Classification With Linear SVM Python

A. TUJUAN

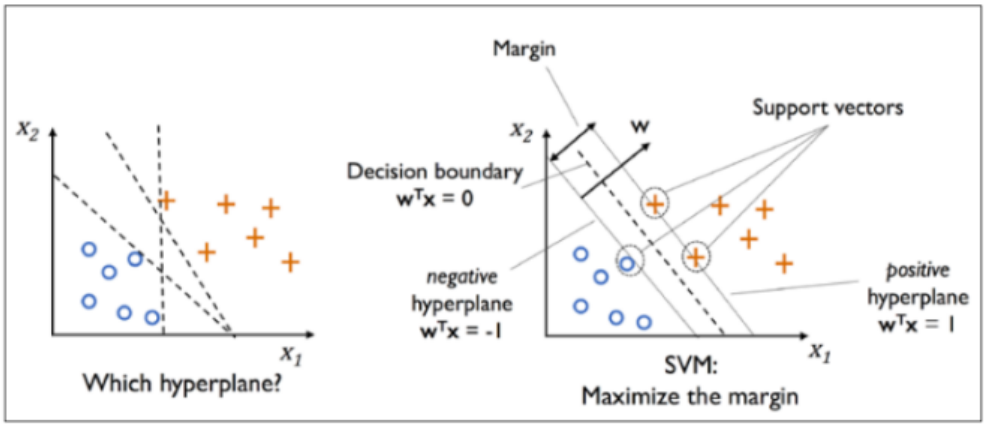
a. Mahasiswa mengetahui penggunaan Linear SVM pada Pengenalan Pola
b. Mahasiswa mengetahui tahapan dalam metode Linear SVM.
c. Mahasiswa mampu melakukan pemrograman dasar Linear SVM

B. ALAT DAN BAHAN

a. Laptop/PC
b. Google Colab

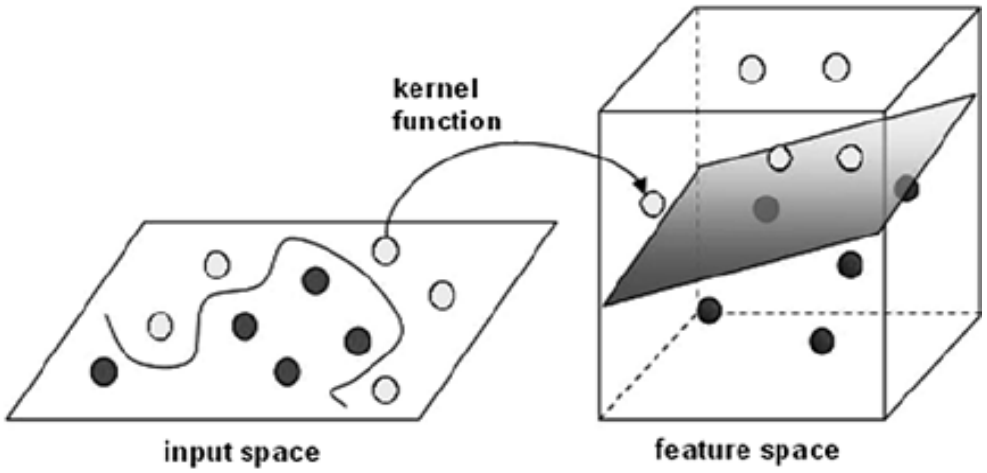
C. TEORI DASAR

Support Vector Machine (SVM) merupakan salah satu metode dalam *supervised learning* yang biasanya digunakan untuk klasifikasi (seperti *Support Vector Classification*) dan regresi (*Support Vector Regression*). Dalam pemodelan klasifikasi, SVM memiliki konsep yang lebih matang dan lebih jelas secara matematis dibandingkan dengan teknik-teknik klasifikasi lainnya. SVM juga dapat mengatasi masalah klasifikasi dan regresi dengan *linear* maupun *non linear*.



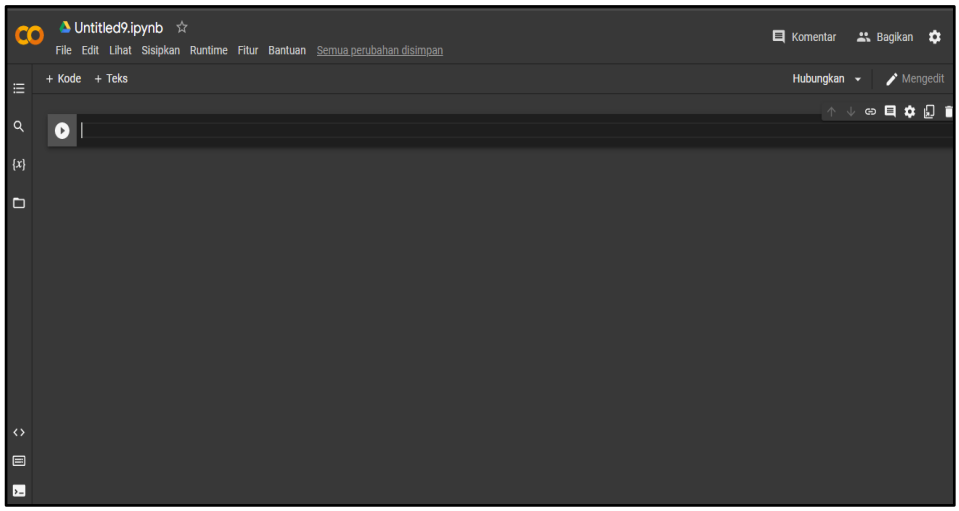
Gambar 1 Hyperplane yang memisahkan dua kelas positif (+1) dan negatif(-1)

SVM digunakan untuk mencari *hyperplane* terbaik dengan memaksimalkan jarak antar kelas. H*yperplane* adalah sebuah fungsi yang dapat digunakan untuk pemisah antar kelas. Dalam 2-D fungsi yang digunakan untuk klasifikasi antar kelas disebut sebagai *line whereas*, fungsi yang digunakan untuk klasifikasi antas kelas dalam 3-D disebut *plane similarly*, sedangan fungsi yang digunakan untuk klasifikasi di dalam ruang kelas dimensi yang lebih tinggi di sebut *hyperplane*. *Hyperplane* yang ditemukan SVM diilustrasikan seperti Gambar 1 posisinya berada ditengah-tengah antara dua kelas, artinya jarak antara hyperplane dengan objek-objek data berbeda dengan kelas yang berdekatan (terluar) yang diberi tanda bulat kosong dan positif. Dalam SVM objek data terluar yang paling dekat dengan *hyperplane* disebut *support vector*. Objek yang disebut *support vector* paling sulit diklasifikasikan dikarenakan posisi yang hampir tumpang tindih (*overlap*) dengan kelas lain. Mengingat sifatnya yang kritis, hanya *support vector* inilah yang diperhitungkan untuk menemukan *hyperplane* yang paling optimal oleh SVM.

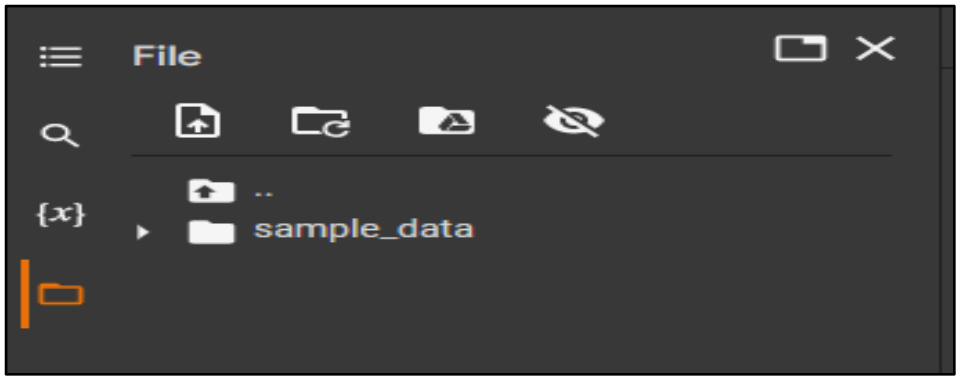


D. LANGKAH KERJA

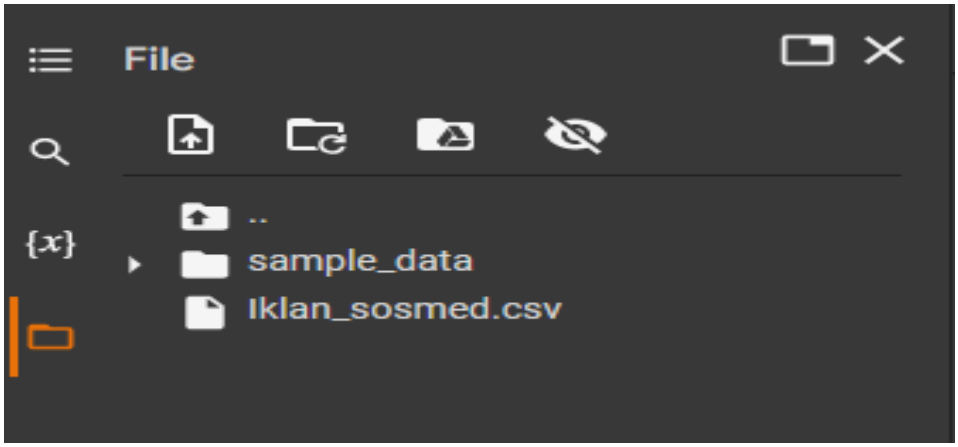
1. Bukalah Google Colab



2. Pilihlah icon file pada sebelah kiri kemudian klik upload



3. Setelah terupload dataset akan muncul pada bagian jendela kiri



4. Masukan kode program dibawah ini

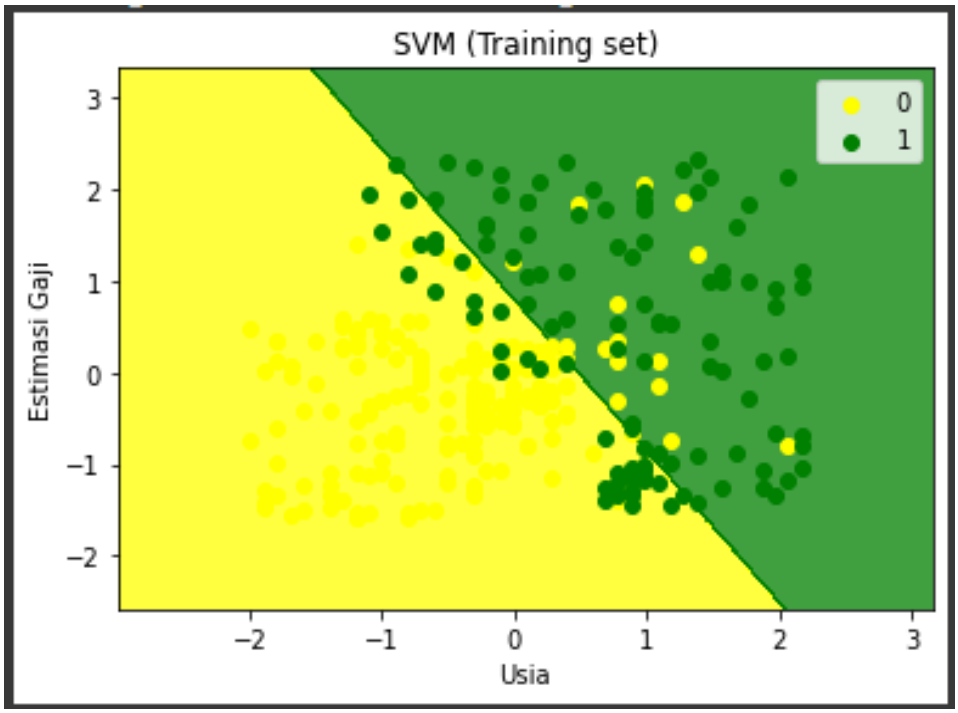
```python
# Mengimpor library
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd

# Mengimpor dataset
dataset = pd.read_csv('Iklan_sosmed.csv')
X = dataset.iloc[:, [2, 3]].values
y = dataset.iloc[:, 4].values

# Menjadi dataset ke dalam Training set and Test set
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 = 0)

# Feature Scaling
from sklearn.preprocessing import StandardScaler
sc = StandardScaler()
X_train = sc.fit_transform(X_train)
X_test = sc.transform(X_test)

# Membuat model SVM terhadap Training set
from sklearn.svm import SVC
classifier = SVC(kernel = 'linear', random_state = 0)
classifier.fit(X_train, y_train)

# Memprediksi hasil test set
y_pred = classifier.predict(X_test)
```

```python
# Membuat confusion matrix
from sklearn.metrics import confusion_matrix
cm = confusion_matrix(y_test, y_pred)

# Visualisasi hasil model SVM dari Training set
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, classifier.predict(np.array([X1.ravel(), X2.ravel()]).T).reshape(X1.shape),
             alpha = 0.75, cmap = ListedColormap(('red', 'green')))
plt.xlim(X1.min(), X1.max())
plt.ylim(X2.min(), X2.max())
for i, j in enumerate(np.unique(y_set)):
    plt.scatter(X_set[y_set == j, 0], X_set[y_set == j, 1],
                c = ListedColormap(('red', 'green'))(i), label = j)
plt.title('SVM (Training set)')
plt.xlabel('Usia')
plt.ylabel('Estimasi Gaji')
plt.legend()
plt.show()

# Visualisasi model SVM terhadap Test set
from matplotlib.colors import ListedColormap
X_set, y_set = X_test, y_test
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, classifier.predict(np.array([X1.ravel(), X2.ravel()]).T).reshape(X1.shape),
             alpha = 0.75, cmap = ListedColormap(('red', 'green')))
```

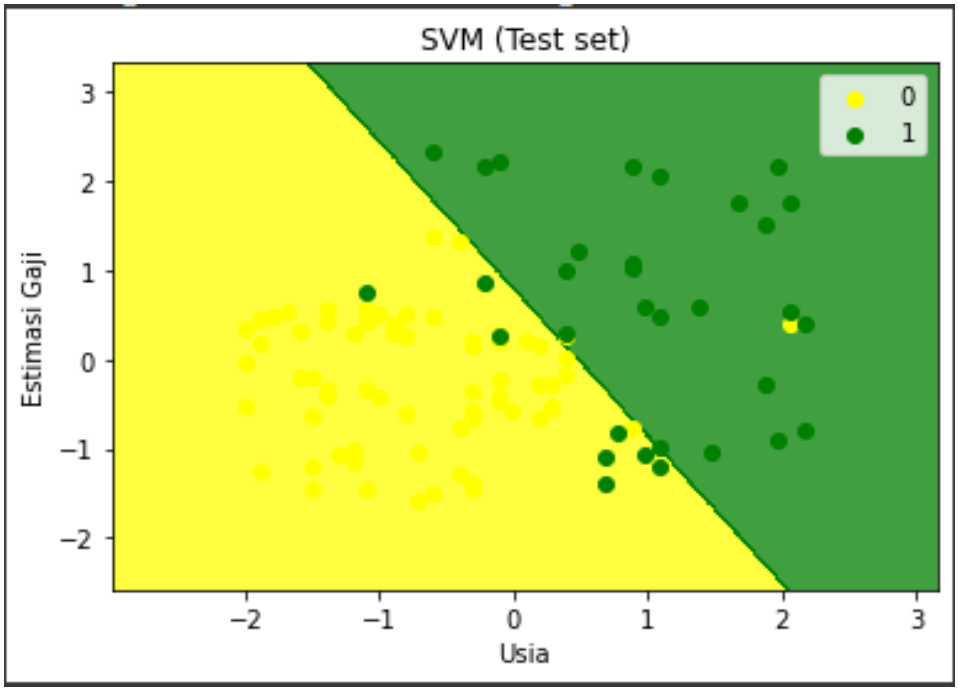
```python
plt.xlim(X1.min(), X1.max())
plt.ylim(X2.min(), X2.max())
for i, j in enumerate(np.unique(y_set)):
    plt.scatter(X_set[y_set == j, 0], X_set[y_set == j, 1],
                c = ListedColormap(('red', 'green'))(i), label = j)
plt.title('SVM (Test set)')
plt.xlabel('Usia')
plt.ylabel('Estimasi Gaji')
plt.legend()
plt.show()
```

E. HASIL PERCOBAAN

1. Tampilan hasil clustering pada data training



2. Tampilan clustering dari data test



F. ANALISIS

Pada percobaan ini dilakukan prediksi data Gaji iklan di sosmed, pada line 22terdapat fungsi mengimpor SVM dari sklearn. Line 23 mendefinisikan objek classifier sebagai model SVM dengan parameter kernel=linear dan random number generator = 0. Penentuan random number generator ini bebas ya. Tips: Arahkan kursor pada SVC kemudian ketik di keyboard CTRL+i untuk bisa melihat parameter apa saja yang diperlukan untuk membuat model SVM. Perhatikan bahwa kernel yang kita pilih adalah linear, artinya SVM akan membuat hyperplane yang bersifat linear untuk 2 dimensi. Kernel=linear bisa juga diartikan tanpa kernel, karena itu adalah algoritma SVM biasa, tanpa penambahan kernel. Di halaman selanjutnya kita akan menggunakan kernel yang berbeda. Line 24 membuat model SVM nya terhadap Training set. Line 27 mendefinisikan objek y_pred sebagai hasil prediksi dari model yang dibuat di line 24 ke Test set. Line 30-31 membuat Confusion matrix (untuk bisa menilai akurasi modelnya nanti). Line 34-49 adalah perintah untuk memvisualisasikan hasil model SVM terhadap Training set. Line 52-67 adalah visualisasi model SVM kita terhadap Test set. Pada hasil percobaan diatas, bisa dilihat bahwa ada 10 dari 100 titik yang salah dideteksi oleh SVM (akurasi 90%). Artinya bahwa model SVM yang menggunakan kernel linear sudah cukup baik untuk membantu pemilik *showroom* mobil menentukan wilayah yang tepat untuk iklan mobil SUV nya di internet.