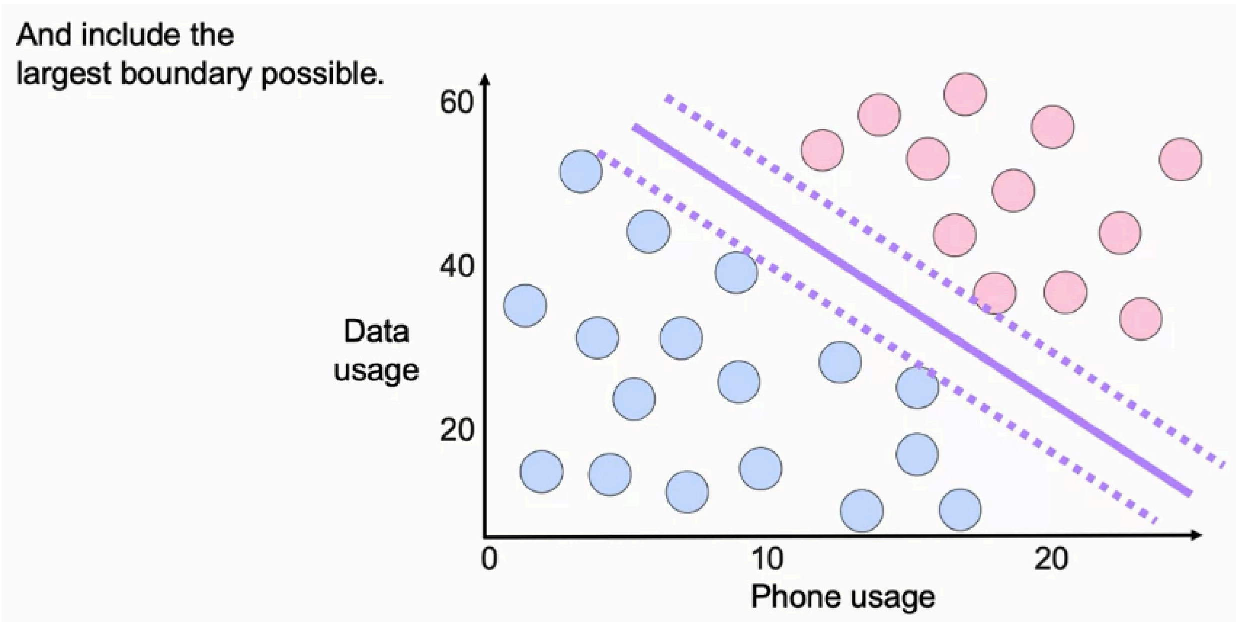
Course3_Module3

Support Vector Machine



1) Bài toán

- Bài toán phân loại nhị phân giám sát với tập huấn luyện $\mathscr{D}$ = $(x_i, y_i)_{i=1}^m$, trong đó $x_i \in \mathbb{R}^d, y_i \in \{-1, +1\}$.
- Mục tiêu: tìm siêu phẳng (decision boundary) tách lớp và tối đa hóa biên (margin) để tổng quát hóa tốt.
- Hàm quyết định tuyến tính:
    - Điểm số: $f(x) = w^T x + b$
    - Dự đoán: $\hat{y} = sign(f(x))$

Định nghĩa biên:

- Functional margin của (x_i, y_i) : $\gamma_i^f = y_i(w^T x_i + b)$
- Geometric margin (khoảng cách tới siêu phẳng):

$$\gamma_i = \frac{y_i\left(w^T x_i + b\right)}{\|w\|}$$

- Biên của dữ liệu là giá trị nhỏ nhất trên các mẫu.

Nguyên lý max-margin (hard-margin, dữ liệu phân tách hoàn hảo): tối đa hóa biên ≡ tối thiểu hóa $\|w\|$ với ràng buộc phân loại đúng và biên ≥ 1.

---

2) SVM hard-margin (trường hợp phân tách được)

Bài toán tối ưu (primal):

$$\begin{aligned} \min_{w,b} \quad & \tfrac{1}{2}\|w\|^2 \\ \text{s.t.} \quad & y_i(w^T x_i + b) \geq 1, \quad i = 1, \ldots, m \end{aligned}$$

- Hệ số 1/2 giúp đạo hàm gọn hơn.
- Tối đa hóa biên tương đương tối thiểu hóa $\|w\|$.

KKT và bài toán đối ngẫu (phác thảo): đưa vào bội số Lagrange $\alpha_i \geq 0$.

Đối ngẫu:

$$\begin{aligned} \max_{\alpha} \quad & \sum_{i=1}^{m} \alpha_i - \tfrac{1}{2} \sum_{i=1}^{m} \sum_{j=1}^{m} \alpha_i \alpha_j \, y_i y_j \, x_i^T x_j \\ \text{s.t.} \quad & \sum_{i=1}^{m} \alpha_i y_i = 0, \quad \alpha_i \geq 0. \end{aligned}$$

Dạng nghiệm (chỉ phụ thuộc support vectors):

$$w = \sum_{i=1}^{m} \alpha_i y_i x_i, \qquad f(x) = \sum_{i=1}^{m} \alpha_i y_i \, x_i^T x + b.$$

Những điểm có $\alpha_i > 0$ là support vectors; chúng nằm trên hoặc trong biên và quyết định siêu phẳng.

---

3) SVM soft-margin (không phân tách hoàn hảo) và hinge loss

- Dữ liệu thực thường không phân tách hoàn hảo. Dùng biến trượt $\xi_i \geq 0$ cho phép vi phạm.

Primal (C-SVM):

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

- $C > 0$ điều chỉnh đánh đổi: C lớn phạt vi phạm mạnh hơn, biên hẹp hơn, khớp dữ liệu huấn luyện hơn.

Dạng không ràng buộc với hinge loss:

- Hinge loss cho mẫu i:

$$\ell_{\text{hinge}}(y_i, f(x_i)) = \max\big(0, 1 - y_i f(x_i)\big).$$

- Tối ưu rủi ro có regularization:

$$\min_{w,b} \ \tfrac{1}{2}\|w\|^2 + C\sum_{i=1}^{m} \max\big(0, 1 - y_i(w^T x_i + b)\big).$$

Ngoại lai: các điểm ở phía sai và xa biên có loss lớn; SVM tập trung vào điểm biên, khá vững với nhiều inlier nhưng vẫn nhạy với outlier mạnh tùy C.

---

4) So sánh Logistic Regression và SVM

- Hàm mất mát: Logistic dùng log-loss, SVM dùng hinge-loss.
  - Log-loss tiệm cận 0 nhưng hiếm khi bằng 0; hinge-loss = 0 khi biên ≥ 1.
- Đầu ra: Logistic có xác suất qua $\sigma(z) = 1/(1 + e^{-z})$. SVM tiêu chuẩn không xác suất; có thể hiệu chỉnh (Platt scaling).
- Biên quyết định: cả hai tuyến tính trong không gian gốc; SVM nhấn mạnh tối đa biên.
- Regularization: thường dùng L2; trong SVM, C là tham số phạt, nghịch với độ mạnh regularization.

5) SVM phi tuyến với kernel trick

Động cơ: Một số dữ liệu không tách tuyến tính trong không gian gốc nhưng tách được trong không gian đặc trưng bậc cao $\phi(x)$.

Kernel trick: tránh biểu diễn tường minh $\phi(x)$. Dùng hàm kernel

$$K(x,z) = \phi(x)^T \phi(z)$$

để tính tích vô hướng trong không gian đặc trưng trực tiếp ở không gian đầu vào.

Đối ngẫu với kernel:

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

Hàm quyết định:

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

Ưu điểm kernelized SVM:

- Độ chính xác tốt trên nhiều bộ dữ liệu
- Linh hoạt lựa chọn kernel, có thể tùy biến theo miền ứng dụng
- Hoạt động tốt cả dữ liệu ít chiều và nhiều chiều

Nhược điểm:

- Thời gian và bộ nhớ huấn luyện tăng nhanh theo số mẫu
- Nhạy với chuẩn hóa đặc trưng và siêu tham số
- Không xác suất, khó diễn giải vì sao dự đoán được đưa ra

---

6) Các kernel phổ biến và công thức

- Linear:

$$K_{\text{lin}}(x,z) = x^T z.$$

- Polynomial (bậc d, hệ số chệch $r \geq 0$):

$$K_{\text{poly}}(x,z) = (\gamma\, x^T z + r)^d.$$

- Radial Basis Function (Gaussian):

$$K_{\text{rbf}}(x,z) = \exp\big(-\gamma\,\|x-z\|^2\big).$$

γ điều khiển bán kính ảnh hưởng: γ lớn ⇒ quyết định phức tạp hơn, dễ overfit nếu quá lớn.

- Sigmoid (liên hệ mạng nơ-ron 2 lớp):

$$K_{\text{sig}}(x,z) = \tanh(\gamma\, x^T z + r).$$

Cực kỳ quan trọng phải chuẩn hóa đặc trưng với RBF, polynomial, sigmoid để độ lớn tích vô hướng và khoảng cách ở mức hợp lý.

---

7) Trực giác hình học Gaussian kernel

- Ánh xạ RBF tạo không gian vô hạn chiều; độ tương đồng giảm theo bình phương khoảng cách Euclid.
- Biên quyết định trở thành các đường cong trơn; khi γ tăng, biên có thể uốn lượn mạnh.
- Biên lớn trong không gian đặc trưng tương ứng các đường cong trong không gian gốc.

---

8) Regularization và siêu tham số

- C: điều khiển đánh đổi giữa rộng biên và vi phạm.
- γ: điều khiển mức “cục bộ” của ảnh hưởng trong RBF, sigmoid, và cả polynomial qua scale.

- Bậc d và r cho polynomial: mức độ phức tạp và offset.
- Chọn mô hình qua cross-validation. Ví dụ lưới:
  - $C \in \{1e-2, 1e-1, 1, 10, 100\}$
  - $\gamma \in \{1e-3, 1e-2, 1e-1, 1\}$
  - $d \in \{2, 3, 4\}$

---

9) Nhạy cảm với ngoại lai và biên

- Mẫu có $y_i f(x_i) < 1$ ở trong biên hoặc bị phân loại sai và chịu hinge loss tuyến tính.
- C lớn giảm khoan dung, dễ fit ngoại lai; C nhỏ tăng khoan dung, mở rộng biên.

---

10) Hệ số và diễn giải (SVM tuyến tính)

- Độ lớn $|w_j|$ gợi ý tầm quan trọng đặc trưng j với biên quyết định.
- Với SVM kernel, w ẩn trong không gian đặc trưng; muốn diễn giải cần công cụ bổ trợ.

---

11) Ghi chú triển khai và cú pháp

- SVM tuyến tính (scikit-learn):

```
from sklearn.svm import LinearSVC
from sklearn.pipeline import Pipeline
from sklearn.preprocessing import StandardScaler

clf = Pipeline([
    ("scaler", StandardScaler()),
    ("svm", LinearSVC(C=1.0, loss="hinge"))
])
clf.fit(X_train, y_train)
```

- SVM kernel (RBF):

```
from sklearn.svm import SVC
from sklearn.model_selection import GridSearchCV
from sklearn.preprocessing import StandardScaler
from sklearn.pipeline import Pipeline

pipe = Pipeline([
    ("scaler", StandardScaler()),
    ("svm", SVC(kernel="rbf"))
])
param_grid = {
    "svm__C": [0.1, 1, 10, 100],
    "svm__gamma": [1e-3, 1e-2, 1e-1, 1]
}
search = GridSearchCV(pipe, param_grid=param_grid, scoring="f1_macro", cv
=5)
search.fit(X_train, y_train)
```

- Ví dụ kernel đa thức:

```
SVC(kernel="poly", degree=3, coef0=1.0, gamma="scale")
```

- Xác suất (nếu cần):

```
SVC(kernel="rbf", probability=True)
```

Mẹo hiệu năng:

- Luôn chuẩn hóa đặc trưng cho SVM kernel.
- Với n lớn, huấn luyện SVC(kernel) có độ phức tạp ~ $O(n^2)$–$O(n^3)$, bộ nhớ ~ $O(n^2)$.
- Dùng LinearSVC hoặc SGD cho bài toán thưa, rất lớn.

---

12) Quy trình end-to-end (tóm tắt)

1. Chia train và validation hoặc dùng CV.
2. Chuẩn hóa đặc trưng.
3. Chọn mô hình: LinearSVC cho tuyến tính; SVC với kernel cho phi tuyến.
4. Tinh chỉnh C, γ, bậc d, hệ số r bằng CV.
5. Huấn luyện trên toàn bộ train với tham số tốt nhất.
6. Đánh giá bằng metric phù hợp (f1_macro nếu mất cân bằng lớp).
7. Nếu cần xác suất, bật probability=True hoặc hiệu chỉnh Platt.
8. Soát lỗi và hình dạng biên; điều chỉnh C và γ tương ứng.

---

13) Tóm tắt công thức then chốt

- Hàm quyết định:

$$\text{tuyến tính: } f(x) = w^T x + b \qquad \text{kernel: } f(x) = \sum_{i \in SV} \alpha_i y_i K(x_i, x) + b$$

- Biên hình học (với functional margin = 1):

$$\gamma = \frac{1}{\|w\|}$$

- Hard-margin primal:

$$\min \tfrac{1}{2}\|w\|^2 \quad \text{s.t.} \quad y_i(w^T x_i + b) \geq 1$$

- Soft-margin primal:

$$\min \tfrac{1}{2}\|w\|^2 + C \sum_i \xi_i \quad \text{s.t.} \quad y_i(w^T x_i + b) \geq 1 - \xi_i,\ \xi_i \geq 0$$

- Hinge loss:

$$\max\big(0,\ 1 - y_i f(x_i)\big)$$

- RBF kernel:

$$\exp\left(-\gamma\,\|x - z\|^2\right)$$

- Polynomial kernel:

$$(\gamma x^T z + r)^d$$

- Sigmoid kernel:

$$\tanh(\gamma x^T z + r)$$

---

14) Hướng dẫn diễn giải thực tế

- Train cao nhưng validation thấp: giảm γ hoặc C để đơn giản hóa biên.
- Underfit: tăng C, tăng γ, hoặc tăng bậc đa thức.
- Nhiều đặc trưng và nhiều điểm: ưu tiên mô hình tuyến tính hoặc xấp xỉ.

---

15) Learning recap

- SVM tìm siêu phẳng tối đa biên.
- Hinge loss và C thể hiện đánh đổi bias-variance.
- Kernel cho phép biên phi tuyến nhờ tích vô hướng trong không gian đặc trưng.
- RBF là mặc định mạnh nhưng cần chuẩn hóa và tinh chỉnh C, γ.