Support Vector Machine (SVM)

**SVM** adalah algoritma pembelajaran mesin yang kuat dan serbaguna yang dapat digunakan untuk tugas klasifikasi (seperti mengklasifikasikan gambar) dan regresi (seperti memprediksi harga rumah). **SVM** bekerja dengan mencari hyperplane terbaik yang memisahkan data ke dalam kelas yang berbeda dengan margin terbesar.

# Konsep Dasar SVM

* **Hyperplane:** Sebuah hyperplane adalah batas keputusan yang memisahkan data ke dalam kelas yang berbeda. Dalam kasus dimensi, hyperplane adalah garis lurus, sedangkan dalam dimensi yang lebih tinggi, itu adalah bidang atau hyperplane.
* **Margin:** Margin adalah jarak antara hyperplane dan titik data terdekat dari masing-masing kelas. **SVM** bertujuan untuk memaksimalkan margin ini, yang mengarah pada model yang lebih umum dan lebih tahan terhadap overfitting.
* **Support Vector:** Titik data yang terletak di tepi margin atau di sisi yang salah dari margin disebut support vector. Support vector ini yang menentukan posisi hyperplane dan margin.
* **Kernel Trick:** Jika data tidak dapat dipisahkan secara linear, **SVM** dapat menggunakan kernel trick untuk memetakan data ke dimensi yang lebih tinggi dimana data dapat dipisahkan secara linear.

# Jenis-jenis SVM

1. **Linear SVm:** Digunakan ketika data dapat dipisahkan secara linear.
2. **Non-Linear SVM:** Digunakan ketika data tidak dapat dipisahkan secara. **SVM** menggunakan kernel trick untuk memetakan data ke dimensi yang lebih tinggi dimana data dapat dipisahkan secara linear. Beberapa kernel yang umum digunakan antara lain:

* **Polynomial Kernel**
* **Radial Basis Function (RBF) Kernel**
* **Sigmoid Kernel**

# Keunggulan SVM:

* **Efektif dalam ruang dimensi tinggi:** SVM dapat bekerja dengan baik pada data dengan banyak fitur.
* **Memory efficient:** SVM menggunakan subset dari data latih (support vectors) untuk membuat keputusan, sehingga lebih efisien dalam penggunaan memori.
* **Versatility:** SVM dapat digunakan untuk klasifikasi dan regresi, dan dapat menangani data linear maupun non-linear dengan menggunakan kernel trick.

# Kelemahan SVM:

* **Sensitif terhadap outlier:** Outlier dapat mempengaruhi posisi hyperplane dan margin.
* **Sulit untuk memilih kernel dan parameter:** Memilih kernel dan parameter yang tepat untuk SVM bisa menjadi tugas yang menantang.
* **Waktu pelatihan yang lama:** SVM bisa lambat untuk dilatih pada dataset yang besar.

**Tugas Hari 24**

Menggunakan dataset “Iris”

**Tugas**

1. **Membuat Model Linear SVM:**

* Gunakan SVC (Support Vector Classifier) dari scikit-learn untuk membuat model linear SVM.
* Bagi data menjadi data latih (80%) dan data uji (20%).
* Latih model pada data latih.
* Evaluasi kinerja model pada data uji menggunakan akurasi.

1. **Membuat Model Non-Linear SVM (RBF Kernel):**

* Gunakan SVC dengan kernel=’rbf’ untuk membuat model non-linear SVM dengan RBF kernel.
* Latih model pada data latih.
* Evaluasi kinerja model pada data uji menggunakan akurasi.

1. **Bandingkan Kinerja:**

* Bandingkan akurasi dari kedua model (linear dan non-linear).
* Model mana yang lebih memiliki kinerja lebih baik? Jelaskan alasan Anda.

**Contoh Kode (Scikit-Learn)**

from sklearn.svm import SVC

# ... (kode untuk membaca dan mempersiapkan dataset Iris)

# Linear SVM

linear\_svm = SVC(kernel='linear')

linear\_svm.fit(X\_train, y\_train)

# Evaluasi...

# Non-linear SVM (RBF Kernel)

rbf\_svm = SVC(kernel='rbf')

rbf\_svm.fit(X\_train, y\_train)

# Evaluasi...

**Selamat Mengerjakan Tugas! 🙂**

**Tugas**

1. **Membuat Model Linear SVM:**

**from sklearn.svm import SVC**

**# Linear SVM**

**linear\_svm = SVC(kernel='linear')**

**linear\_svm.fit(X\_train, y\_train)**

1. **Membuat Model Non-Linear SVM (RBF Kernel):**

**# Non-Linear SVC (rbf)**

**rbf\_svm = SVC(kernel='rbf')**

**rbf\_svm.fit(X\_train, y\_train)**

1. **Bandingkan Kinerja:**

* Evaluasi:

# Evaluasi Linear SVM

y\_pred\_linear = linear\_svm.predict(X\_test)

accuracy\_linear = accuracy\_score(y\_test, y\_pred\_linear)

print("\nLinear SVM:")

print(f'Akurasi: {accuracy\_linear:.2f}')

# Evaluasi Non-Linear SVM (RBF Kernel)

y\_pred\_rbf = rbf\_svm.predict(X\_test)

accuracy\_rbf = accuracy\_score(y\_test, y\_pred\_rbf)

print("\nNon-Linear SVM (RBF Kernel):")

print(f'Akurasi: {accuracy\_rbf:.2f}')

* Bandingkan Kinerja:

if accuracy\_linear > accuracy\_rbf:

print("\nLinear SVM memiliki akurasi yang lebih baik.")

elif accuracy\_rbf > accuracy\_linear:

print("\nNon-Linear SVM (RBF kernel) memiliki akurasi yang lebih baik.")

else:

print("\nKedua Model memiliki akurasi yang sama.")

# Penjelasan Kode:

1. **Membuat Model Linear SVM:**
   * linear\_svm = SVC(kernel='linear'): Membuat objek model SVM dengan kernel linear.
   * linear\_svm.fit(X\_train, y\_train): Melatih model pada data latih.
2. **Membuat Model Non-Linear SVM (RBF Kernel):**
   * rbf\_svm = SVC(kernel='rbf'): Membuat objek model SVM dengan kernel Radial Basis Function (RBF).
   * rbf\_svm.fit(X\_train, y\_train): Melatih model pada data latih.
3. **Evaluasi:**
   * y\_pred\_linear = linear\_svm.predict(X\_test): Melakukan prediksi pada data uji menggunakan model Linear SVM.
   * accuracy\_linear = accuracy\_score(y\_test, y\_pred\_linear): Menghitung akurasi model Linear SVM.
   * Langkah yang sama dilakukan untuk model Non-Linear SVM (RBF Kernel).
4. **Bandingkan Kinerja:**
   * Membandingkan nilai akurasi (accuracy\_linear dan accuracy\_rbf) dari kedua model.
   * Mencetak kesimpulan model mana yang memiliki akurasi lebih baik.

**Feedback:**

Kode Anda sudah sangat baik dan menunjukkan pemahaman yang baik tentang SVM. Berikut adalah beberapa saran untuk perbaikan:

* **Metrik Evaluasi Tambahan:** Selain akurasi, Anda juga dapat menggunakan metrik evaluasi lain seperti presisi, recall, dan F1-score untuk mendapatkan gambaran yang lebih lengkap tentang kinerja model.
* **Visualisasi:** Anda dapat membuat plot untuk memvisualisasikan batas keputusan (decision boundary) dari kedua model. Ini akan membantu Anda memahami bagaimana kedua model memisahkan data ke dalam kelas yang berbeda.
* **Hyperparameter Tuning:** Anda dapat mencoba mengatur hyperparameter model SVM (seperti C dan gamma) untuk melihat bagaimana hal tersebut mempengaruhi kinerja model. Grid search atau random search dapat membantu Anda menemukan kombinasi hyperparameter yang optimal.

**Contoh Kode Visualisasi Batas Keputusan:**

Python

# ... (kode sebelumnya)

# Visualisasi batas keputusan untuk Linear SVM

def plot\_decision\_boundary(model, ax, X, y):

x\_min, x\_max = X[:, 0].min() - 1, X[:, 0].max() + 1

y\_min, y\_max = X[:, 1].min() - 1, X[:, 1].max() + 1

xx, yy = np.meshgrid(np.arange(x\_min, x\_max, 0.1),

np.arange(y\_min, y\_max, 0.1))

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

Z = Z.reshape(xx.shape)

ax.contourf(xx, yy, Z, alpha=0.4, cmap=plt.cm.coolwarm)

ax.scatter(X[:, 0], X[:, 1], c=y, s=20, edgecolor='k')

# Visualisasi untuk kedua model

fig, axs = plt.subplots(1, 2, figsize=(12, 6))

plot\_decision\_boundary(linear\_svm, axs[0], X, y)

axs[0].set\_title('Linear SVM')

plot\_decision\_boundary(rbf\_svm, axs[1], X, y)

axs[1].set\_title('Non-Linear SVM (RBF Kernel)')

plt.show()