

Nama : **Zahrani Cahya Priesa**

NIM : **1103223074**

Mata Kuliah : **Machine Learning**

## **Analisis Mendalam Kode: Implementasi Support Vector Machine (SVM)**

Pada chapter 5 ini menjelaskan cara kerja setiap model SVM yang digunakan dalam *notebook* ini, berfokus pada mekanisme algoritmik dan peran penentu dari hiperparameter kunci.

### **1. Klasifikasi SVM Linier: *Hard vs. Soft Margin***

**Model:** `sklearn.svm.LinearSVC`

Model ini dirancang untuk mencari hiperbidang (*hyperplane*) yang memisahkan kelas-kelas dengan margin terbesar.

- **Tujuan:** Memaksimalkan margin sambil meminimalkan pelanggaran data.
- **Pra-pemrosesan Data:** Kode selalu menggunakan `StandardScaler` sebelum melatih `LinearSVC`. Ini krusial karena SVM sangat sensitif terhadap skala fitur. Penskalaan memastikan bahwa semua fitur berkontribusi secara adil terhadap penentuan margin.

### **Peran Hiperparameter Kunci: C (Regularisasi)**

Hiperparameter	Peran	Implikasi Kinerja
C	Mengontrol <i>soft margin</i> —keseimbangan antara margin terlebar dan pelanggaran margin.	* C kecil: Regularisasi kuat. Margin menjadi lebih lebar, memungkinkan beberapa kesalahan klasifikasi (pelanggaran <i>soft margin</i> ). Mengurangi risiko <i>overfitting</i> . * C besar: Regularisasi lemah. Margin menjadi sempit, memaksa model untuk mengklasifikasikan hampir semua data latih dengan benar. Sensitif terhadap <i>outliers</i> .

**Kesimpulan Kode Linier:** Penggunaan LinearSVC dan penyetelan memungkinkan kita menemukan batas linier yang kokoh (*robust*) yang dapat bergeneralisasi dengan baik, meskipun ada *noise* dalam data.

## 2. Klasifikasi SVM Non-Linier: *Kernel Trick*

**Model:** `sklearn.svm.SVC`

Ketika data tidak dapat dipisahkan secara linier (seperti dataset `make_moons`), model ini menggunakan teknik non-linier yang kuat.

Konsep Kunci: *Kernel Trick*

1. Tujuan: Secara efektif memetakan data asli (dimensi rendah) ke ruang dimensi yang jauh lebih tinggi di mana data mungkin menjadi linier terpisah.
2. Cara Kerja: Daripada benar-benar melakukan transformasi dimensi tinggi (yang mahal), Fungsi *Kernel* secara implisit menghitung kesamaan antara pasangan titik data di ruang dimensi tinggi tersebut.

**Kernel yang Digunakan: RBF (*Radial Basis Function*)**

- **Tipe Kernel:** `kernel="rbf"` (sering juga disebut *Gaussian Kernel*), yang paling umum digunakan untuk data non-linier.

**Peran Hiperparameter Kunci: (Gamma)**

Hiperparameter	Peran	Implikasi Kinerja
Gamma	Mengontrol "radius" pengaruh dari sebuah <i>Support Vector</i> .	<p>* <b>besar:</b> Jarak pengaruh kecil. Setiap <i>support vector</i> hanya memengaruhi beberapa instans di sekitarnya. <b>Risiko overfitting tinggi</b> karena batas keputusan menjadi sangat tidak teratur.</p> <p>* <b>kecil:</b> Jarak pengaruh luas. Batas keputusan menjadi lebih halus dan bergeneralisasi. <b>Risiko underfitting lebih tinggi</b> pada data yang kompleks.</p>

**Kode Implementasi:** Eksperimen dengan kombinasi SVC(kernel="rbf", gamma=5, C=0.001) dan variasi lainnya menunjukkan bagaimana dan harus disetel bersamaan (*hyperparameter tuning*) untuk mencapai batas keputusan yang paling optimal.

### 3. Regresi SVM (SVR)

#### Model: LinearSVR dan SVR

SVM juga dapat digunakan untuk regresi. Berbeda dengan klasifikasi yang mencari margin *terluas* di antara kelas, regresi SVM mencari margin *terlebar* di sekitar prediksi linier.

Peran Hiperparameter Kunci: (Epsilon)

- Tujuan: Menentukan "jalan" toleransi di sekitar prediksi model.
- Cara Kerja:
  - Setiap titik data yang berada di dalam margin dianggap sebagai prediksi yang "baik" dan tidak berkontribusi pada fungsi biaya model.
  - Hanya titik data yang berada di luar margin (pelanggaran) yang memengaruhi dan meningkatkan fungsi biaya.

Hiperparameter	Peran	Implikasi Kinerja
<b>Epsilon</b>	Menentukan lebar margin toleransi pada Regresi SVM.	Epsilon besar: Margin toleransi lebar. Model lebih sederhana dan lebih tergeneralisasi (mengizinkan lebih banyak varian dalam data). Regularisasi kuat.
		Epsilon kecil: Margin toleransi sempit. Model dipaksa untuk lebih akurat mendekati setiap titik data. Risiko <i>overfitting</i> .

**Kesimpulan Kode Regresi:** Model SVR menggunakan kernel (seperti RBF) dan untuk menemukan fungsi non-linier yang sesuai dengan data sambil tetap memberikan toleransi kesalahan yang dikontrol.