WORKSHOP MACHINE LEARNING

Klasifikasi using K-Nearest Neighbor

1. Import library dan gunakan dataset iris.csv

```
import pandas as pd
from sklearn.neighbors import KNeighborsClassifier
from sklearn import svm
from sklearn.naive_bayes import GaussianNB
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score
from sklearn.metrics import confusion_matrix
from sklearn.metrics import classification_report
```

```
data = pd.read_csv('Iris.csv')
print(data.head(10))
```

```
   Id  SepalLengthCm  SepalWidthCm  PetalLengthCm  PetalWidthCm      Species
0   1            5.1           3.5            1.4           0.2  Iris-setosa
1   2            4.9           3.0            1.4           0.2  Iris-setosa
2   3            4.7           3.2            1.3           0.2  Iris-setosa
3   4            4.6           3.1            1.5           0.2  Iris-setosa
4   5            5.0           3.6            1.4           0.2  Iris-setosa
5   6            5.4           3.9            1.7           0.4  Iris-setosa
6   7            4.6           3.4            1.4           0.3  Iris-setosa
7   8            5.0           3.4            1.5           0.2  Iris-setosa
8   9            4.4           2.9            1.4           0.2  Iris-setosa
9  10            4.9           3.1            1.5           0.1  Iris-setosa
```

2. Menampilkan data X yaitu atribut dan Y yaitu kelas.

```
[4]: X = data.iloc[:,:-1].values
     y=data['Species']
```

```
[5]: X
```

```
[5]: array([[1.00e+00, 5.10e+00, 3.50e+00, 1.40e+00, 2.00e-01],
            [2.00e+00, 4.90e+00, 3.00e+00, 1.40e+00, 2.00e-01],
            [3.00e+00, 4.70e+00, 3.20e+00, 1.30e+00, 2.00e-01],
            [4.00e+00, 4.60e+00, 3.10e+00, 1.50e+00, 2.00e-01],
            [5.00e+00, 5.00e+00, 3.60e+00, 1.40e+00, 2.00e-01],
            [6.00e+00, 5.40e+00, 3.90e+00, 1.70e+00, 4.00e-01],
            [7.00e+00, 4.60e+00, 3.40e+00, 1.40e+00, 3.00e-01],
            [8.00e+00, 5.00e+00, 3.40e+00, 1.50e+00, 2.00e-01],
            [9.00e+00, 4.40e+00, 2.90e+00, 1.40e+00, 2.00e-01],
            [1.00e+01, 4.90e+00, 3.10e+00, 1.50e+00, 1.00e-01],
            [1.10e+01, 5.40e+00, 3.70e+00, 1.50e+00, 2.00e-01],
            [1.20e+01, 4.80e+00, 3.40e+00, 1.60e+00, 2.00e-01],
            [1.30e+01, 4.80e+00, 3.00e+00, 1.40e+00, 1.00e-01],
            [1.40e+01, 4.30e+00, 3.00e+00, 1.10e+00, 1.00e-01],
            [1.50e+01, 5.80e+00, 4.00e+00, 1.20e+00, 2.00e-01],
            [1.60e+01, 5.70e+00, 4.40e+00, 1.50e+00, 4.00e-01],
            [1.70e+01, 5.40e+00, 3.90e+00, 1.30e+00, 4.00e-01],
            [1.80e+01, 5.10e+00, 3.50e+00, 1.40e+00, 3.00e-01],
```

```
[6]: y
```

```
[6]: 0         Iris-setosa
     1         Iris-setosa
     2         Iris-setosa
     3         Iris-setosa
     4         Iris-setosa
               ...
     145    Iris-virginica
     146    Iris-virginica
     147    Iris-virginica
     148    Iris-virginica
     149    Iris-virginica
     Name: Species, Length: 150, dtype: object
```

3. Split data training dan testing

```
X_train, X_test, y_train, y_test = train_test_split(X,y, test_size=0.2, random_state=27)
```

```
print(X_train)
print(y_train)
```

```
[4.40e+01 5.00e+00 3.50e+00 1.60e+00 6.00e-01]
[1.10e+01 5.40e+00 3.70e+00 1.50e+00 2.00e-01]
[1.00e+00 5.10e+00 3.50e+00 1.40e+00 2.00e-01]
[8.10e+01 5.50e+00 2.40e+00 3.80e+00 1.10e+00]
[2.80e+01 5.20e+00 3.50e+00 1.50e+00 2.00e-01]
[8.80e+01 6.30e+00 2.30e+00 4.40e+00 1.30e+00]
[1.90e+01 5.70e+00 3.80e+00 1.70e+00 3.00e-01]
[1.41e+02 6.70e+00 3.10e+00 5.60e+00 2.40e+00]
[1.20e+01 4.80e+00 3.40e+00 1.60e+00 2.00e-01]
[4.60e+01 4.80e+00 3.00e+00 1.40e+00 3.00e-01]
[5.50e+01 6.50e+00 2.80e+00 4.60e+00 1.50e+00]
[1.43e+02 5.80e+00 2.70e+00 5.10e+00 1.90e+00]
[1.27e+02 6.20e+00 2.80e+00 4.80e+00 1.80e+00]
[9.60e+01 5.70e+00 3.00e+00 4.20e+00 1.20e+00]
[1.03e+02 7.10e+00 3.00e+00 5.90e+00 2.10e+00]
[1.45e+02 6.70e+00 3.30e+00 5.70e+00 2.50e+00]
[9.00e+01 5.50e+00 2.50e+00 4.00e+00 1.30e+00]
[1.38e+02 6.40e+00 3.10e+00 5.50e+00 1.80e+00]
[1.33e+02 6.40e+00 2.80e+00 5.60e+00 2.20e+00]
```

```
KNN_model = KNeighborsClassifier(n_neighbors=5)
```

```
KNN_model.fit(X_train, y_train)
```

```
KNeighborsClassifier
KNeighborsClassifier()
```

```
KNN_prediction = KNN_model.predict(X_test)
```

```
print(accuracy_score(KNN_prediction, y_test))
print(classification_report(KNN_prediction, y_test))
```

4. Output

```
1.0
                precision    recall  f1-score   support

    Iris-setosa       1.00      1.00      1.00         7
Iris-versicolor       1.00      1.00      1.00        11
 Iris-virginica       1.00      1.00      1.00        12

       accuracy                           1.00        30
      macro avg       1.00      1.00      1.00        30
   weighted avg       1.00      1.00      1.00        30
```