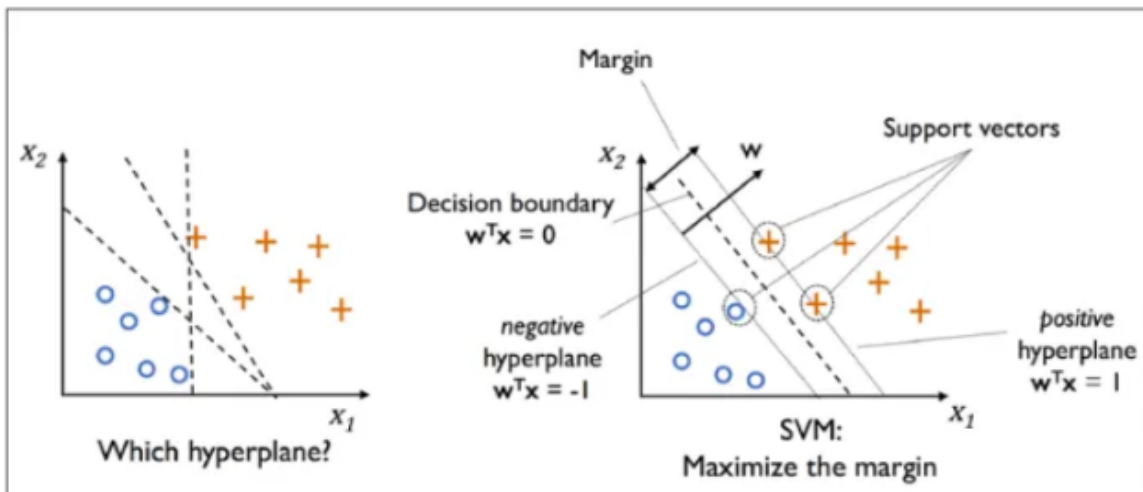


# SVM (Support Vector Machine)

Adinda Putri - 13523071

**SVM** merupakan algoritma yang digunakan untuk classification dan regression dengan linear maupun non linear. Prinsipnya, SVM digunakan untuk mencari **hyperplane** terbaik dengan memaksimalkan jarak antarkelas. Hyperplane adalah fungsi untuk memisahkan antarkelas. Pada ruang 2-D fungsi tersebut disebut **line** **whereas**, sedangkan pada ruang 3-D fungsi tersebut disebut **plane** **similarly**. Sementara itu, pada ruang dimensi tinggi, fungsi tersebut disebut hyperplane.



Gambar 1. Ilustrasi Beberapa Konsep Penting dalam SVM

Sumber: <https://medium.com/@samsudiney/penjelasan-sederhana-tentang-apa-itu-svm-149fec72bd02>

Beberapa konsep penting dalam SVM adalah sebagai berikut.

- **Hyperplane**: Persamaan/fungsi  $wx + b$  (pada linear classification) yang memisahkan kelas.
- **Support Vectors**: Data point terdekat dengan hyperplane.
- **Margin**: Jarak antara hyperplane dan support vectors.
- **Soft Margin**: Memungkinkan beberapa misclassification dan menyeimbangkan margin maximization dan misclassification penalty jika data tidak terpisah sempurna.
- **Kernel**: Fungsi yang memetakan data ke dimensi lebih tinggi agar SVM bisa meng-handle data nonlinear

- **Hinge Loss:** Loss function
- **C:** Regularization term. Semakin tinggi nilai C semakin ketat penalty untuk misclassification
- **Dual Problem:** Melibatkan Lagrange multipliers dari support vectors untuk memfasilitasi kernel trick dan komputasi efisien
- **Optimizer:** Dapat berupa gradient descent atau quadratic programming

## Cara Kerja

Berikut cara kerja untuk SVM dengan **QP (Quadratic Programming)**:

### 1. Problem Primal SVM (Soft Margin)

Untuk **SVM dengan soft margin**, primal problem-nya adalah

$$\begin{aligned} \min_{w,b,\xi} \quad & \frac{1}{2} \|w\|^2 + C \sum_{i=1}^n \xi_i \\ \text{s.t.} \quad & y_i (w^T \phi(x_i) + b) \geq 1 - \xi_i, \quad \xi_i \geq 0 \end{aligned}$$

keterangan: s.t. : subject to,  $\phi(x_i)$  : transformasi fitur (identitas untuk linear, mapping ke dimensi tinggi untuk kernel,  $\xi_i$  : slack variable, C: regularisasi (trade-off margin vs misclassification).

### 2. Dual Problem (Quadratic Programming)

Primal problem sulit diselesaikan untuk kernel non-linear. Solusinya adalah mengubah primal problem menjadi **dual problem**:

$$\begin{aligned} \max_{\alpha} \quad & \sum_{i=1}^n \alpha_i - \frac{1}{2} \sum_{i,j=1}^n \alpha_i \alpha_j y_i y_j K(x_i, x_j) \\ \text{s.t.} \quad & 0 \leq \alpha_i \leq C, \quad \sum_{i=1}^n \alpha_i y_i = 0 \end{aligned}$$

dengan  $\alpha_i$ : Lagrange multipliers,  $K(x_i, x_j) = \phi(x_i)^T \phi(x_j)$  = kernel function. Beberapa jenis kernel. Beberapa jenis kernel adalah sebagai berikut.

- Linear kernel:

$$K(x_i, x) = x_i^\top x$$

- RBF kernel:

$$K(x_i, x) = \exp(-\gamma \|x_i - x\|^2)$$

### 3. Setup Quadratic Programming (CVXOPT)

QP standar:

$$\min \frac{1}{2} \alpha^T P \alpha + q^T \alpha \quad \text{s.t.} \quad G \alpha \leq h, \quad A \alpha = b$$

Mapping dari dual SVM didapatkan:

- $P = (y_i y_j K(x_i, x_j))_{i,j}$
- $q = -1$  (vektor  $-1$ )
- $G = \begin{bmatrix} -I \\ I \end{bmatrix}, h = \begin{bmatrix} 0 \\ C \end{bmatrix}$
- $A = y^T, b = 0$

Solvers CVXOPT akan menghasilkan  $\alpha$ .

### 4. Support Vectors dan Bias

- Support vectors (SV): titik dengan  $\alpha_i > 0$  atau

$$SV = \{x_i \mid \alpha_i > 1e-5\}$$

- Bias  $b$ :

$$b = \frac{1}{|SV|} \sum_{i \in SV} \left( y_i - \sum_{j \in SV} \alpha_j y_j K(x_j, x_i) \right)$$

## 5. Decision Function:

- Linear kernel:

$$f(x) = w^T x + b, \quad w = \sum_{i \in SV} \alpha_i y_i x_i$$

- Kernel non-linear:

$$f(x) = \sum_{i \in SV} \alpha_i y_i K(x_i, x) + b$$

- Prediksi kelas:

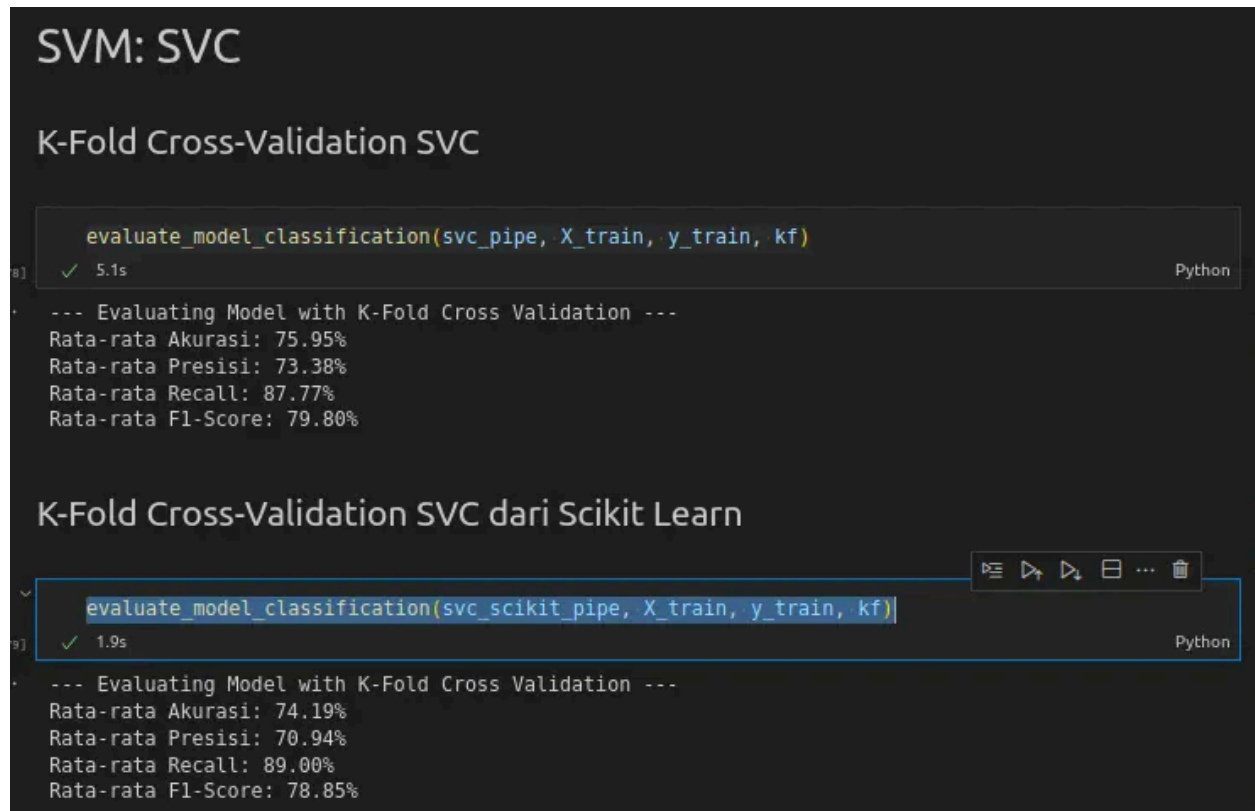
$$\hat{y} = \text{sign}(f(x))$$

Untuk semua kernel, cara kerjanya sama yaitu:

1. Formulasikan dual SVM
2. Masukkan kernel  $K(x_i, x_j)$
3. Solve QP untuk mendapatkan  $\alpha_i$

4. Tentukan support vectors dan bias
5. Gunakan decision function untuk prediksi

### Perbandingan model dari scratch dengan dari Scikit-Learn



The image shows a Jupyter Notebook with two sections. The first section, titled 'SVM: SVC', shows a code cell for 'K-Fold Cross-Validation SVC' that takes 5.1s to execute. The second section, titled 'K-Fold Cross-Validation SVC dari Scikit Learn', shows a similar code cell that takes 1.9s to execute. Both sections display the same evaluation metrics: Accuracy (75.95% vs 74.19%), Precision (73.38% vs 70.94%), Recall (87.77% vs 89.00%), and F1-Score (79.80% vs 78.85%).

```
evaluate_model_classification(svc_pipe, X_train, y_train, kf)
```

✓ 5.1s Python

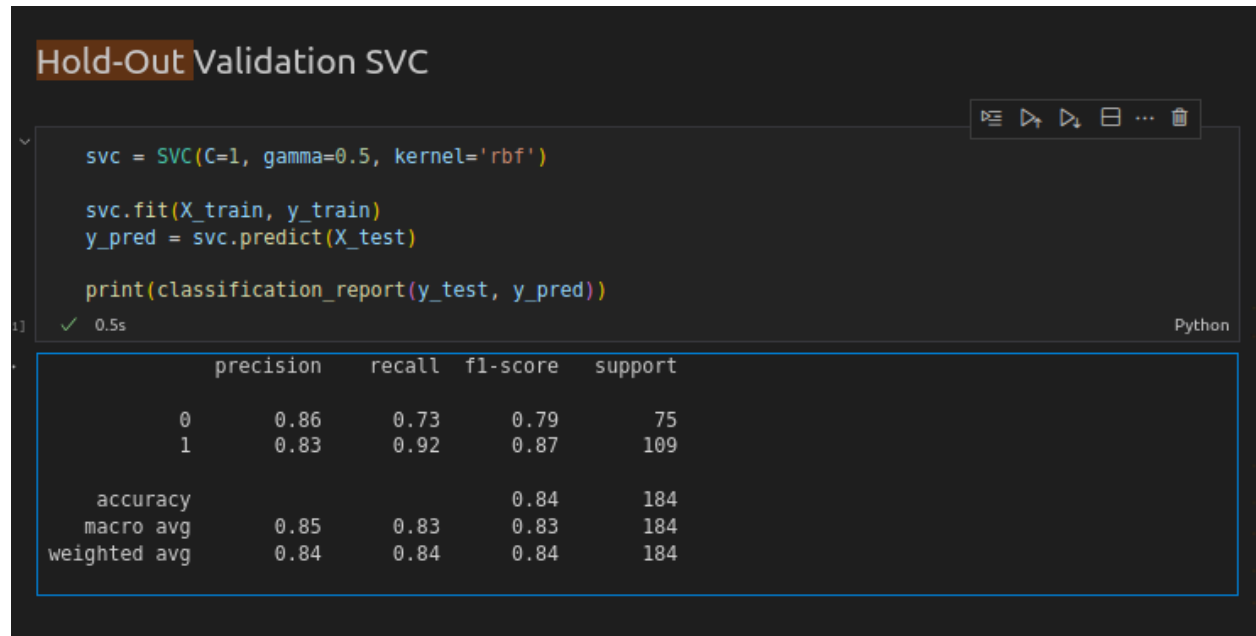
--- Evaluating Model with K-Fold Cross Validation ---  
Rata-rata Akurasi: 75.95%  
Rata-rata Presisi: 73.38%  
Rata-rata Recall: 87.77%  
Rata-rata F1-Score: 79.80%

```
evaluate_model_classification(svc_scikit_pipe, X_train, y_train, kf)
```

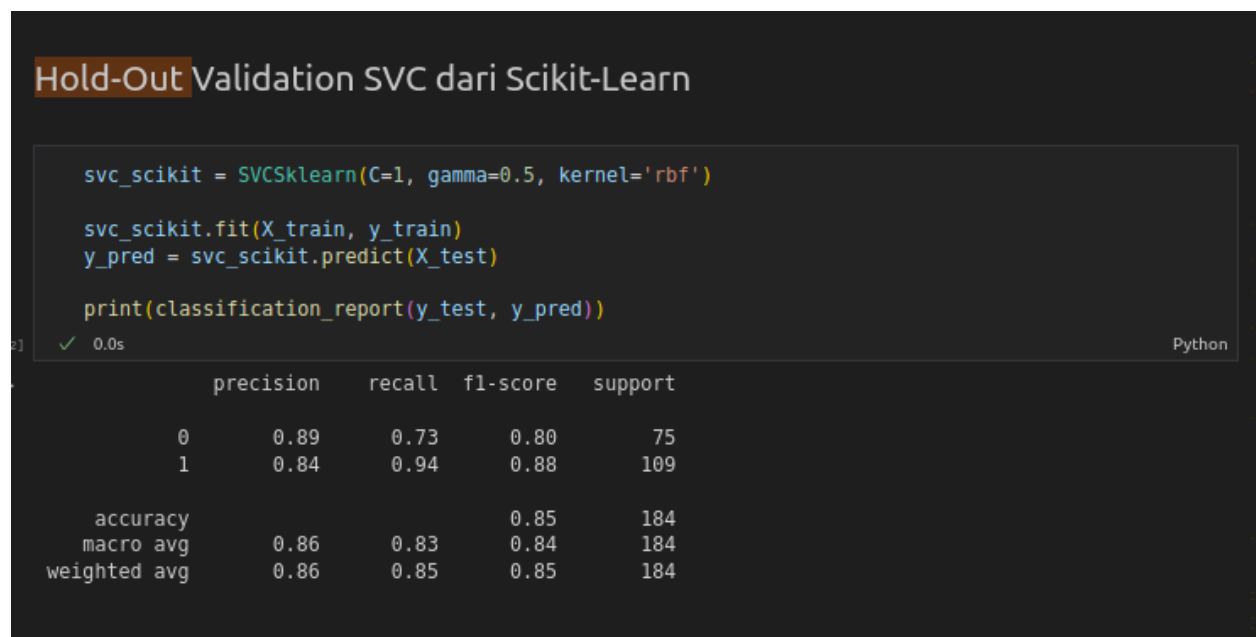
✓ 1.9s Python

--- Evaluating Model with K-Fold Cross Validation ---  
Rata-rata Akurasi: 74.19%  
Rata-rata Presisi: 70.94%  
Rata-rata Recall: 89.00%  
Rata-rata F1-Score: 78.85%

Gambar 1. K-Fold Cross-Validation SVM dari Scratch dan SVM dari Sklearn  
Sumber: Penulis



Gambar 2. Hold-Out Validation SVM dari Scratch  
Sumber: Penulis



Gambar 3. Hold-Out Validation SVM dari Sklearn  
Sumber: Penulis

Dari hasil evaluasi di atas dapat dilihat bahwa kedua model memberikan nilai yang relatif berdekatan. Variasi yang tinggi antara hasil hold-out dengan k-fold dapat terjadi karena adanya overfitting. Meskipun diberikan nilai parameter yang sama, model menghasilkan performa

berbeda yang dapat terjadi karena implementasi internalnya. SVM dari Scikit-Learn menggunakan libsvm yang sudah highly optimized, sedangkan SVM dari scratch memanfaatkan library cvxopt untuk menyelesaikan permasalahan quadratic programming. Selain itu, optimizer dari masing-masing model juga mungkin berbeda.

### **Ruang Improvement**

- Model hanya bisa menerima dua input kernel, dapat ditingkatkan lagi sehingga bisa menerima kernel lain.
- Eksperimen dengan multiple solver option misalnya dengan tambahan SMO (Sequential Minimal Optimization dengan menggunakan LIBSVM
- Menambahkan dekomposisi algoritma untuk efisiensi data besar.

### **Referensi:**

[1] *Penjelasan Sederhana tentang Apa Itu SVM?*, Medium, oleh Samsudiney, 25 Juli 2019.

[Daring]. Tersedia:

<https://medium.com/%40mohzulkiflikatili/list/pengenalan-pola-cbff7c3642b3>. [Diakses: 3 September 2025].

[2] *Support Vector Machine (SVM) Algorithm*, GeeksforGeeks, diperbarui 7 Agustus 2025.

[Daring]. Tersedia:

<https://www.geeksforgeeks.org/machine-learning/support-vector-machine-algorithm/>. [Diakses: 3 September 2025].

[3] *Implementing a Soft-Margin Kernelized Support Vector Machine Binary Classifier with Quadratic Programming in R and Python*, DataScienceCentral.com, oleh Sandipan Dey, 23 April 2018. [Daring]. Tersedia:

<https://www.datasciencecentral.com/implementing-a-soft-margin-kernelized-support-vector-machine/>. [Diakses: 3 September 2025].