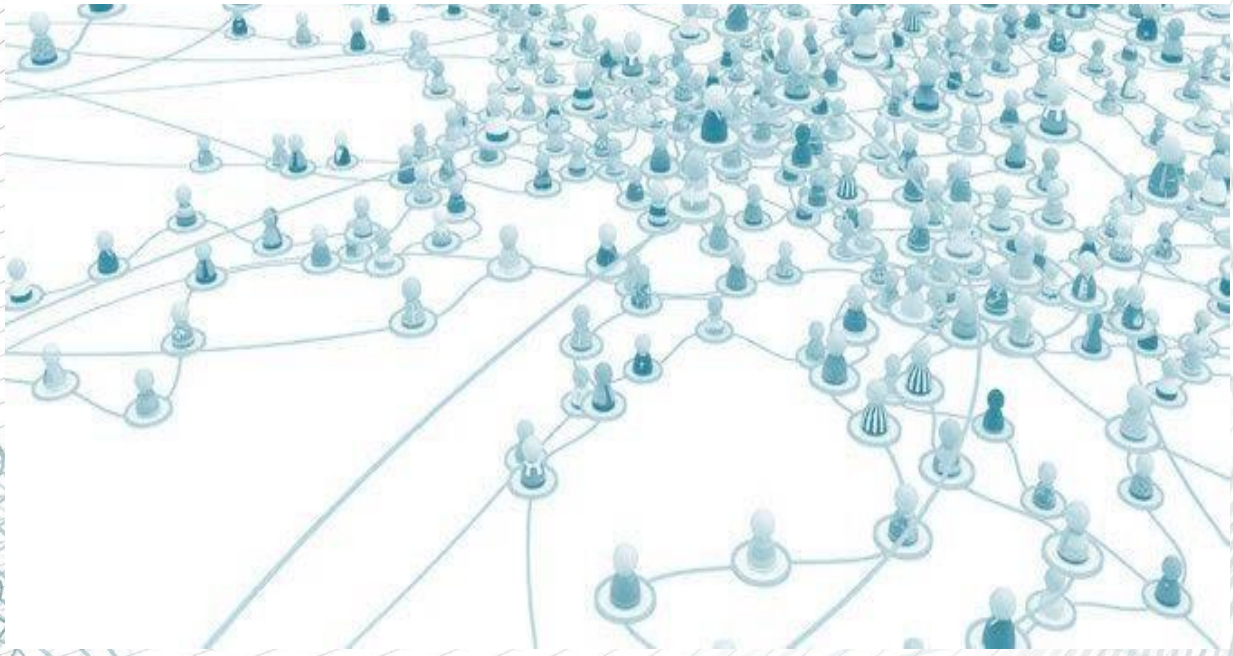
MODUL PRAKTIKUM

SD3107-Pembelajaran Mesin



Program Studi Sains Data
Fakultas Sains
Institut Teknologi Sumatera

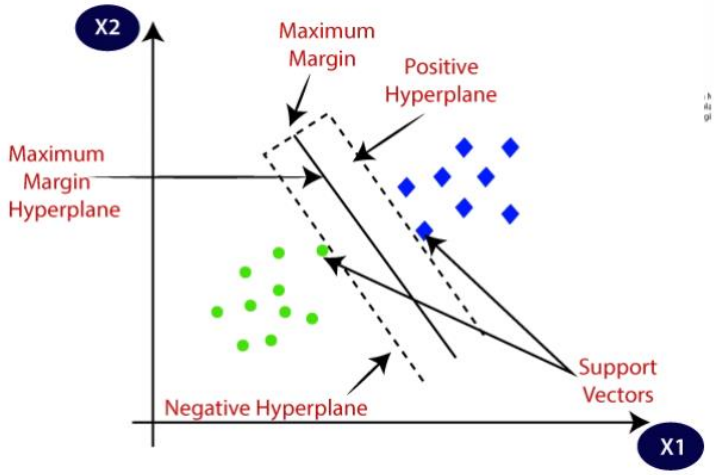
2024

MODUL 3

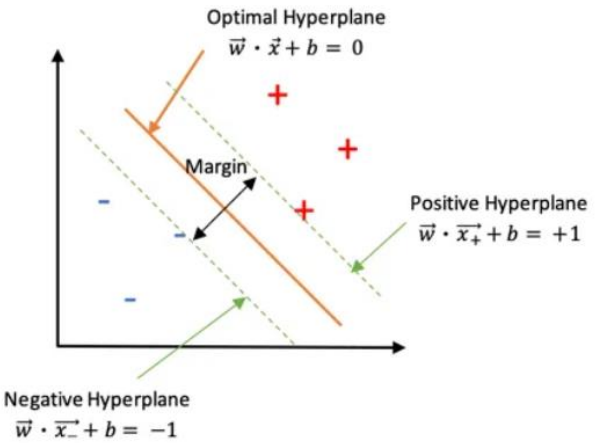
Support Vector Machine for Classification

A. Konsep Dasar

Support Vector Machine (SVM) merupakan salah satu metode *supervised learning* yang biasanya digunakan untuk klasifikasi (seperti *Support Vector Classification*) dan regresi (*Support Vector Regression*). Pada klasifikasi, SVM memisahkan kelas pada ruang fitur menggunakan pemisah yang disebut dengan *hyperplane.* Tujuan dari SVM sendiri adalah mencari *hyperplane* terbaik untuk memisahkan kelas pada dataset. Berikut ini adalah gambar dari sebuah SVM yang memisahkan dataset menajdi dua kelas.



Pada gambar di atas, sebuah hyperplane memisahkan dua data berwarna hijau dan biru. Untuk membuat sebuah klasifikasi yang optimal, dibutuhkan sebuah *hyperplane* yang memiliki margin maksimal yang artinya *hyperplane* memiliki jarak maksimum dengan titik terdekat data yang disebut *Support Vector*. Hyperplane memiliki sepasang garis sejajar yang disebut *hyperplane negatif* dan *hyperplane positif.* Berikut ini adalah gambar yang menunukkan Support Vector, margin, dan *hyperplane* dari sebuah klasifikasi SVM.



Pada klasifikasi SVM, data yang diolah dinotasikan sebagai $\{x_1,\ldots,x_n\}$ sedangkan label keluaran dinotasikan sebagai $\gamma_i \in \{-1, +1\}$ untuk n = 1,2,….,n di mana n adalah banyaknya

data. *Hyperplane* pada SVM sebagai bidang pembatas kedua data akan membagi data menggunakan persamaan berikut.

$$w^T.x_i + b \geq +1, untuk\ y_i = +1$$
$$w^T.x_i + b \geq -1, untuk\ y_i = -1$$

Persamaan di atas merupakan persamaan bidang dengan w = *vector* bobot yang tegak lurus terhadap *hyperplane* (bidang normal), dan b = posisi bidang *relative* terhadap pusat koordinat.

B. Tujuan Praktikum

I. Tujuan Instruksional Umum

Praktikum bertujuan untuk menerapkan teori *Support Vector Machine* (SVM) untuk klasifikasi dalam pembelajaran mesin.

II. Tujuan Instruksional Khusus

1. Mahasiswa mampu menguasai konsep dasar *Support Vector Machine.*
2. Mahasiswa mampu menyelesaikan studi kasus klasifikasi pada data linear sederhana menggunakan SVM
3. Mahasiswa mampu menganalisis hasil dari studi kasus menggunakan metode *Support Vector Machine.*

C. Dataset dan bahasa pemrograman Python

C.1 Dataset

Dataset yang digunakan pada praktikum ini dapat dilihat pada tabel berikut.

- Data Latih

Fitur 1 (Panjang Kalimat)	Fitur 2 (Jumlah Tautan)	Spam
10	1	0
25	3	1
15	0	0
30	5	1
12	1	0
20	4	1
8	0	0
22	2	1
14	1	0
18	3	1

- Data Uji

Panjang kalimat	Jumlah Tautan	Spam
11	0	0
18	3	1
16	0	0
24	5	1
7	0	0

Pada tabel di atas, kolom ketiga dan keempat merupakan kolom dengan data biner Ya dan Tidak yang sudah diubah ke dalam nilai 1 dan 0.

C.2 Bahasa Pemrograman Python

Pada praktikum modul 3 ini, kita akan menggunakan bahasa pemrograman python dan beberapa library atau pustaka untuk memudahkan implementasi program. Pustaka yang akan digunakan diantaranya numpy, pandas, sklearn, dan matplotlib.

a. NumPy: digunakan untuk operasi matematika berbasis array dan matriks. NumPy sangat penting dalam komputasi numerik dengan Python pada praktikum ini.
b. Pandas : library untuk pengolahan data berbasis tabel (DataFrame). Pandas juga dapat melakukan pengolahan komputasi numerik seperti numpy, contohnya seperti rata-rata dan standar deviasi.
c. Scikit-learn (sklearn): library untuk implementasi berbagai algoritma machine learning seperti klasifikasi, regresi, clustering, dan reduksi dimensi. Pada praktikum ini, kita akan menggunakan pustaka sklearn.svm.
d. Matplotlib dan Seaborn: digunakan untuk proses visualisasi data. Matplotlib dapat digunakan untuk membuat visualisasi data dalam bentuk grafik, bagan, peta, dan lainnya. Pustaka lain yang dapat digunakan untuk visualisasi data adalah Seaborn.

D. Implementasi Support Vector Machine untuk Klasifikasi

Pada praktikum ini kita akan menggunakan Google colab yang dapat diakses menggunakan tautan https://colab.research.google.com/.

1) Import pustaka atau library yang akan digunakan

```
# import library
import numpy as np
from sklearn.svm import SVC
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score
from sklearn.metrics import confusion_matrix
import matplotlib.pyplot as plt
import seaborn as sns
```

2) Masukkan dataset yang akan digunakan.

```
# masukkan  dataset
X = np.array([[10,1],[25,3],[15,0],
              [30,5],[12,1],[20,4],
              [8,0],[22,2],[14,1],[18,3]])
y = np.array([0,1,0,1,0,1,0,1,0,1])
```

3) Import fungsi SVM dari library sklearn.

```
# tentukan algoritma yang akan digunakan
# dengan menggunakan SVC, masukkan algoritma ke dalam variabel model
# kernel diatur langsung ke linear karena jika tidak, sklearn secara otomatis mengatur SVC menjadi rbf
model = SVC(kernel='linear')
```

4) Latih model menggunakan dataset yang sudah disiapkan.

```
# latih model menggunakan data yang sudah disiapkan
# gunakan data X sebagai fitur masukan dan data y sebagai keluaran
model.fit(X,y)
```

```
SVC
SVC(kernel='linear')
```

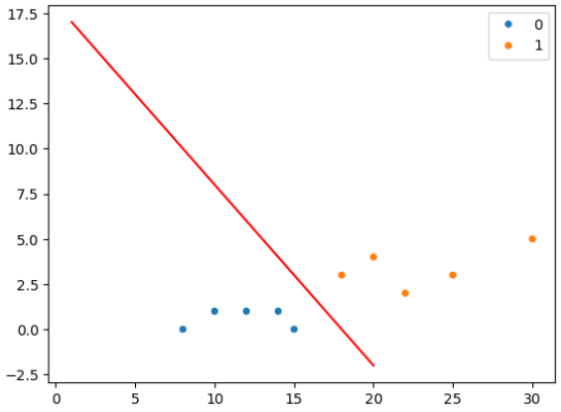
5) Lihat Support Vectors pada model.

```
# support vector
print(f'Support vectors:\n {model.support_vectors_}\n')

print(f'Support vektor pada masing-masing kelas:\n {model.n_support_}')
```

```
Support vectors:
 [[15.  0.]
 [14.  1.]
 [18.  3.]]

Support vektor pada masing-masing kelas:
 [2 1]
```

6) Menampilkan data dan *hyperplane* SVM

```
sns.scatterplot(x=X[:,0], y=X[:,1], hue=y, s=30)
w = model.coef_[0]
b = model.intercept_[0]
# buat skala pada sumbu x dari 0-10
x_points = np.linspace(1, 20)
# buat nilai pada sumbu y menggunakan nilai x
y_points = -(w[0] / w[1]) * x_points - b / w[1]
# plot hyperplane menggunakan warna merah
plt.plot(x_points, y_points, c='r')
plt.show()
```



7) Lakukan pengujian pada data baru.

Setelah berhasil membuat model, akan dilakukan prediksi menggunakan data baru

untuk melihat akurasi dari model SVM yang telah dibuat menggunakan data uji sebagai berikut.

```
# masukkan data uji ke dalam dua variabel berbeda
X_uji = np.array([[11,0],[18,3],[16,0],[24,5],[7,0]])
y_uji = np.array([0,1,0,1,0])
```

```
prediksi = model.predict(X_uji)
print(f'Hasil prediksi SVM: {prediksi}')
```

```
Hasil prediksi SVM: [0 1 0 1 0]
```

8) Hitung akurasi model menggunakan

```
# hitung akurasi
accuracy = accuracy_score(y_uji, prediksi)
print(f'Akurasi: {accuracy}')
```

```
Akurasi: 1.0
```

Data Non Linear

Pada kasus nyata, dataset yang digunakan tidak selalu linear. SVM dapat melakukan klasifikasi terhadap data non-linier dimana merupakan kumpulan data yang tidak dapat dipisahkan secara linier. Berikut adalah klasifikasi pada data non linear menggunakan SVM.

a) Import library

```
# import library
import numpy as np
import pandas as pd
from sklearn.svm import SVC
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score
from sklearn.metrics import confusion_matrix
import matplotlib.pyplot as plt
import seaborn as sns
```

b) Import data yang akan digunakan

Dataset dapat diunduh pada https://shorturl.at/K9lFX

```
data = pd.read_csv('okupansi.csv')
data.head()
```

	Temperature	Humidity	Light	CO2	HumidityRatio	Occupancy
0	23.18	27.2720	426.0	721.25	0.004793	1
1	23.15	27.2675	429.5	714.00	0.004783	1
2	23.15	27.2450	426.0	713.50	0.004779	1
3	23.15	27.2000	426.0	708.25	0.004772	1
4	23.10	27.2000	426.0	704.50	0.004757	1

c) Tentukan fitur dan target

```
X = data.drop('Occupancy', axis=1) #### Fitur
y = data['Occupancy']              #### Target
```

d) Bagi data latih dan data uji

```
from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.25, random_state=50)
```

e) Import fungsi SVM

```
model2 = SVC()
model2 = model2.fit(X_train, y_train)
model2
```

```
SVC
SVC()
```

Pada fungsi SVC di atas, kernel secara otomatis akan diisi menggunakan kernel rbf atau **Radial Basic Function** yang dapat digunakan untuk klasifikasi data non-linear. Kernel RBF atau juga disebut kernel Gaussian adalah konsep kernel yang paling banyak digunakan untuk memecahkan masalah klasifikasi data yang tidak dapat dipisahkan secara linear. Kernel ini dikenal memiliki performa yang baik dengan parameter tertentu, dan hasil dari pelatihan memiliki nilai error yang kecil dibandingkan dengan kernel lainnya.

f) Lakukan prediksi terhadap data uji yang telah dibagi

```
model2_pred = model2.predict(X_test)
model2_pred
```

g) Hitung akurasi model

```
from sklearn.metrics import accuracy_score
print("Accuracy:", accuracy_score(y_test, model2_pred))
```

```
Accuracy: 0.9891944990176817
```