

Machine Learning KNN
Tugas Laporan Pertemuan 4



Dibuat oleh :

Adhyasta Adwaya Alkarim (5230411165)

Kelas : IV Machine Learning Praktikum

PROGAM STUDI INFORMATIKA FALKUTAS SAINTEK

UNIVERSITAS TEKNOLOGI YOGYAKARTA

2025/2026

```
[ ] import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
```

Step awal kita perlu import library dari python yaitu numpy as np untuk menghitung angka numerik, import matplotlib untuk output gambar diagram dari hasil analisis data, import pandas untuk import data set ke colab

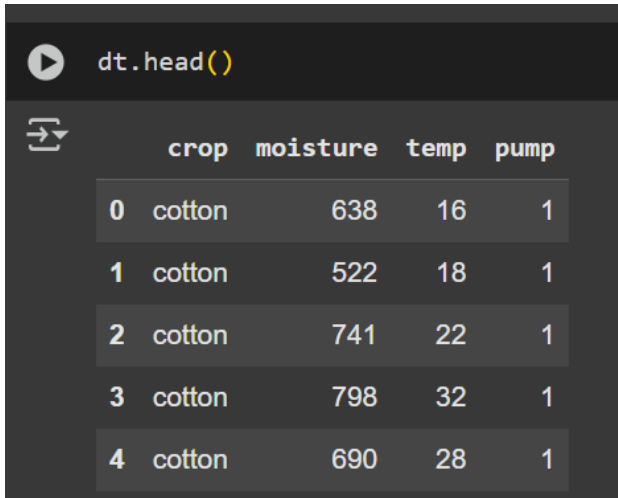
```
dt = pd.read_csv('/content/drive/MyDrive/Data Intelligent Irrigation System.csv')
dt
```

Buat variabel untuk import data dari drive ke colab

	crop	moisture	temp	pump
0	cotton	638	16	1
1	cotton	522	18	1
2	cotton	741	22	1
3	cotton	798	32	1
4	cotton	690	28	1
...
195	cotton	941	13	1
196	cotton	902	45	1
197	cotton	894	42	1
198	cotton	1022	45	1
199	cotton	979	10	1

200 rows × 4 columns

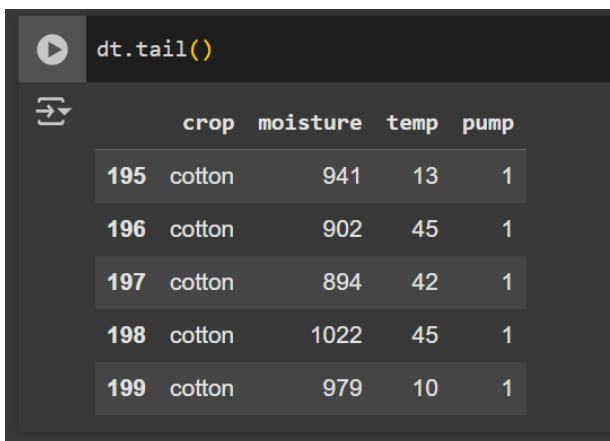
Hasil dari data import total ada 200 rows x 4 columns



```
dt.head()
```

	crop	moisture	temp	pump
0	cotton	638	16	1
1	cotton	522	18	1
2	cotton	741	22	1
3	cotton	798	32	1
4	cotton	690	28	1

kode head() untuk menampilkan data di awal atau bagian kepala data di ambil dari index 0 sampai 4 atau total baris ada 5 baris awal



```
dt.tail()
```

	crop	moisture	temp	pump
195	cotton	941	13	1
196	cotton	902	45	1
197	cotton	894	42	1
198	cotton	1022	45	1
199	cotton	979	10	1

kode tail untuk menampilkan data di akhir atau bagian ekor data di ambil dari index 195 sampai 199 atau akhir dari data

```
[ ] x = dt.iloc[:, [1,2]].values
    y = dt.iloc[:, 3].values
```

variabel x = memuat atribut variabel y = memuat label/target [] = mengambil semua baris [1,2] dan [-1] = mengambil kolom atau index tertentu

Kode ini menjelaskan x = memuat attribute y = memuat label/target [] = mengambil semua baris [1,2] dan [-1] = mengambil kolom atau index tertentu di setiap kolom nya

```
[ ] from sklearn.model_selection import train_test_split # Corrected import statement
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size = 0.4, random_state = 0)
```

```
[ ] print(x_train)
```

```
[[ 974  28]
 [ 824  37]
 [   4  42]
 [ 941  13]
 [ 482  27]
 [ 659  29]
 [ 741  22]
 [ 788  12]
 [   87  12]
 [ 894  42]
 [ 749  38]
 [ 671  23]
 [ 914  35]
 [ 611  12]
 [ 902  45]
 [1005  43]
 [1010  10]
 [ 354  20]
 [ 817  18]
 [ 306  25]
```

Kode di atas membagi dataset menjadi data latih (60%) dan data uji (40%) secara acak, tapi tetap konsisten karena ada parameter `random_state=0`. Machine learning ini untuk melatih menggunakan data latih dan mengujinya menggunakan data uji dapat mengevaluasi performa model dengan baik

```
print(x_test)
```

```
[[ 507  45]
 [ 981  36]
 [ 729  41]
 [ 293  25]
 [ 893  39]
 [ 876  40]
 [ 697  25]
 [ 673  35]
 [ 803  21]
 [   75  37]
 [ 687  11]
 [ 574  32]
 [ 883  40]
 [ 206  37]
 [ 748  27]
```

Pada variabel `x_test` adalah bagian dari dataset `x` yang diambil sebanyak 40% dari total data karena `test_size = 0.4` dan data yang ada di `x_test` dipilih secara acak tapi tetap konsisten dalam setiap eksekusi nya karena ada parameter `random_state=0`

```
[ ] from sklearn.preprocessing import StandardScaler
sc = StandardScaler()
x_train = sc.fit_transform(x_train)
x_test = sc.transform(x_test)
```

```
[ ] from sklearn.neighbors import KNeighborsClassifier
    knn = KNeighborsClassifier(n_neighbors=3) # Corrected argument name to n_neighbors
    knn.fit(x_train, y_train)

    y_predict = knn.predict(x_test)

    from sklearn.metrics import confusion_matrix
    cm = confusion_matrix(y_test, y_predict)
    print(cm)
```

```
[[21  2]
 [ 1 56]]
```

Kode ini berguna untuk standarisasi atau normalisasi pada isi value data, fitur dalam dataset menggunakan StandarScaler dari sklearn.preprocessing ini sangat penting dalam machine learning, dan hasil output dari normalisasi ini adalah matrix 2x2

```
# Cek Akurasi
from sklearn.metrics import classification_report
akurasi = classification_report(y_test, y_predict)
print(akurasi)

from sklearn.metrics import accuracy_score
akurasi = accuracy_score(y_test, y_predict)
print('Tingkat Akurasi Adalah: %d persen'%(akurasi*100))
```

	precision	recall	f1-score	support
0	0.95	0.91	0.93	23
1	0.97	0.98	0.97	57
accuracy			0.96	80
macro avg	0.96	0.95	0.95	80
weighted avg	0.96	0.96	0.96	80

Tingkat Akurasi Adalah: 96 persen

Kode ini menjelaskan soal hasil cek akurasi dari data set yang di uji, classification_report memberikan detail metrix evaluasi untuk setiap kelas dan accuracy_score menunjukkan persentase keseluruhan prediksi yang benar

```
from matplotlib.colors import ListedColormap
x_set, y_set = x_train, y_train
x1, x2 = np.meshgrid(start=[x_set[:,0].min()-1, stop=x_set[:,0]
```

Kode ini untuk menampilkan atau membuat mesh grid untuk visualisasi area Keputusan pada data 2D classification menggunakan matplotlib dan kode ini bertujuan untuk membuat grid nilai x1 dan x2 guna mempersiapkan visualisasi area keputusan yang di ambil dari analisis data set yang di uji

```

from matplotlib.colors import ListedColormap
x_set, y_set = x_train, y_train

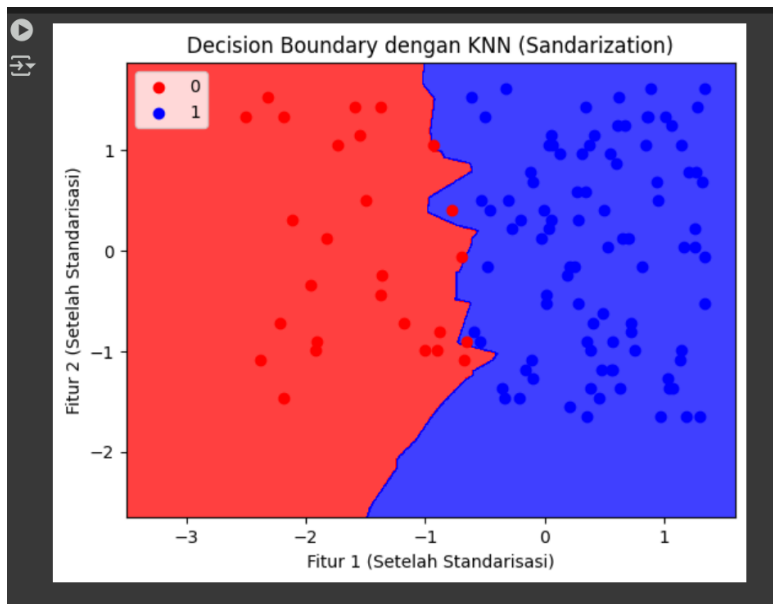
x1, x2 = np.meshgrid(
    np.arange(start=x_set[:,0].min() - 1, stop=x_set[:,0].max() + 1, step=0.01),
    np.arange(start=x_set[:,1].min() - 1, stop=x_set[:,1].max() + 1, step=0.01)
)

plt.contourf(
    x1, x2, knn.predict(np.array([x1.ravel(), x2.ravel()]).T).reshape(x1.shape),
    alpha=0.75, cmap=ListedColormap(('red', 'blue'))
)

for i, j in enumerate(np.unique(y_set)):
    plt.scatter(
        x_set[y_set == j, 0], x_set[y_set == j, 1],
        color=ListedColormap(('red', 'blue'))(i), label=j
    )

plt.xlim(x1.min(), x1.max())
plt.ylim(x2.min(), x2.max())
plt.xlabel('Fitur 1 (Setelah Standarisasi)')
plt.ylabel('Fitur 2 (Setelah Standarisasi)')
plt.legend()
plt.title('Decision Boundary dengan KNN (Sandarization)')
plt.show()

```



Kode ini memvisualisasikan decision boundary dari model KNN setelah standarisasi

1) Decision Boundary (Area Keputusan)

- Warna merah dan biru menunjukan area keputusan KNN dari data set yang di uji
- Model KNN akan mengklasifikasi titik data berdasarkan tetangga terdekatnya
- Garis pemisah di antara warna adalah batas keputusan model KNN

2) Titik Data

- Titik merah dan titik biru adalah data latih (x_{train}) yang telah distandarisasi
- Masing masing titik mewakili kelasnya sendiri sesuai dengan label y_{train}