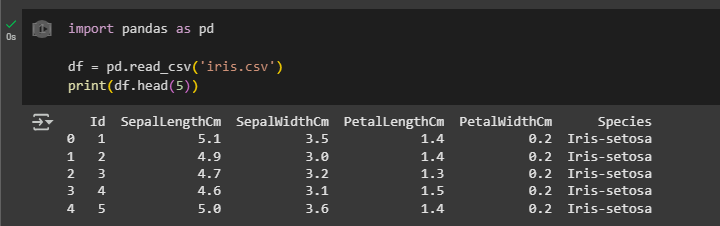
Nama : Marchelo Imanuel Salhuteru

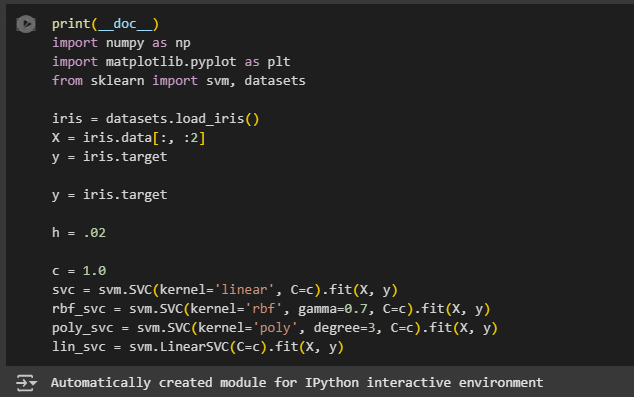
Nim : 225410046

Kelas : Informatika2



Pembahasan :

Pada praktik 1 kita mengimport library dimana kita akan mengimport data file dari iris.csv,dengan menggunakan pandas as pd setelah di import kemudian kita menampilkan 5 data head maka tampilan akan muncul seperti pada gambar di atas.



Pembahasan :

Pada praktik ke 2 kita melakukan beberapa hal dimana kita mengklasifikasi menggunakan support vector machine (SVM) pada data yang sudah di import sebelumnya, dimana kita mengimport library yang di butuhkan seperti numpy untuk operasi numerik, matplotlib.pyplot untuk plotting, dan svm serta datasets dari sklearn untuk model SVM dan dataset Iris. Selanjutnya kita memuat data iris dengan menggunakan datasets.load\_iris(). Dataset ini berisi pengukuran sepal dan petal dari tiga spesies bunga iris, selanjutnya Memilih Fitur: Memilih dua fitur pertama dari dataset Iris (X = iris.data[:, :2]), yaitu Sepal Length dan Sepal Width. Variabel y berisi target (spesies bunga iris), kemudian masuk pada Inisialisasi Parameter: Mengatur nilai h (ukuran langkah untuk meshgrid) dan c (parameter regularisasi untuk SVM), selanjutnya dengan Melatih empat model SVM yang berbeda dengan kernel yang berbeda:

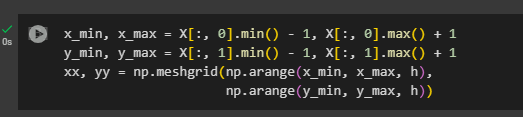
svc: SVM dengan kernel linear

rbf\_svc: SVM dengan kernel Radial Basis Function (RBF)

poly\_svc: SVM dengan kernel polinomial derajat 3

lin\_svc: LinearSVC, implementasi SVM linear yang lebih efisien untuk dataset besar.

Setiap model dilatih menggunakan data X (fitur) dan y (target) Secara keseluruhan, kode ini menyiapkan data dan melatih beberapa model SVM untuk kemudian dapat digunakan untuk visualisasi batas keputusan klasifikasi yang dibuat oleh masing-masing model.



Pembahasan :

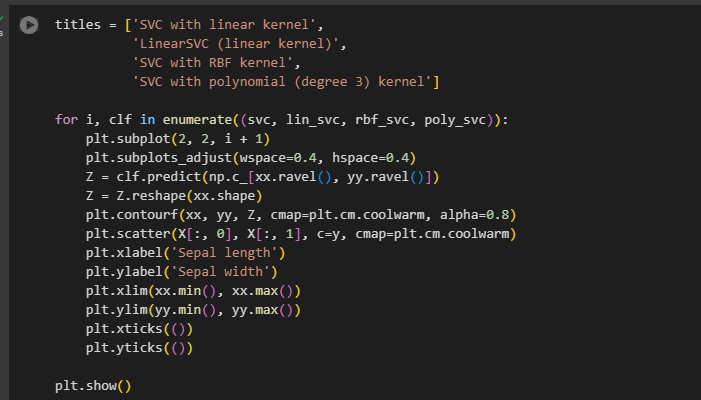
Selanjutnya pada praktik ke 3 dimana kita membuat sebuah grid titik-titik yang akan kita gunakan untuk memvisualisasikan batas keputusan dari model svm dengan menentukan :

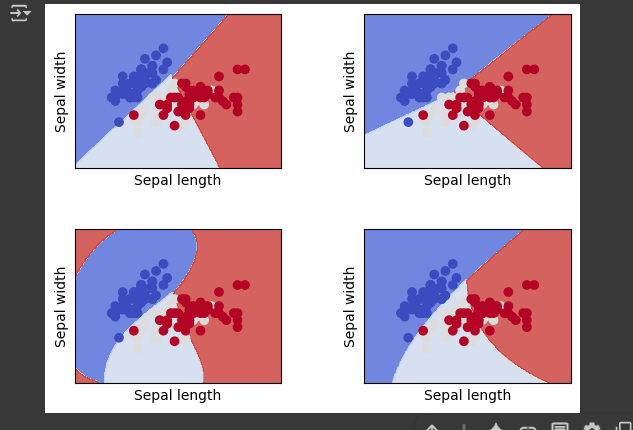
Batas plot:

x\_min, x\_max = X[:, 0].min() - 1, X[:, 0].max() + 1: Menentukan nilai minimum dan maksimum untuk sumbu x (fitur pertama, yaitu Sepal Length atau Petal Width, tergantung pada sel sebelumnya). Penambahan dan pengurangan 1 dilakukan untuk memberikan sedikit ruang di sekitar data.

y\_min, y\_max = X[:, 1].min() - 1, X[:, 1].max() + 1: Menentukan nilai minimum dan maksimum untuk sumbu y (fitur kedua, yaitu Sepal Width atau Petal Length). Penambahan dan pengurangan 1 juga dilakukan untuk memberikan ruang.

Membuat Meshgrid:

xx, yy = np.meshgrid(np.arange(x\_min, x\_max, h), np.arange(y\_min, y\_max, h)): Membuat grid persegi panjang dari titik-titik. np.arange(x\_min, x\_max, h) membuat array nilai dari x\_min hingga x\_max dengan langkah h. np.meshgrid kemudian mengambil dua array 1D ini dan mengembalikannya sebagai grid 2D xx dan yy. Setiap titik (xx[i, j], yy[i, j]) dalam grid ini akan digunakan untuk memprediksi kelasnya menggunakan model SVM, yang hasilnya kemudian akan digunakan untuk menggambar batas keputusan.



Pembahasan :

Selanjutnya pada praktik akhir dimana kita memvisualisasikan batas keputusan dari empat model svm yang berbeda dalam satu plot dengan empat subplot dengan tahap berikut :

Daftar Judul Plot: titles adalah daftar string yang berisi judul untuk setiap subplot, yang menjelaskan jenis kernel SVM yang digunakan.

Iterasi Melalui Model SVM: Loop for mengiterasi melalui daftar model SVM yang sebelumnya dilatih (svc, lin\_svc, rbf\_svc, poly\_svc). Variabel i adalah indeks (dari 0 hingga 3) dan clf adalah objek model SVM saat ini.

Membuat Subplot:

plt.subplot(2, 2, i + 1): Membuat subplot dalam grid 2x2. i + 1 menentukan posisi subplot saat ini (1 hingga 4).

plt.subplots\_adjust(wspace=0.4, hspace=0.4): Menyesuaikan jarak horizontal (wspace) dan vertikal (hspace) antara subplot agar lebih mudah dibaca.

Memprediksi Kelas untuk Setiap Titik dalam Grid:

Z = clf.predict(np.c\_[xx.ravel(), yy.ravel()]): Menggunakan model SVM saat ini (clf) untuk memprediksi kelas untuk setiap titik dalam meshgrid xx dan yy. np.c\_[xx.ravel(), yy.ravel()] menggabungkan array xx dan yy yang diratakan menjadi array 2D di mana setiap baris adalah pasangan koordinat (x, y) dari titik dalam grid.

Z = Z.reshape(xx.shape): Mengubah bentuk hasil prediksi Z agar sesuai dengan bentuk meshgrid xx, sehingga dapat digunakan untuk membuat kontur.

Menggambar Batas Keputusan dan Titik Data:

plt.contourf(xx, yy, Z, cmap=plt.cm.coolwarm, alpha=0.8): Menggambar area kontur yang diisi, mewakili wilayah keputusan yang diprediksi oleh model. xx dan yy adalah koordinat grid, Z adalah kelas yang diprediksi untuk setiap titik, cmap=plt.cm.coolwarm menentukan skema warna, dan alpha=0.8 mengatur transparansi.

plt.scatter(X[:, 0], X[:, 1], c=y, cmap=plt.cm.coolwarm): Menggambar titik-titik data pelatihan asli di atas kontur batas keputusan. X[:, 0] adalah koordinat x dari data, X[:, 1] adalah koordinat y, c=y mewarnai titik-titik berdasarkan kelas sebenarnya, dan cmap=plt.cm.coolwarm menggunakan skema warna yang sama.

Menambahkan Label dan Menyesuaikan Tampilan Plot:

plt.xlabel('Sepal length'): Menambahkan label untuk sumbu x (sebelumnya diubah menjadi 'Petal width').

plt.ylabel('Sepal width'): Menambahkan label untuk sumbu y (sebelumnya diubah menjadi 'Petal length').

plt.xlim(xx.min(), xx.max()), plt.ylim(yy.min(), yy.max()): Mengatur batas sumbu x dan y agar sesuai dengan rentang meshgrid.

plt.xticks(()), plt.yticks(()): Menghilangkan tanda centang pada sumbu x dan y untuk tampilan yang lebih bersih pada plot kontur.

Menampilkan Plot: plt.show() menampilkan plot yang berisi keempat subplot.

Latihan 1

print(\_\_doc\_\_)

import numpy as np

import matplotlib.pyplot as plt

from sklearn import svm, datasets

iris = datasets.load\_iris()

X = iris.data[:, 2:4] # Menggunakan PetalLength dan PetalWidth

y = iris.target

h = .02

c = 1.0

svc = svm.SVC(kernel='linear', C=c).fit(X, y)

rbf\_svc = svm.SVC(kernel='rbf', gamma=0.7, C=c).fit(X, y)

poly\_svc = svm.SVC(kernel='poly', degree=3, C=c).fit(X, y)

lin\_svc = svm.LinearSVC(C=c).fit(X, y)

titles = ['SVC with linear kernel',

          'LinearSVC (linear kernel)',

          'SVC with RBF kernel',

          'SVC with polynomial (degree 3) kernel']

for i, clf in enumerate((svc, lin\_svc, rbf\_svc, poly\_svc)):

    plt.subplot(2, 2, i + 1)

    plt.subplots\_adjust(wspace=0.4, hspace=0.4)

    Z = clf.predict(np.c\_[xx.ravel(), yy.ravel()])

    Z = Z.reshape(xx.shape)

    plt.contourf(xx, yy, Z, cmap=plt.cm.coolwarm, alpha=0.8)

    plt.scatter(X[:, 0], X[:, 1], c=y, cmap=plt.cm.coolwarm)

    plt.xlabel('Petal width') # Mengubah label sumbu x

    plt.ylabel('Petal length') # Mengubah label sumbu y

    plt.xlim(xx.min(), xx.max())

    plt.ylim(yy.min(), yy.max())

    plt.xticks(())

    plt.yticks(())

    plt.title(titles[i]) # Menambahkan judul pada setiap plot

plt.show()

