LISTING BIG DATA ANALYTC 11

Nama : Marchelo Imanuel Salhuteru

Nim : 225410046

Kelas : informatika2

```
Generated code may be subject to a license | JuneNouh/task2AI
import pandas as pd
from sklearn.neighbors import KNeighborsClassifier
from sklearn import svm
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
from sklearn.model_selection import train_test_split
```

```
[17] data = pd.read_csv('iris.csv')
     print(data.head(5))
```

```
   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
```

Pembahasan :
Pertama kita masuk dengan mengimport pandas as pd dimana
pandas: Untuk memuat dan memanipulasi data dari file CSV.
KNeighborsClassifier: Untuk menggunakan model K-Nearest Neighbors.
svm: Untuk menggunakan model Support Vector Machine (SVM).
train_test_split: Untuk membagi data menjadi set pelatihan dan pengujian.
accuracy_score, confusion_matrix, classification_report: Untuk mengevaluasi performa model.

Selanjutnya kita masuk dengan menampilkan data dimana Kode ini memuat data dari file iris.csv ke dalam sebuah DataFrame pandas, lalu menampilkan 5 baris pertama dari DataFrame tersebut. Ini dilakukan untuk melihat format dan isi awal data yang akan digunakan.

```
[18] print(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

[19] #pandas".iloc" expects row_indexer, column_indexer
     x = data.iloc[:,:-1].values
     y= data['Species']
```

Pembahasan :
selanjutnya kita menampilkan data kode ini menampilkan 5 baris pertama dari DataFrame data. Ini mirip dengan perintah data.head(5) yang sudah dijalankan sebelumnya, hanya saja data.head() tanpa argumen akan menampilkan 5 baris secara default.

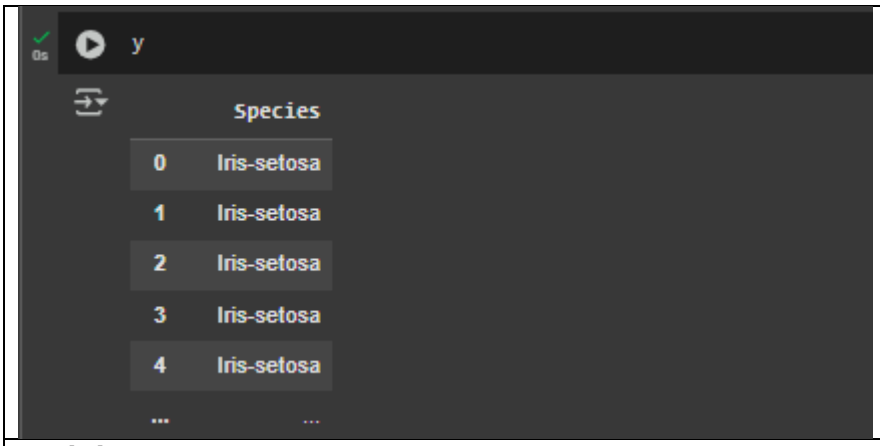
Kemudian Kode ini memisahkan data menjadi fitur (x) dan target (y).

x = data.iloc[:,:-1].values: Memilih semua baris (:) dan semua kolom kecuali kolom terakhir (:-1) dari DataFrame data sebagai fitur. .values mengubahnya menjadi array NumPy.
y = data['Species']: Memilih kolom 'Species' dari DataFrame data sebagai target atau label yang ingin diprediksi.

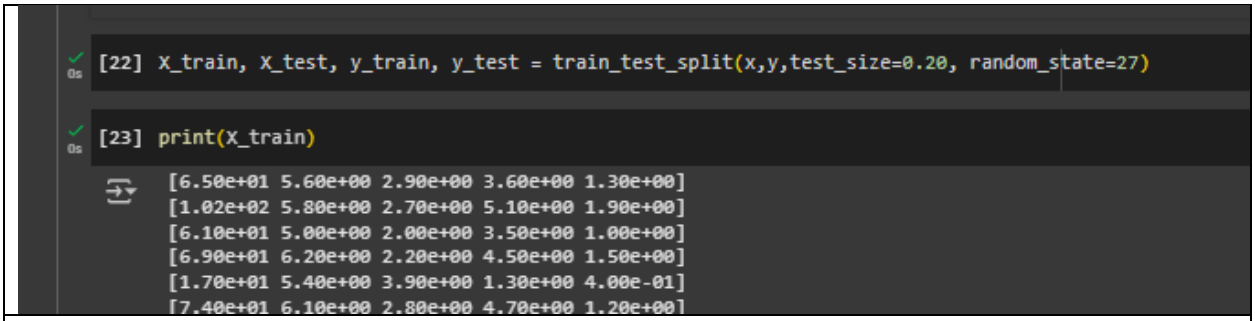
```
x

       [9.30e+01, 5.80e+00, 2.60e+00, 4.00e+00, 1.20e+00],
       [9.40e+01, 5.00e+00, 2.30e+00, 3.30e+00, 1.00e+00],
       [9.50e+01, 5.60e+00, 2.70e+00, 4.20e+00, 1.30e+00],
       [9.60e+01, 5.70e+00, 3.00e+00, 4.20e+00, 1.20e+00],
       [9.70e+01, 5.70e+00, 2.90e+00, 4.20e+00, 1.30e+00],
       [9.80e+01, 6.20e+00, 2.90e+00, 4.30e+00, 1.30e+00],
       [9.90e+01, 5.10e+00, 2.50e+00, 3.00e+00, 1.10e+00],
       [1.00e+02, 5.70e+00, 2.80e+00, 4.10e+00, 1.30e+00],
       [1.01e+02, 6.30e+00, 3.30e+00, 6.00e+00, 2.50e+00],
       [1.02e+02, 5.80e+00, 2.70e+00, 5.10e+00, 1.90e+00],
       [1.03e+02, 7.10e+00, 3.00e+00, 5.90e+00, 2.10e+00],
```

Pembahasan :
Selanjutnya kita menampilkan data dari data X apa isi data tersebut.



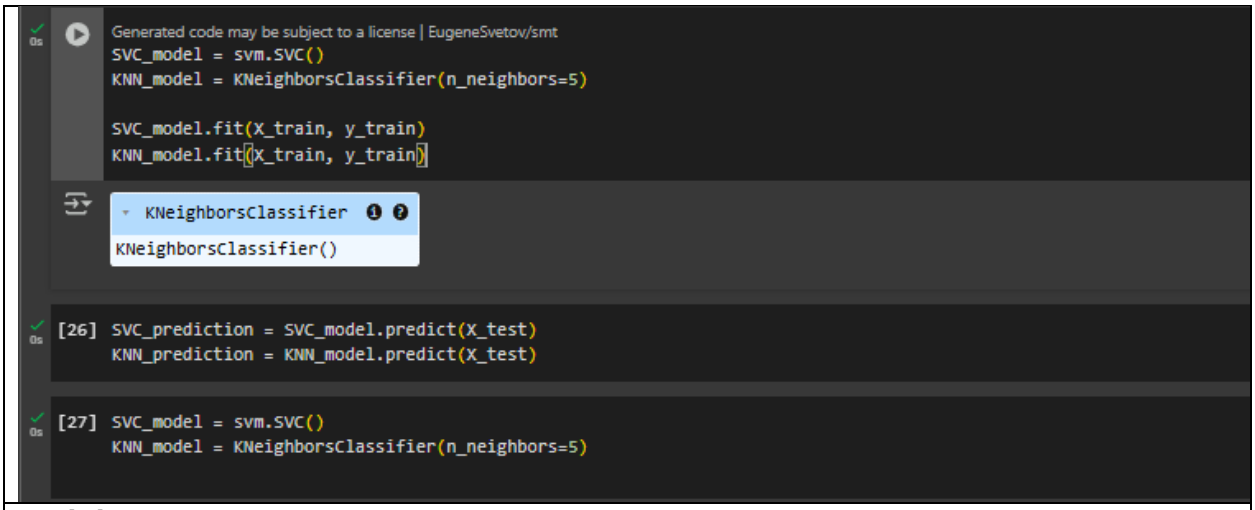
Pembahaasan :
Selanjutnya kita juga menampilkan isi dati data Y dimana data tersebut tertampil data dari data Species.



Pembahasan :
Selanjutnya kita masuk dengan menggunakan sebuah kode dimana ini membagi data fitur (x) dan target (y) menjadi empat bagian:

X_train: Fitur untuk melatih model.
X_test: Fitur untuk menguji model.
y_train: Target untuk melatih model.
y_test: Target untuk menguji model.
Argumen test_size=0.20 berarti 20% data akan digunakan untuk pengujian, dan 80% sisanya untuk pelatihan. random_state=27 memastikan bahwa pembagian data akan sama setiap kali kode dijalankan.

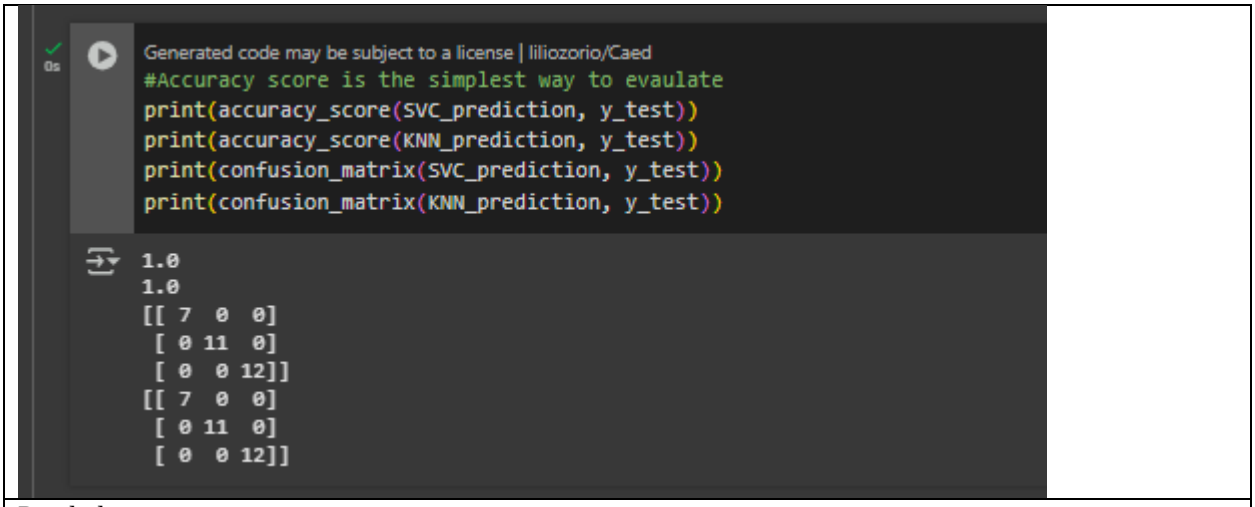
Kemudian menampilkan isi data dari data X_train
Print(X_train)



Pembahasan :
Kemudian kita masuk dengan menginisalisasi model data dimana Membuat objek untuk dua model klasifikasi: Support Vector Machine (SVM) dan K-Nearest Neighbors (dengan 5 tetangga terdekat). Pelatihan Model: Melatih kedua model tersebut menggunakan data pelatihan (X_train dan y_train) yang sebelumnya sudah dibagi. Proses pelatihan ini membuat model "belajar" dari data.

Selanjutnya dengan menggunakan model SVM (SVC_model) dan model K-Nearest Neighbors (KNN_model) yang sudah dilatih untuk membuat prediksi pada data pengujian (X_test). Hasil prediksinya disimpan dalam variabel SVC_prediction dan KNN_prediction.

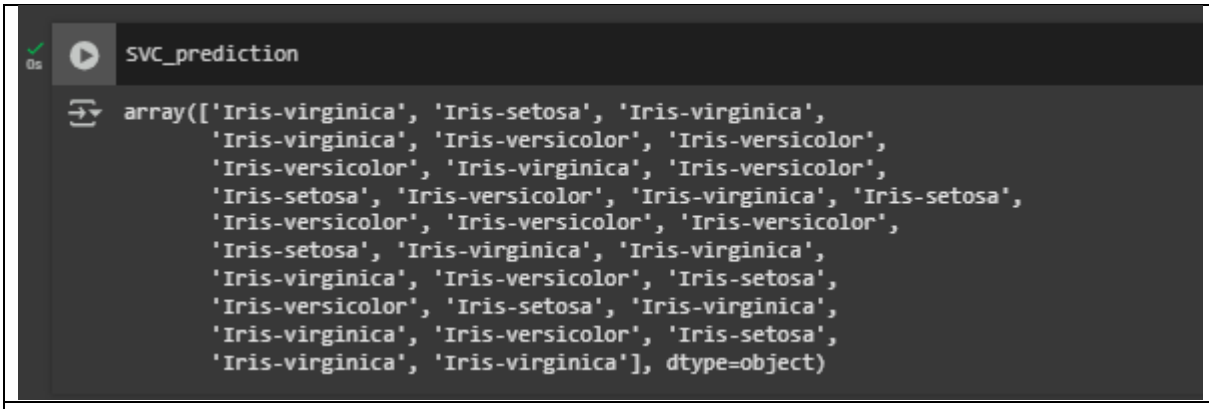
Kemudian dengan menginisialisasi ulang (membuat objek baru) untuk model Support Vector Machine (SVM) dan K-Nearest Neighbors (dengan 5 tetangga terdekat). Ini seperti memulai kembali model dari awal tanpa pelatihan sebelumnya.



Pembahasan :
Selanjutnya kira masuk dengan mengevaluasi seberapa baik kinerja kedua model (SVM dan KNN) yang sudah dilatih dan digunakan untuk prediksi:

print(accuracy_score(SVC_prediction, y_test)) dan print(accuracy_score(KNN_prediction, y_test)): Menghitung dan mencetak akurasi (persentase prediksi yang benar) untuk model SVM dan KNN

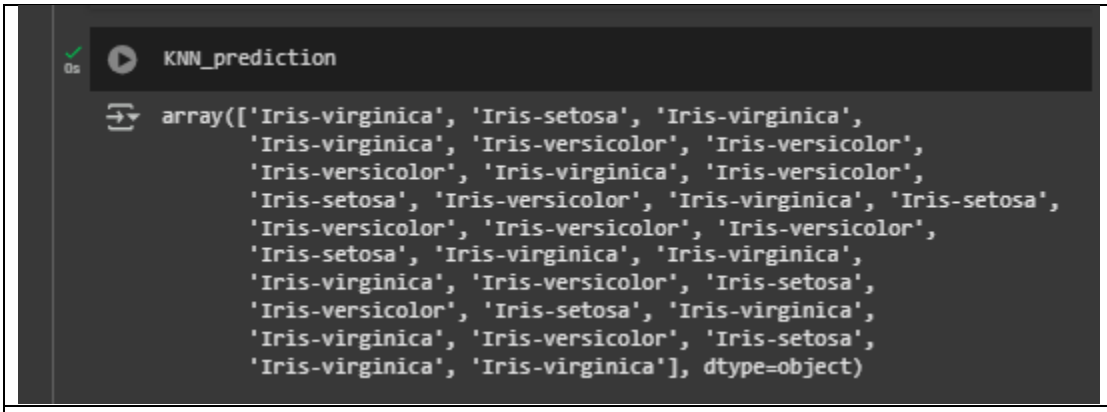
dengan membandingkan hasil prediksi (SVC_prediction dan KNN_prediction) dengan nilai target sebenarnya di data pengujian (y_test).
print(confusion_matrix(SVC_prediction, y_test)) dan print(confusion_matrix(KNN_prediction, y_test)): Menghitung dan mencetak confusion matrix (matriks kebingungan) untuk kedua model. Confusion matrix memberikan rincian lebih lanjut tentang hasil prediksi, menunjukkan jumlah true positives, true negatives, false positives, dan false negatives untuk setiap kelas.
Output yang dihasilkan menunjukkan bahwa kedua model memiliki akurasi 1.0 (atau 100%) pada data pengujian, dan confusion matrix menunjukkan bahwa semua prediksi untuk setiap kelas adalah benar.



```
SVC_prediction

array(['Iris-virginica', 'Iris-setosa', 'Iris-virginica',
       'Iris-virginica', 'Iris-versicolor', 'Iris-versicolor',
       'Iris-versicolor', 'Iris-virginica', 'Iris-versicolor',
       'Iris-setosa', 'Iris-versicolor', 'Iris-virginica', 'Iris-setosa',
       'Iris-versicolor', 'Iris-versicolor', 'Iris-versicolor',
       'Iris-setosa', 'Iris-virginica', 'Iris-virginica',
       'Iris-virginica', 'Iris-versicolor', 'Iris-setosa',
       'Iris-versicolor', 'Iris-setosa', 'Iris-virginica',
       'Iris-virginica', 'Iris-versicolor', 'Iris-setosa',
       'Iris-virginica', 'Iris-virginica'], dtype=object)
```

Pembahasan :
Setelah sudah kita masuk dengan menampilkan isi data dari data SVC_prediction dimana data tersebut berisi array dari berbagai data iris.



```
KNN_prediction

array(['Iris-virginica', 'Iris-setosa', 'Iris-virginica',
       'Iris-virginica', 'Iris-versicolor', 'Iris-versicolor',
       'Iris-versicolor', 'Iris-virginica', 'Iris-versicolor',
       'Iris-setosa', 'Iris-versicolor', 'Iris-virginica', 'Iris-setosa',
       'Iris-versicolor', 'Iris-versicolor', 'Iris-versicolor',
       'Iris-setosa', 'Iris-virginica', 'Iris-virginica',
       'Iris-virginica', 'Iris-versicolor', 'Iris-setosa',
       'Iris-versicolor', 'Iris-setosa', 'Iris-virginica',
       'Iris-virginica', 'Iris-versicolor', 'Iris-setosa',
       'Iris-virginica', 'Iris-virginica'], dtype=object)
```

Pembahasan :
Sama dengan sebelumnya juga menampilkan data dari KNN_prediction.