# TUGAS PEMBELAJARAN MESIN

## “Data Iris”


**Pengampu :**

**Dr.Juni Nurma Sari S.Kom, M.MT**

**Nama:**

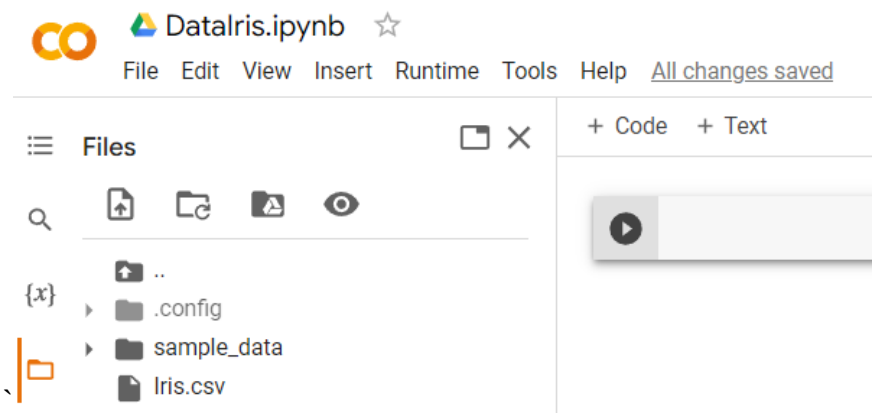
Wanda Trisnahayu (2055301143)

**Kelas : 3 TI C**

**D4 - TEKNIK INFORMATIKA**

**TAHUN 2022**

1. Import Dataset/Library




```
[5]  import numpy as np
     import pandas as pd
     import seaborn as sns
     sns.set_palette('husl')
     import matplotlib.pyplot as plt
     %matplotlib inline

     from sklearn import metrics
     from sklearn.neighbors import KNeighborsClassifier
     from sklearn.linear_model import LogisticRegression
     from sklearn.model_selection import train_test_split

     data = pd.read_csv('Iris.csv')
```

2. Data Preview

```
[6]  data.head()
```

| | 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 |

3. Info data

```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 150 entries, 0 to 149
Data columns (total 6 columns):
 #   Column         Non-Null Count  Dtype
---  ------         --------------  -----
 0   Id             150 non-null    int64
 1   SepalLengthCm  150 non-null    float64
 2   SepalWidthCm   150 non-null    float64
 3   PetalLengthCm  150 non-null    float64
 4   PetalWidthCm   150 non-null    float64
 5   Species        150 non-null    object
dtypes: float64(4), int64(1), object(1)
memory usage: 7.2+ KB
```

4. Deskripsi data

```
[8] data.describe()
```

| | Id | SepalLengthCm | SepalWidthCm | PetalLengthCm | PetalWidthCm |
|---|---|---|---|---|---|
| **count** | 150.000000 | 150.000000 | 150.000000 | 150.000000 | 150.000000 |
| **mean** | 75.500000 | 5.843333 | 3.054000 | 3.758667 | 1.198667 |
| **std** | 43.445368 | 0.828066 | 0.433594 | 1.764420 | 0.763161 |
| **min** | 1.000000 | 4.300000 | 2.000000 | 1.000000 | 0.100000 |
| **25%** | 38.250000 | 5.100000 | 2.800000 | 1.600000 | 0.300000 |
| **50%** | 75.500000 | 5.800000 | 3.000000 | 4.350000 | 1.300000 |
| **75%** | 112.750000 | 6.400000 | 3.300000 | 5.100000 | 1.800000 |
| **max** | 150.000000 | 7.900000 | 4.400000 | 6.900000 | 2.500000 |

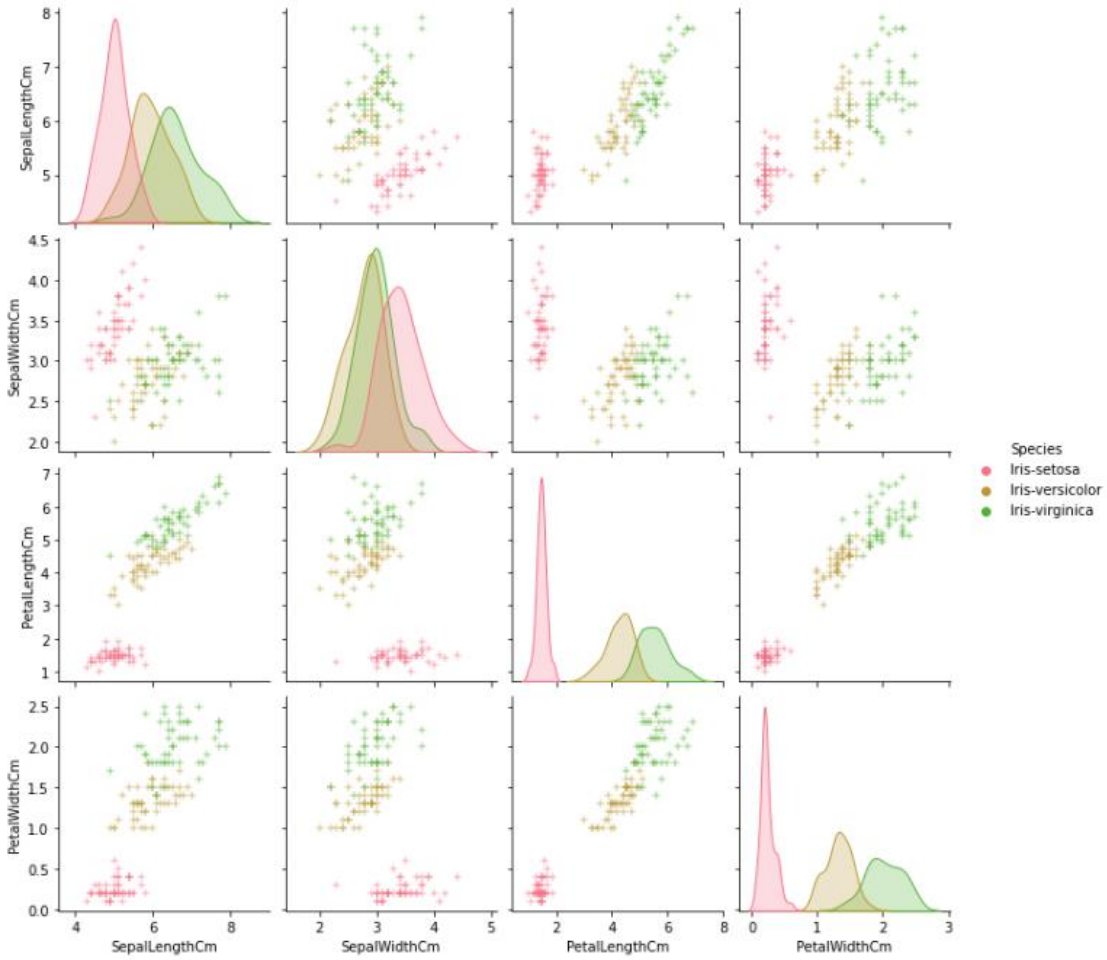
5. Menghitung jumlah data berdasar spesies

```
data['Species'].value_counts()
```

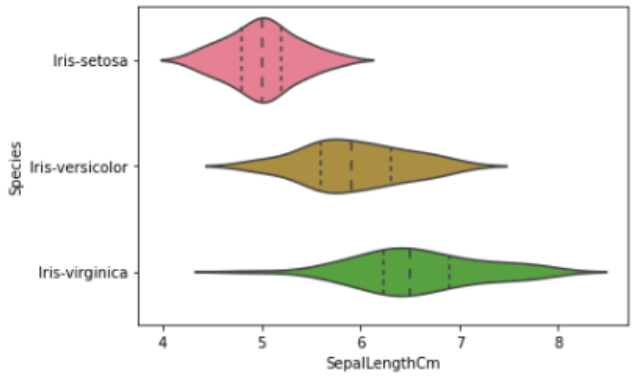
```
Iris-setosa        50
Iris-versicolor    50
Iris-virginica     50
Name: Species, dtype: int64
```

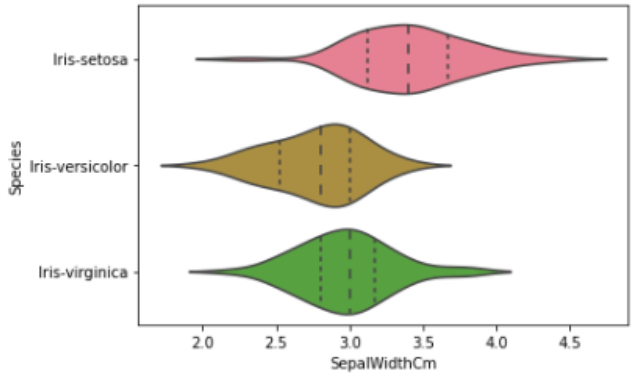
6. Visualisasi data

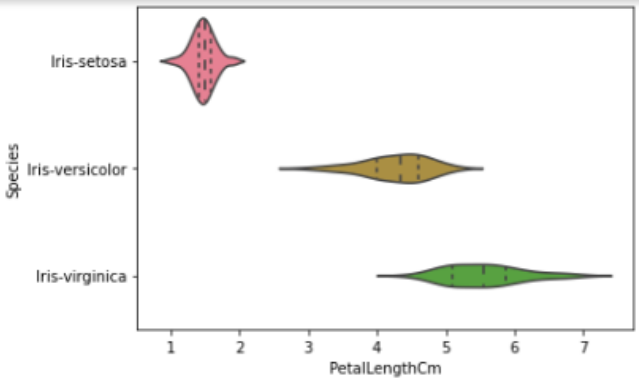
```
[10] tmp = data.drop('Id', axis=1)
     g = sns.pairplot(tmp, hue='Species', markers='+')
     plt.show()
```

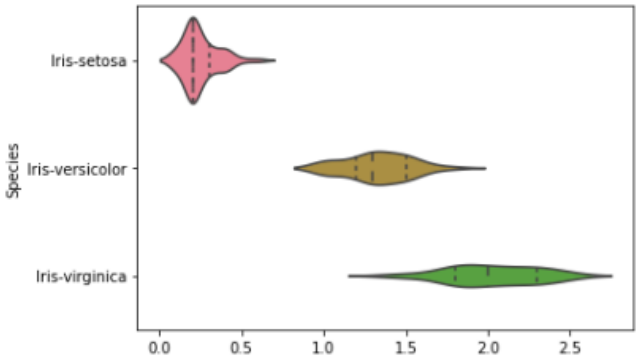


```
[11] g = sns.violinplot(y='Species', x='SepalLengthCm', data=data, inner='quartile')
     plt.show()
     g = sns.violinplot(y='Species', x='SepalWidthCm', data=data, inner='quartile')
     plt.show()
     g = sns.violinplot(y='Species', x='PetalLengthCm', data=data, inner='quartile')
     plt.show()
     g = sns.violinplot(y='Species', x='PetalWidthCm', data=data, inner='quartile')
     plt.show()
```



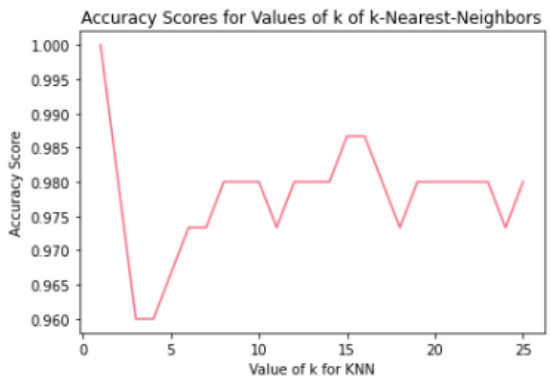
7. Modeling dengan scikit-learn

```
X = data.drop(['Id', 'Species'], axis=1)
y = data['Species']
# print(X.head())
print(X.shape)
# print(y.head())
print(y.shape)
```

```
(150, 4)
(150,)
```

8. Train and test dataset yang sama

```
# experimenting with different n values
k_range = list(range(1,26))
scores = []
for k in k_range:
    knn = KNeighborsClassifier(n_neighbors=k)
    knn.fit(X, y)
    y_pred = knn.predict(X)
    scores.append(metrics.accuracy_score(y, y_pred))

plt.plot(k_range, scores)
plt.xlabel('Value of k for KNN')
plt.ylabel('Accuracy Score')
plt.title('Accuracy Scores for Values of k of k-Nearest-Neighbors')
plt.show()
```




9. LogisticRegression

```
[14] logreg = LogisticRegression()
     logreg.fit(X, y)
     y_pred = logreg.predict(X)
     print(metrics.accuracy_score(y, y_pred))
```

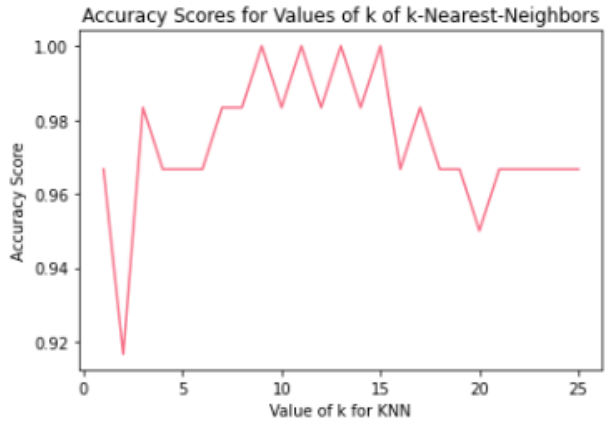
```
0.9733333333333334
```

10. Pisahkan dataset menjadi training set and a testing set

```
[15] X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.4, random_state=5)
     print(X_train.shape)
     print(y_train.shape)
     print(X_test.shape)
     print(y_test.shape)
```

```
(90, 4)
(90,)
(60, 4)
(60,)
```

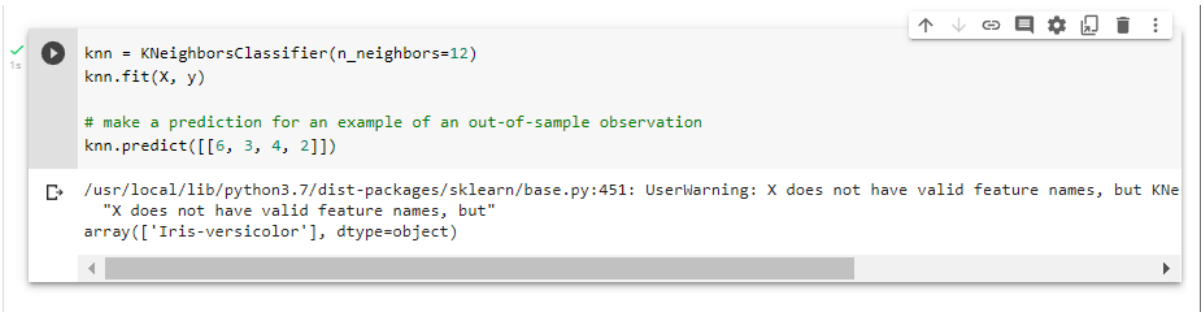
```
# experimenting with different n values
k_range = list(range(1,26))
scores = []
for k in k_range:
    knn = KNeighborsClassifier(n_neighbors=k)
    knn.fit(X_train, y_train)
    y_pred = knn.predict(X_test)
    scores.append(metrics.accuracy_score(y_test, y_pred))

plt.plot(k_range, scores)
plt.xlabel('Value of k for KNN')
plt.ylabel('Accuracy Score')
plt.title('Accuracy Scores for Values of k of k-Nearest-Neighbors')
plt.show()
```




```
logreg = LogisticRegression()
logreg.fit(X_train, y_train)
y_pred = logreg.predict(X_test)
print(metrics.accuracy_score(y_test, y_pred))
```

```
0.9833333333333333
```

11. Memilih KNN untuk Memodelkan Prediksi Spesies Iris dengan k = 12



```
knn = KNeighborsClassifier(n_neighbors=12)
knn.fit(X, y)

# make a prediction for an example of an out-of-sample observation
knn.predict([[6, 3, 4, 2]])
```

```
/usr/local/lib/python3.7/dist-packages/sklearn/base.py:451: UserWarning: X does not have valid feature names, but KNe
  "X does not have valid feature names, but"
array(['Iris-versicolor'], dtype=object)
```