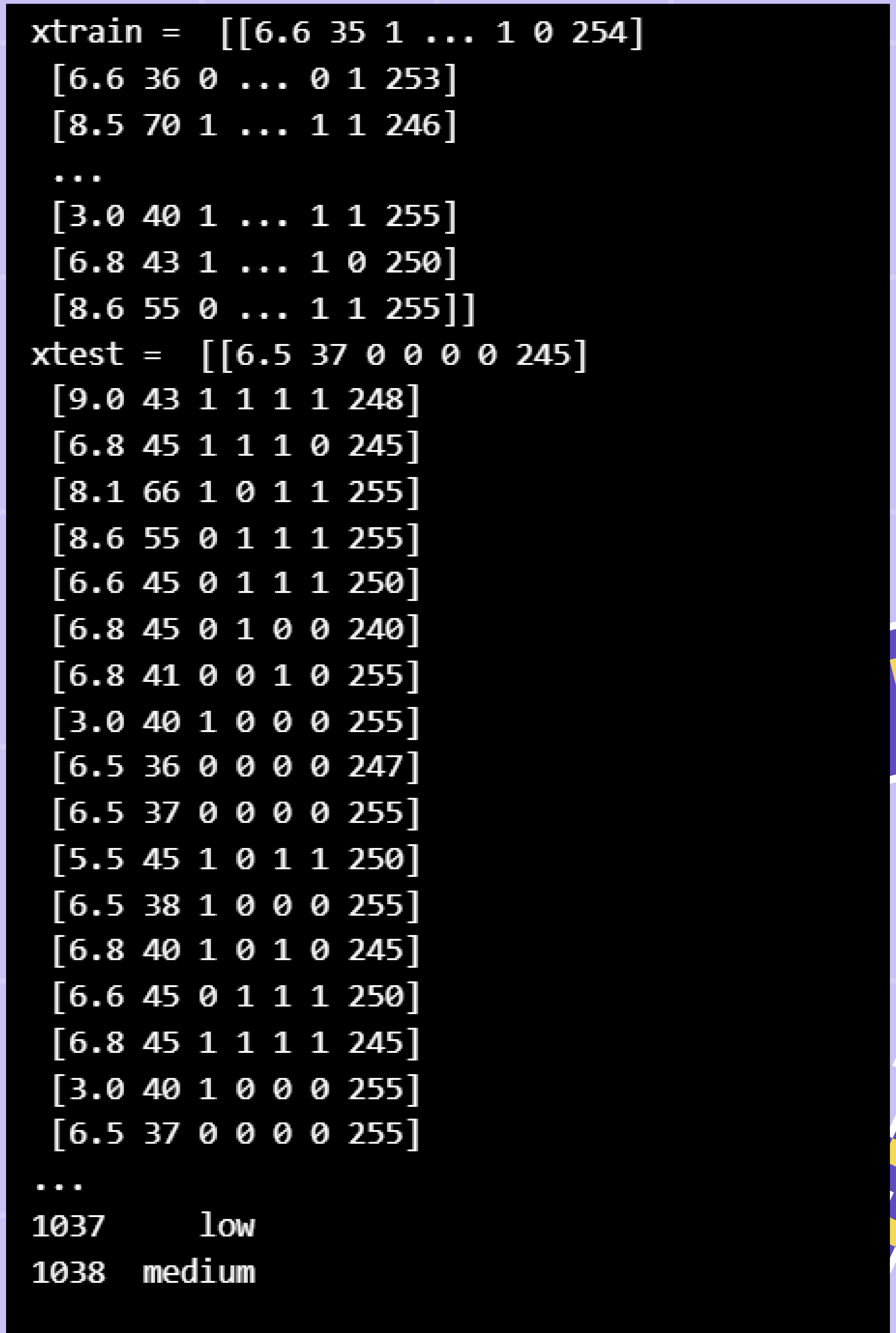




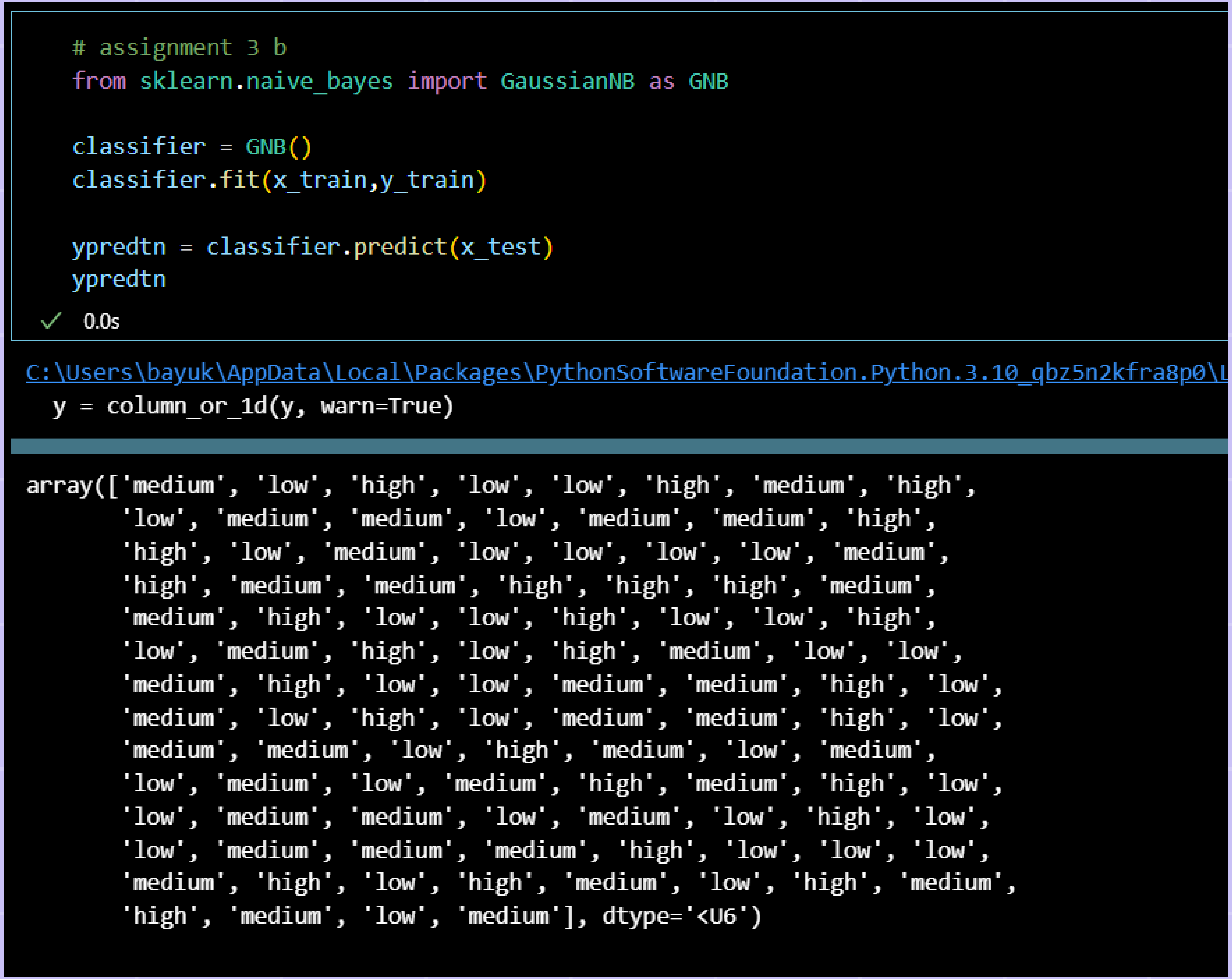


```python
# assignment 2 b
# K-Fold (k=10)
from sklearn.model_selection import KFold

kf = KFold(n_splits = 10, random_state = 0, shuffle = True)
p = 0
for x_train, x_test in kf.split(dataset):
    p = p+1
    x_train = dataset.loc[x_train]
    x_test = dataset.loc[x_test]
    y_train = x_train.loc[:,["Grade"]]
    y_test = x_test.loc[:,["Grade"]]

x_train = np.array(x_train)[:,:-1]
x_test = np.array(x_test)[:,:-1]
print("xtrain = ", x_train)
print("xtest = ", x_test)
print("ytrain = ", y_train)
print("ytest = ", y_test)
```

✓ 0.0s



```
xtrain =  [[6.6 35 1 ... 1 0 254]
 [6.6 36 0 ... 0 1 253]
 [8.5 70 1 ... 1 1 246]
 ...
 [3.0 40 1 ... 1 1 255]
 [6.8 43 1 ... 1 0 250]
 [8.6 55 0 ... 1 1 255]]
xtest =  [[6.5 37 0 0 0 0 245]
 [9.0 43 1 1 1 1 248]
 [6.8 45 1 1 1 0 245]
 [8.1 66 1 0 1 1 255]
 [8.6 55 0 1 1 1 255]
 [6.6 45 0 1 1 1 250]
 [6.8 45 0 1 0 0 240]
 [6.8 41 0 0 1 0 255]
 [3.0 40 1 0 0 0 255]
 [6.5 36 0 0 0 0 247]
 [6.5 37 0 0 0 0 255]
 [5.5 45 1 0 1 1 250]
 [6.5 38 1 0 0 0 255]
 [6.8 40 1 0 1 0 245]
 [6.6 45 0 1 1 1 250]
 [6.8 45 1 1 1 1 245]
 [3.0 40 1 0 0 0 255]
 [6.5 37 0 0 0 0 255]
...
1037      low
1038   medium
```

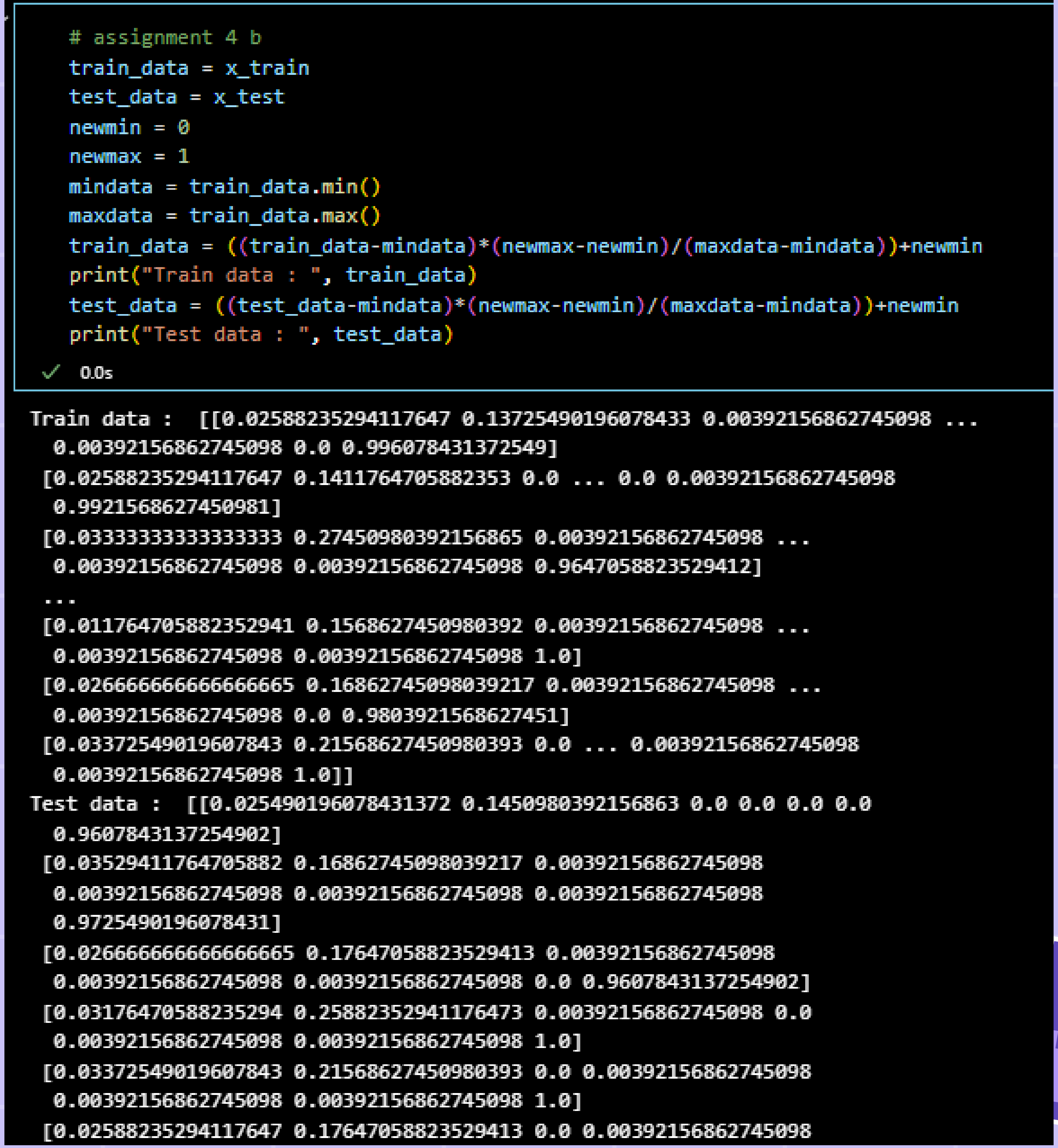


```
# assignment 3 b
from sklearn.naive_bayes import GaussianNB as GNB

classifier = GNB()
classifier.fit(x_train,y_train)

ypredtn = classifier.predict(x_test)
ypredtn
```

✓ 0.0s

```
C:\Users\bayuk\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.10_qbz5n2kfra8p0\L
  y = column_or_1d(y, warn=True)
```

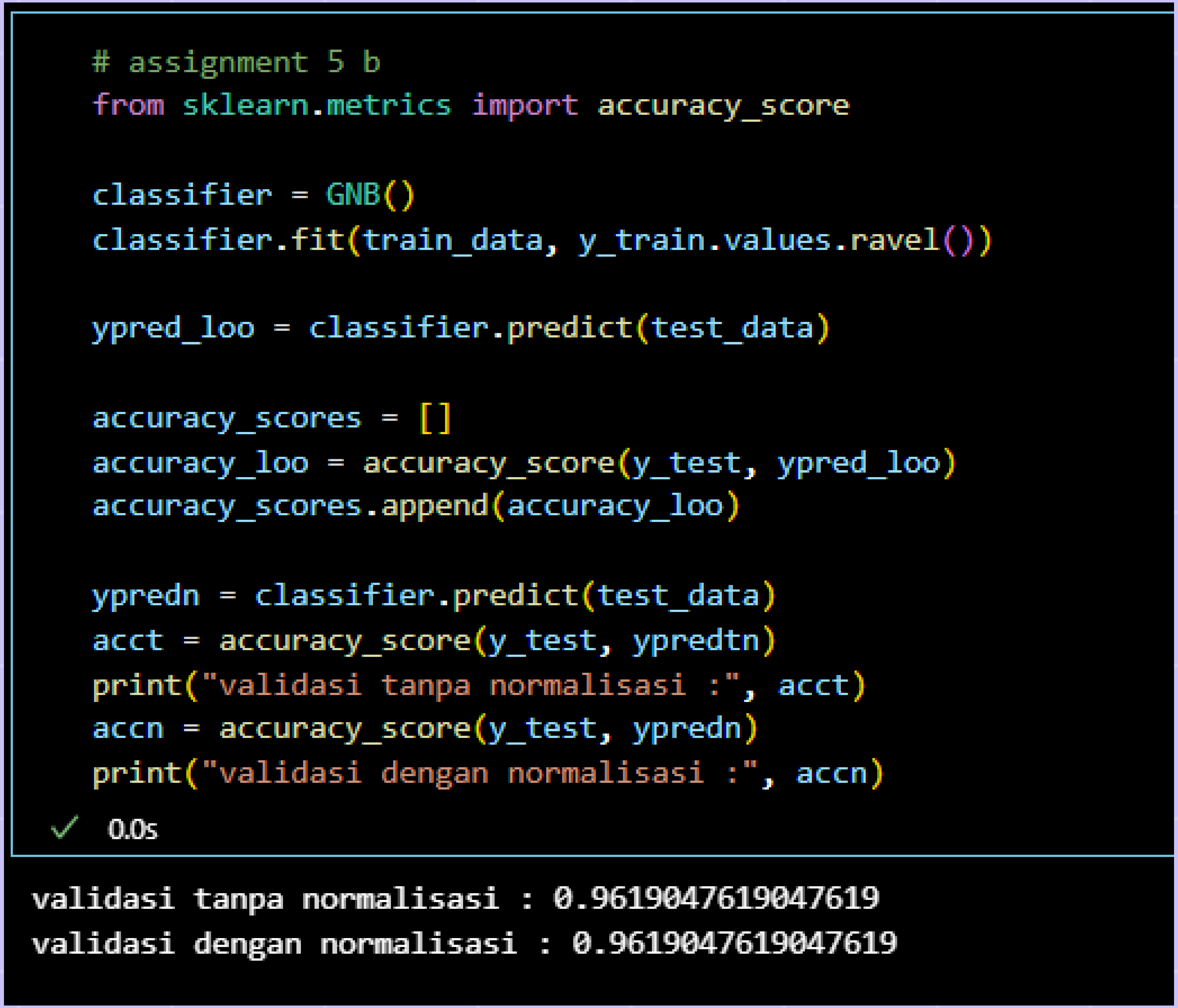
```
array(['medium', 'low', 'high', 'low', 'low', 'high', 'medium', 'high',
       'low', 'medium', 'medium', 'low', 'medium', 'medium', 'high',
       'high', 'low', 'medium', 'low', 'low', 'low', 'low', 'medium',
       'high', 'medium', 'medium', 'high', 'high', 'high', 'medium',
       'medium', 'high', 'low', 'low', 'high', 'low', 'low', 'high',
       'low', 'medium', 'high', 'low', 'high', 'medium', 'low', 'low',
       'medium', 'high', 'low', 'low', 'medium', 'medium', 'high', 'low',
       'medium', 'low', 'high', 'low', 'medium', 'medium', 'high', 'low',
       'medium', 'medium', 'low', 'high', 'medium', 'low', 'medium',
       'low', 'medium', 'low', 'medium', 'high', 'medium', 'high', 'low',
       'low', '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'], dtype='<U6')
```



```
# assignment 4 b
train_data = x_train
test_data = x_test
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)
test_data = ((test_data-mindata)*(newmax-newmin)/(maxdata-mindata))+newmin
print("Test data : ", test_data)
```

✓ 0.0s

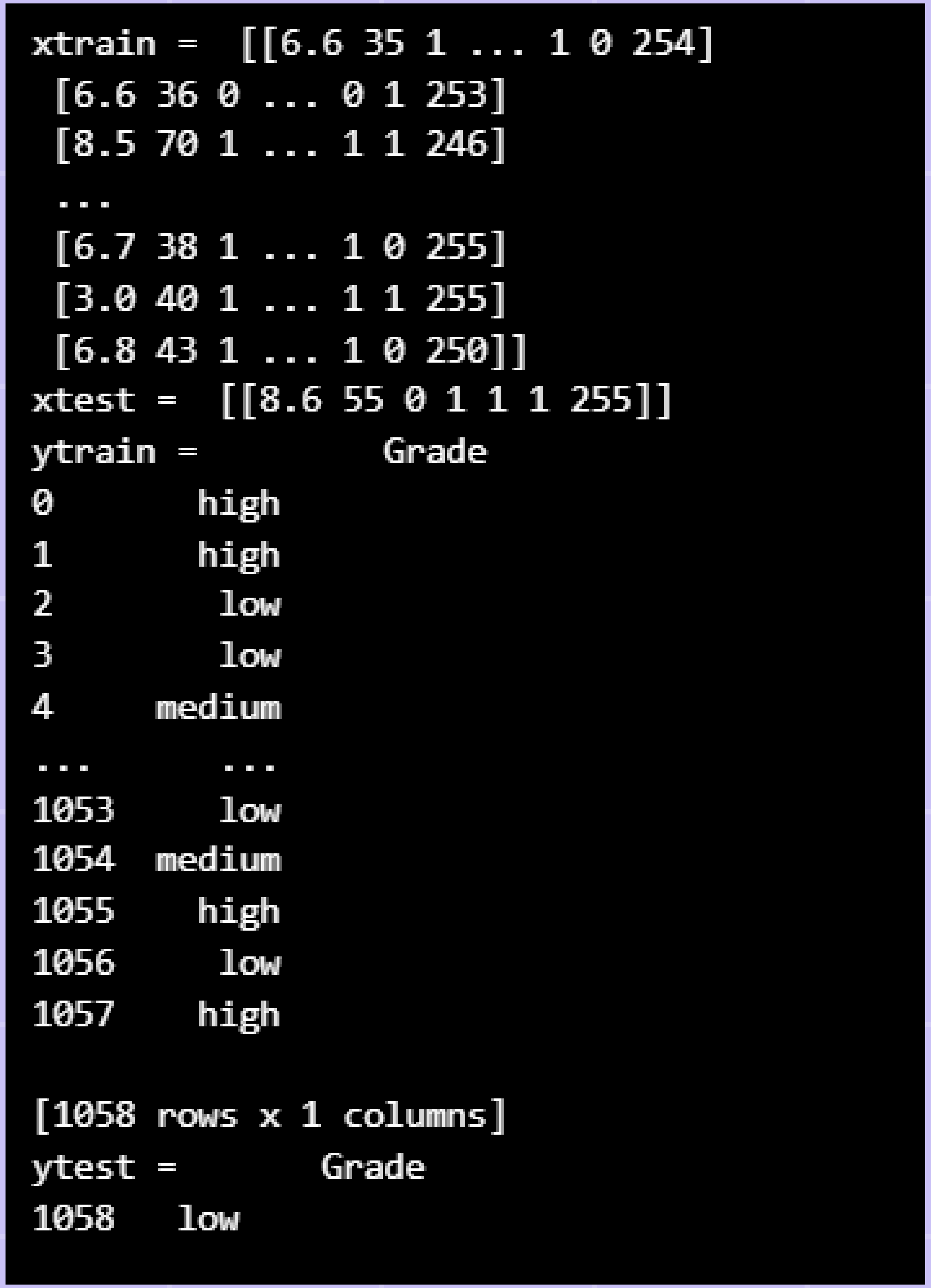
```
Train data :  [[0.025882352941176 47 0.13725490196078433 0.00392156862745098 ...
  0.00392156862745098 0.0 0.996078431372549]
 [0.02588235294117647 0.1411764705882353 0.0 ... 0.0 0.00392156862745098
  0.9921568627450981]
 [0.0333333333333333 0.27450980392156865 0.00392156862745098 ...
  0.00392156862745098 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
```



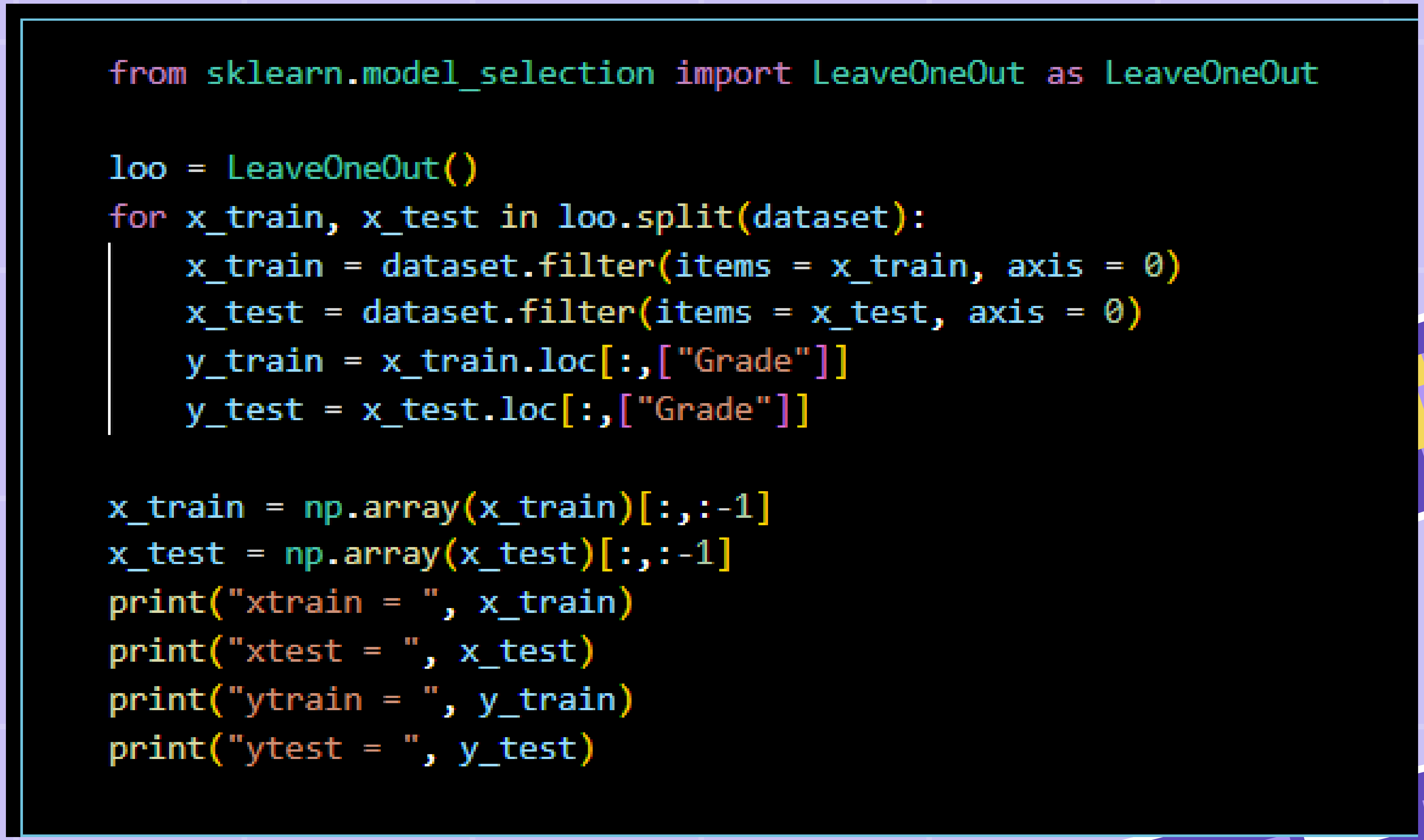
```python
# assignment 5 b
from sklearn.metrics import accuracy_score

classifier = GNB()
classifier.fit(train_data, y_train.values.ravel())

ypred_loo = classifier.predict(test_data)

accuracy_scores = []
accuracy_loo = accuracy_score(y_test, ypred_loo)
accuracy_scores.append(accuracy_loo)

ypredn = classifier.predict(test_data)
acct = accuracy_score(y_test, ypredtn)
print("validasi tanpa normalisasi :", acct)
accn = accuracy_score(y_test, ypredn)
print("validasi dengan normalisasi :", accn)
```

✓ 0.0s

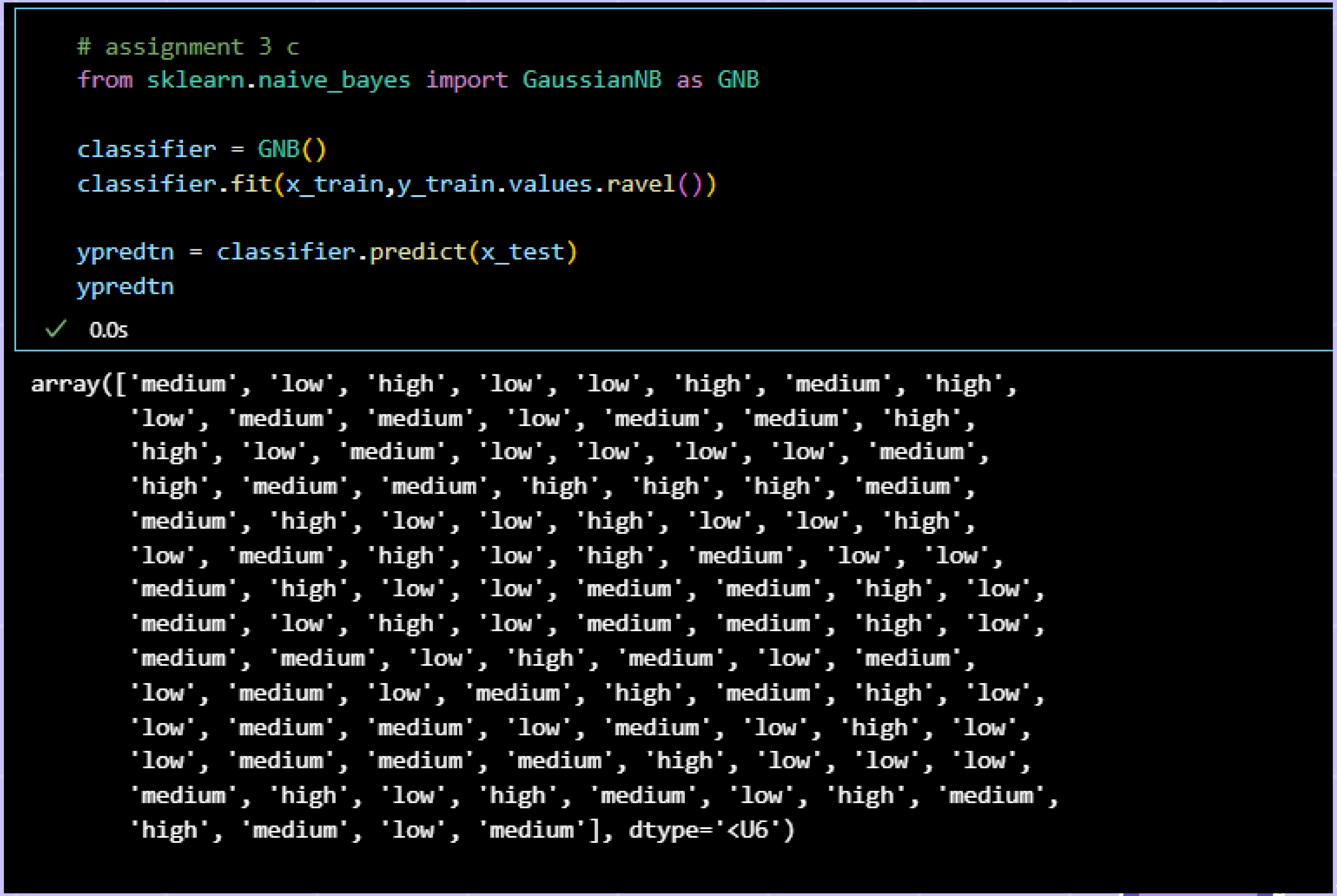
```
validasi tanpa normalisasi : 0.9619047619047619
validasi dengan normalisasi : 0.9619047619047619
```



```
xtrain =  [[6.6 35 1 ... 1 0 254]
 [6.6 36 0 ... 0 1 253]
 [8.5 70 1 ... 1 1 246]
 ...
 [6.7 38 1 ... 1 0 255]
 [3.0 40 1 ... 1 1 255]
 [6.8 43 1 ... 1 0 250]]
xtest =  [[8.6 55 0 1 1 1 255]]
ytrain =         Grade
0        high
1        high
2         low
3         low
4      medium
...       ...
1053      low
1054   medium
1055     high
1056      low
1057     high

[1058 rows x 1 columns]
ytest =        Grade
1058    low
```



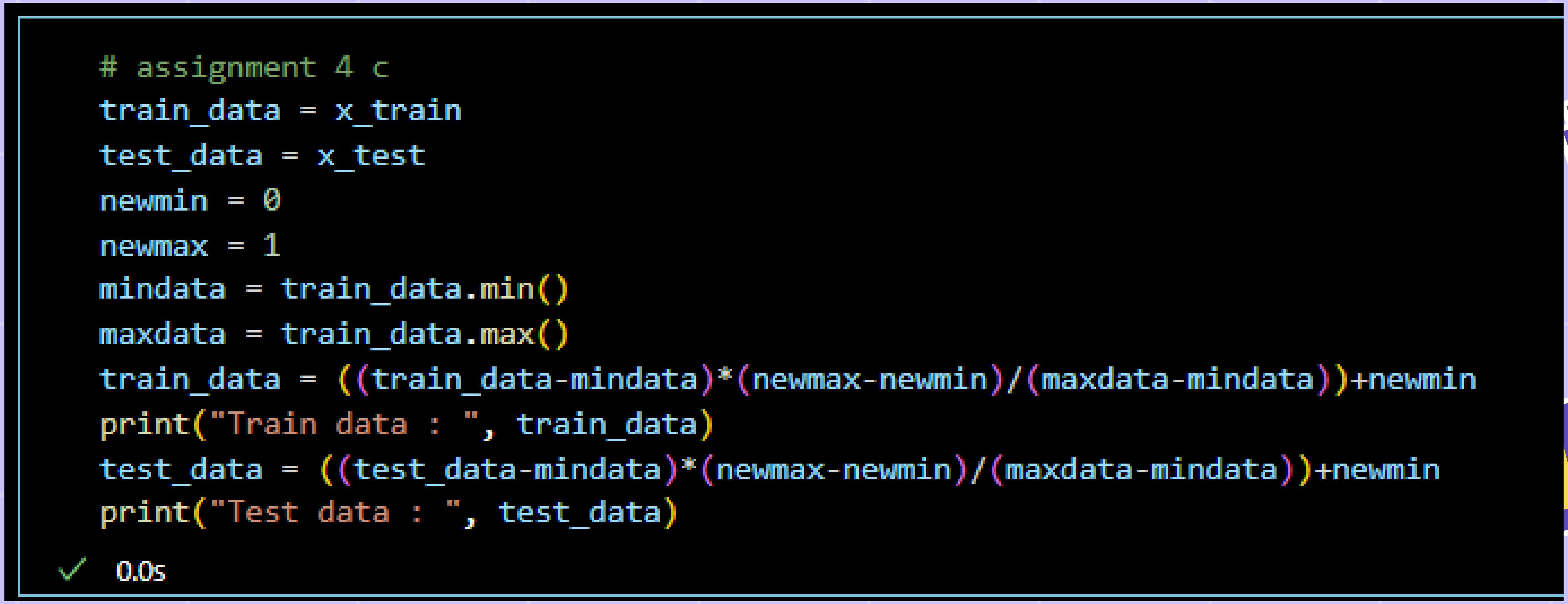
```python
from sklearn.model_selection import LeaveOneOut as LeaveOneOut

loo = LeaveOneOut()
for x_train, x_test in loo.split(dataset):
    x_train = dataset.filter(items = x_train, axis = 0)
    x_test = dataset.filter(items = x_test, axis = 0)
    y_train = x_train.loc[:,["Grade"]]
    y_test = x_test.loc[:,["Grade"]]

x_train = np.array(x_train)[:,:-1]
x_test = np.array(x_test)[:,:-1]
print("xtrain = ", x_train)
print("xtest = ", x_test)
print("ytrain = ", y_train)
print("ytest = ", y_test)
```



```
# assignment 3 c
from sklearn.naive_bayes import GaussianNB as GNB

classifier = GNB()
classifier.fit(x_train,y_train.values.ravel())

ypredtn = classifier.predict(x_test)
ypredtn
```

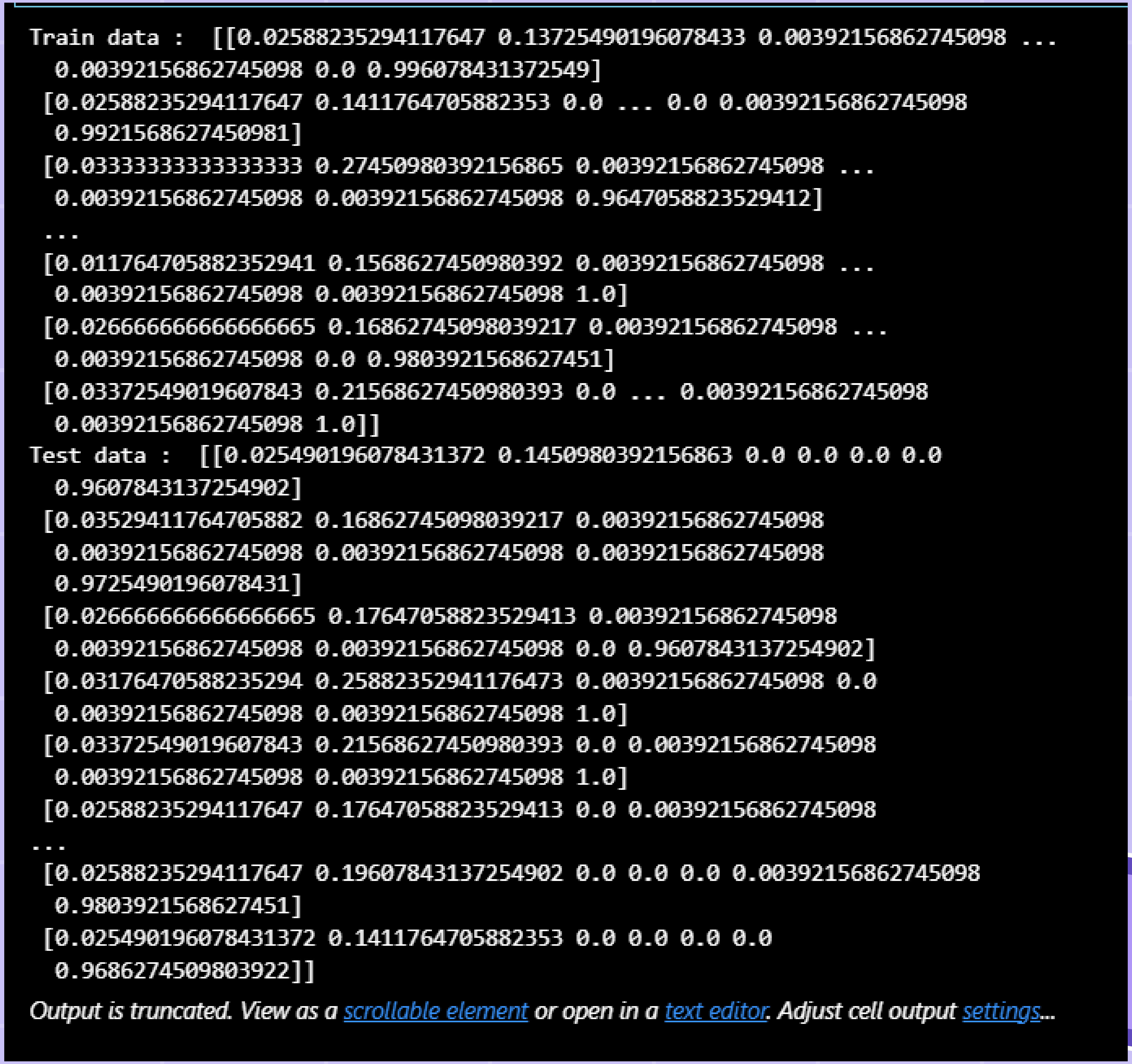
✓ 0.0s

```
array(['medium', 'low', 'high', 'low', 'low', 'high', 'medium', 'high',
       'low', 'medium', 'medium', 'low', 'medium', 'medium', 'high',
       'high', 'low', 'medium', 'low', 'low', 'low', 'low', 'medium',
       'high', 'medium', 'medium', 'high', 'high', 'high', 'medium',
       'medium', 'high', 'low', 'low', 'high', 'low', 'low', 'high',
       'low', 'medium', 'high', 'low', 'high', 'medium', 'low', 'low',
       'medium', 'high', 'low', 'low', 'medium', 'medium', 'high', 'low',
       'medium', 'low', 'high', 'low', 'medium', 'medium', 'high', 'low',
       'medium', 'medium', 'low', 'high', 'medium', 'low', 'medium',
       'low', 'medium', 'low', 'medium', 'high', 'medium', 'high', 'low',
       'low', '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'], dtype='<U6')
```



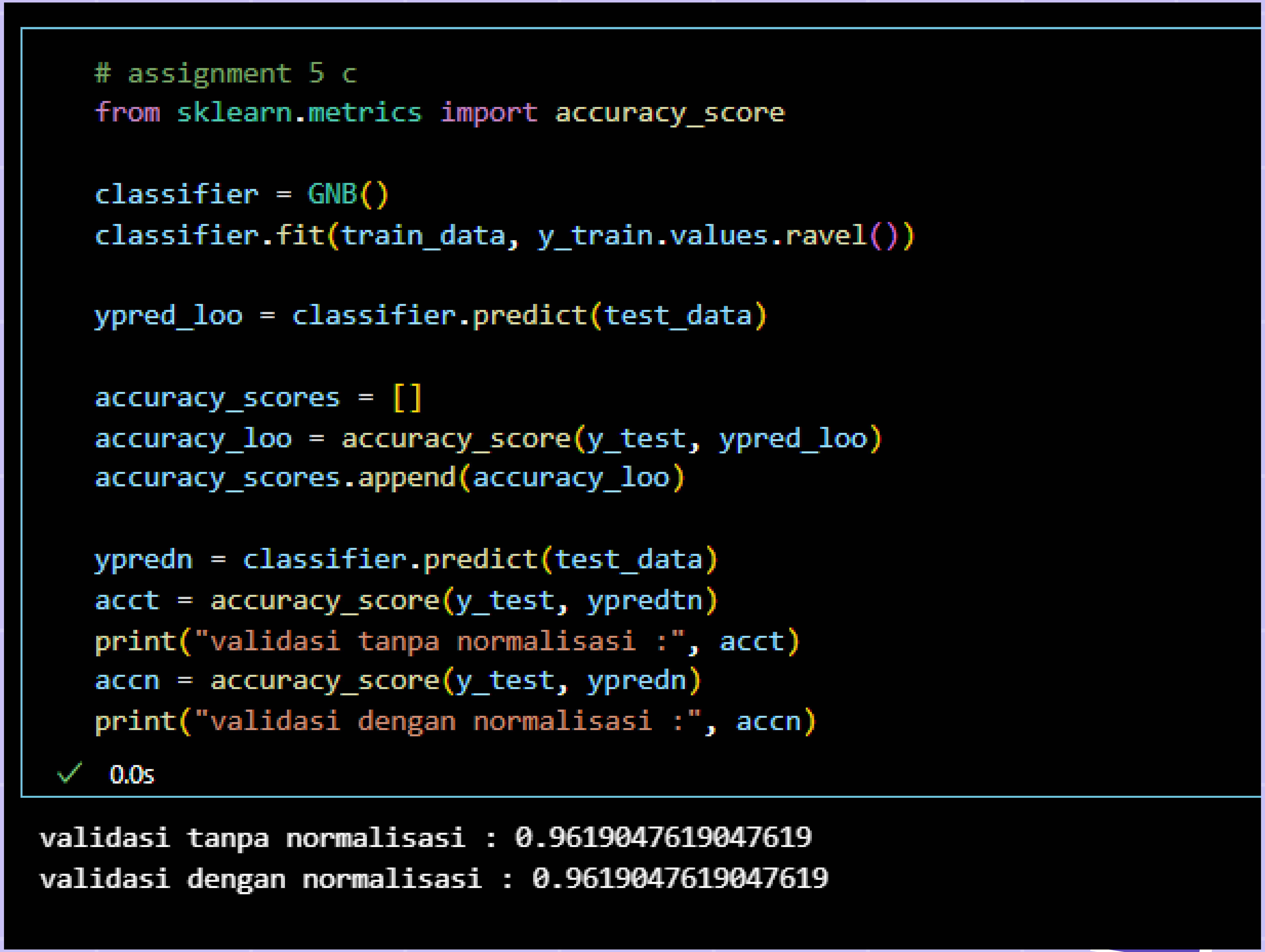
```
# assignment 4 c
train_data = x_train
test_data = x_test
newmin = 0
newmax = 1
mindata = train_data.min()
maxdata = train_data.max()
train_data = ((train_data-mindata)*(newmax-newmin)/(maxdata-mindata))+newmin
print("Train data : ", train_data)
test_data = ((test_data-mindata)*(newmax-newmin)/(maxdata-mindata))+newmin
print("Test data : ", test_data)
```

✓ 0.0s



```
Train data :  [[0.025882352941176 47 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
...
 [0.02588235294117647 0.19607843137254902 0.0 0.0 0.0 0.00392156862745098
  0.9803921568627451]
 [0.025490196078431372 0.1411764705882353 0.0 0.0 0.0 0.0
  0.9686274509803922]]
```

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



```python
# assignment 5 c
from sklearn.metrics import accuracy_score

classifier = GNB()
classifier.fit(train_data, y_train.values.ravel())

ypred_loo = classifier.predict(test_data)

accuracy_scores = []
accuracy_loo = accuracy_score(y_test, ypred_loo)
accuracy_scores.append(accuracy_loo)

ypredn = classifier.predict(test_data)
acct = accuracy_score(y_test, ypredtn)
print("validasi tanpa normalisasi :", acct)
accn = accuracy_score(y_test, ypredn)
print("validasi dengan normalisasi :", accn)
```

✓ 0.0s

```
validasi tanpa normalisasi : 0.9619047619047619
validasi dengan normalisasi : 0.9619047619047619
```