# KHAI THÁC DỮ LIỆU

Nguyễn Xuân Việt Đức - 22280012

Bài tập thực hành - Lần 9

## Bài 9: SỰ PHÂN LỚP DỮ LIỆU

### II. Tóm tắt lý thuyết:

### 1. Tổng quan lý thuyết SVM

Support Vector Machines (SVM) là thuật toán học có giám sát được phát triển bởi Vladimir Vapnik và cộng sự tại AT&T Bell Laboratories vào những năm 1990. SVM ban đầu được thiết kế cho bài toán phân loại nhị phân, nhưng sau đó đã được mở rộng cho nhiều loại bài toán khác như hồi quy và phân cụm.

Ý tưởng cơ bản của SVM là tìm một siêu phẳng (hyperplane) tối ưu để phân tách dữ liệu thành các lớp khác nhau sao cho lề (margin) giữa siêu phẳng và các điểm dữ liệu gần nhất là lớn nhất có thể. Điều này được gọi là nguyên tắc "Maximum Margin".

### 2. Hard Margin SVM

#### 2.1. Nguyên lý:

Hard Margin SVM áp dụng trong trường hợp dữ liệu có thể phân tách tuyến tính hoàn toàn. Mục tiêu là tìm siêu phẳng với lề lớn nhất sao cho tất cả các điểm dữ liệu được phân loại chính xác.

#### 2.2. Định nghĩa toán học:

Giả sử chúng ta có tập dữ liệu huấn luyện $\{(\mathbf{x}_i, y_i)\}_{i=1}^n$, trong đó $\mathbf{x}_i \in \mathbb{R}^d$ là vector đặc trưng và $y_i \in \{-1, 1\}$ là nhãn lớp. Siêu phẳng được định nghĩa bởi:

$$f(\mathbf{x}) = \mathbf{w}^T\mathbf{x} + b = 0 \tag{1}$$

Trong đó $\mathbf{w} \in \mathbb{R}^d$ là vector pháp tuyến của siêu phẳng và $b \in \mathbb{R}$ là độ lệch.

Bài toán tối ưu hóa để tìm siêu phẳng tối ưu trong Hard Margin SVM được phát biểu như sau:

$$\min_{\mathbf{w},b} \quad \frac{1}{2}\|\mathbf{w}\|^2 \tag{2}$$

$$\text{subject to} \quad y_i(\mathbf{w}^T\mathbf{x}_i + b) \geq 1, \quad \forall i = 1, 2, \ldots, n \tag{3}$$

Khoảng cách từ siêu phẳng đến các điểm gần nhất (support vectors) là $\frac{1}{\|\mathbf{w}\|}$.
Do đó, việc cực tiểu hóa $\|\mathbf{w}\|^2$ tương đương với việc cực đại hóa margin.

### 2.3. Hạn chế:

Hard Margin SVM có một số hạn chế quan trọng:

- Yêu cầu dữ liệu phải phân tách tuyến tính hoàn toàn
- Rất nhạy cảm với nhiễu và outliers
- Không thể áp dụng cho nhiều tập dữ liệu thực tế

## 3. Soft Margin SVM

### 3.1. Nguyên Lý:

Soft Margin SVM là phiên bản mở rộng của Hard Margin SVM, cho phép một số điểm dữ liệu vi phạm ràng buộc margin hoặc thậm chí bị phân loại sai. Điều này làm cho mô hình linh hoạt hơn và có khả năng xử lý dữ liệu có nhiễu hoặc không phân tách tuyến tính hoàn toàn.

### 3.2. Định nghĩa toán học:

Bài toán tối ưu hóa trong Soft Margin SVM được mở rộng bằng cách đưa vào các biến hỗ trợ $\xi_i \geq 0$ (slack variables):

$$\min_{\mathbf{w},b,\boldsymbol{\xi}} \quad \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{n}\xi_i \tag{4}$$

$$\text{subject to} \quad y_i(\mathbf{w}^T\mathbf{x}_i + b) \geq 1 - \xi_i, \quad \forall i = 1, 2, \ldots, n \tag{5}$$

$$\xi_i \geq 0, \quad \forall i = 1, 2, \ldots, n \tag{6}$$

Trong đó $C > 0$ là hằng số điều chỉnh (regularization constant) kiểm soát sự đánh đổi giữa việc tối đa hóa margin và giảm thiểu lỗi phân loại.
Tham số $C$ trong Soft Margin SVM đóng vai trò rất quan trọng:

- $C$ lớn: Đặt trọng số cao cho việc phân loại chính xác, margin hẹp hơn
- $C$ nhỏ: Cho phép nhiều lỗi phân loại hơn để đạt được margin rộng hơn

Việc lựa chọn giá trị $C$ phù hợp thường được thực hiện thông qua kiểm chứng chéo (cross-validation).

## 4. Kernel SVM

### 4.1. Nguyên lý

Kernel SVM mở rộng khả năng của SVM để xử lý dữ liệu không phân tách tuyến tính. Ý tưởng chính là ánh xạ dữ liệu từ không gian đặc trưng gốc sang một không gian đặc trưng có chiều cao hơn, nơi dữ liệu có thể phân tách tuyến tính.

### 4.2. Kỹ thuật kernel (Kernel Trick)

Thay vì tính toán ánh xạ $\phi : \mathbb{R}^d \to \mathbb{R}^D$ ($D \gg d$) một cách trực tiếp, kỹ thuật kernel cho phép tính toán tích vô hướng trong không gian đặc trưng cao chiều thông qua một hàm kernel $K$:

$$K(\mathbf{x}_i, \mathbf{x}_j) = \phi(\mathbf{x}_i)^T \phi(\mathbf{x}_j) \tag{7}$$

### 4.3. Hàm dự đoán

Hàm dự đoán của Kernel SVM có dạng:

$$f(\mathbf{x}) = \text{sign}\left(\sum_{i=1}^{n} \alpha_i y_i K(\mathbf{x}_i, \mathbf{x}) + b\right) \tag{8}$$

Trong đó $\alpha_i \geq 0$ là các hệ số Lagrange thu được từ bài toán đối ngẫu.

### 4.4. Các kernel phổ biến

1. **Kernel tuyến tính:** $K(\mathbf{x}_i, \mathbf{x}_j) = \mathbf{x}_i^T \mathbf{x}_j$
2. **Kernel đa thức:** $K(\mathbf{x}_i, \mathbf{x}_j) = (\mathbf{x}_i^T \mathbf{x}_j + c)^d$, với $c \geq 0$ và $d \in \mathbb{N}$
3. **Kernel RBF (Gaussian):** $K(\mathbf{x}_i, \mathbf{x}_j) = \exp\left(-\gamma \|\mathbf{x}_i - \mathbf{x}_j\|^2\right)$, với $\gamma > 0$
4. **Kernel sigmoid:** $K(\mathbf{x}_i, \mathbf{x}_j) = \tanh(\alpha \mathbf{x}_i^T \mathbf{x}_j + c)$, với $\alpha > 0$ và $c < 0$

### 4.5. Tham số gamma ($\gamma$)

Đối với Kernel RBF, tham số $\gamma$ kiểm soát ảnh hưởng của từng mẫu:

- $\gamma$ lớn: Ảnh hưởng của mỗi mẫu giảm nhanh theo khoảng cách, tạo ra đường biên quyết định "gồ ghề" hơn
- $\gamma$ nhỏ: Ảnh hưởng của mỗi mẫu giảm chậm, tạo ra đường biên quyết định mượt mà hơn

## 5. So sánh và ứng dụng

### 5.1. Hard margin vs Soft margin

- **Hard Margin:**
  - Không cho phép lỗi phân loại
  - Yêu cầu dữ liệu phân tách tuyến tính hoàn toàn
  - Nhạy cảm với nhiễu và outliers
- **Soft Margin:**
  - Cho phép một số lỗi phân loại
  - Linh hoạt hơn với dữ liệu nhiễu
  - Có thể áp dụng cho dữ liệu không phân tách tuyến tính hoàn toàn
  - Yêu cầu điều chỉnh tham số $C$

### 5.2. Lựa chọn Kernel

- **Kernel tuyến tính:** Phù hợp với dữ liệu có số chiều cao, đơn giản và hiệu quả tính toán
- **Kernel đa thức:** Phù hợp khi các đặc trưng có quan hệ phi tuyến bậc thấp
- **Kernel RBF:** Linh hoạt nhất, thích hợp với nhiều loại dữ liệu phức tạp, nhưng đòi hỏi điều chỉnh tham số cẩn thận
- **Kernel sigmoid:** Thường ít được sử dụng hơn, có mối liên hệ với mạng neural

### 5.3. Ứng dụng

SVM được ứng dụng rộng rãi trong nhiều lĩnh vực:

- **Phân loại văn bản và phân tích cảm xúc:** SVM hiệu quả với dữ liệu văn bản có số chiều cao
- **Nhận dạng hình ảnh:** Sử dụng SVM cho phân loại đối tượng trong ảnh
- **Phân tích dữ liệu sinh học:** Phân loại gene, dự đoán cấu trúc protein
- **Dự đoán tài chính:** Phân tích xu hướng thị trường
- **Phát hiện gian lận và bất thường:** SVM hiệu quả trong việc phát hiện các mẫu bất thường

Support Vector Machines là một phương pháp học máy mạnh mẽ với nền tảng lý thuyết vững chắc từ lý thuyết học thống kê. Với khả năng xử lý dữ liệu phức tạp thông qua kỹ thuật kernel, SVM vẫn là một thuật toán quan trọng trong học máy hiện đại, đặc biệt trong các bài toán có số lượng đặc trưng lớn hơn số lượng mẫu.

## III. Nội dung thực hành

Áp dụng với tập dữ liệu banknote authentication từ UCI Machine Learning Repository.

```
banknote_data = fetch_openml(name='banknote-authentication', version=1, as_frame=True)

X = banknote_data.data.values
y = banknote_data.target.values
```

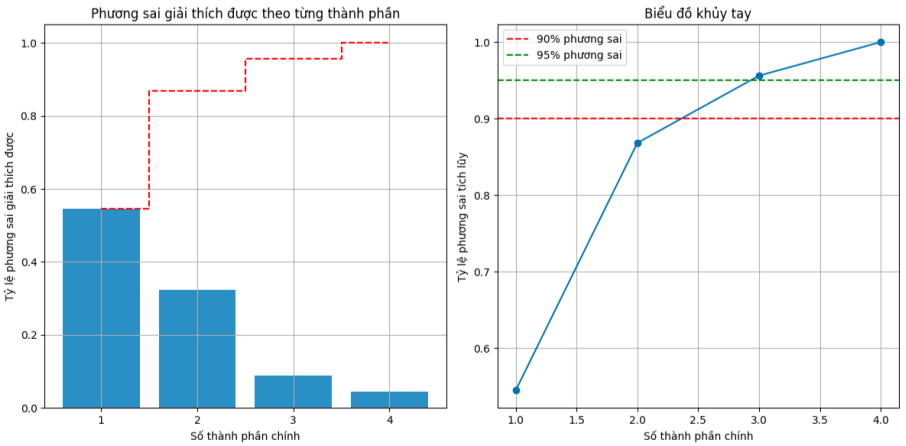
```
X
```

```
array([[  3.6216 ,   8.6661 ,  -2.8073 ,  -0.44699],
       [  4.5459 ,   8.1674 ,  -2.4586 ,  -1.4621 ],
       [  3.866  ,  -2.6383 ,   1.9242 ,   0.10645],
       ...,
       [ -3.7503 , -13.4586 ,  17.5932 ,  -2.7771 ],
       [ -3.5637 ,  -8.3827 ,  12.393  ,  -1.2823 ],
       [ -2.5419 ,  -0.65804,   2.6842 ,   1.1952 ]], shape=(1372, 4))
```

```
y
```

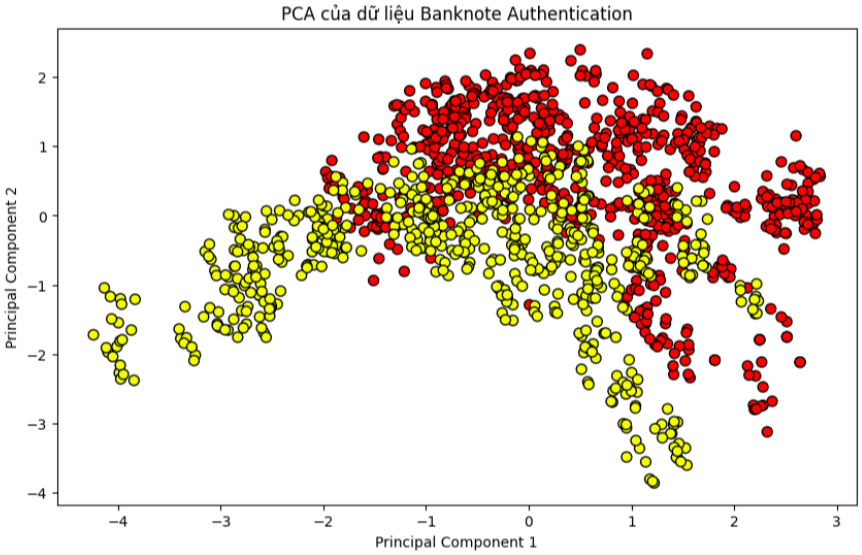
```
['1', '1', '1', '1', '1', ..., '2', '2', '2', '2', '2']
Length: 1372
Categories (2, object): ['1', '2']
```

PCA:

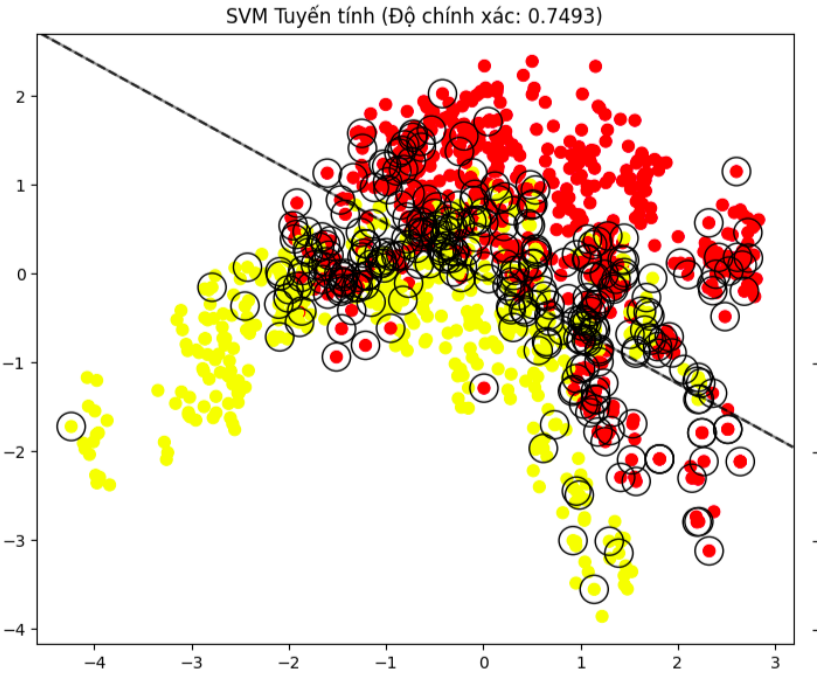



Qua đồ thị khủy tay thì số lượng component tối ưu cho dữ liệu là 2, giữ đượng gần 80 phần trăm thông tin.

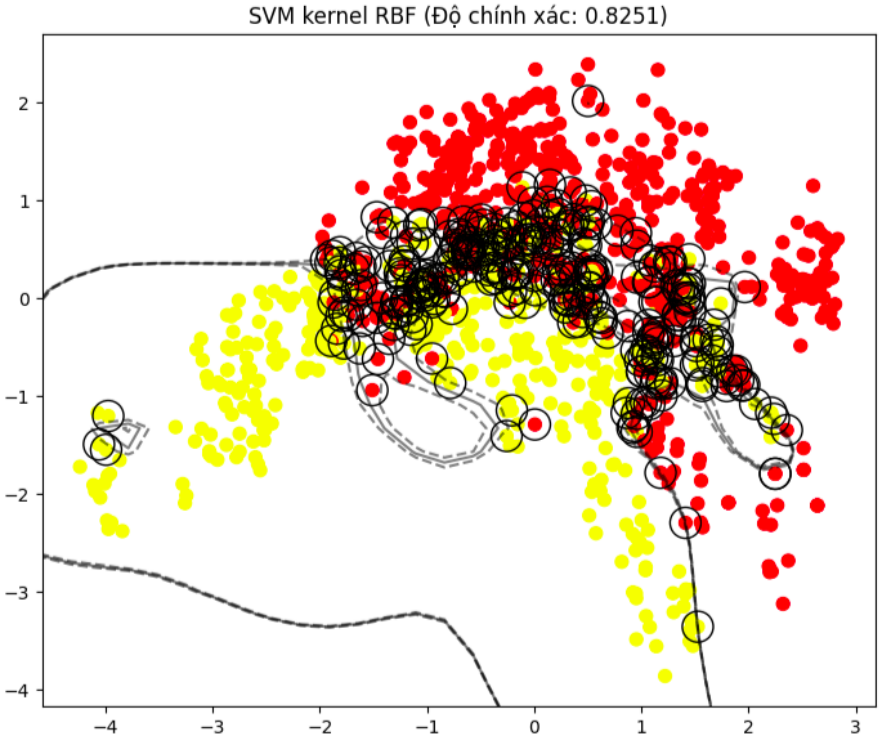
Đồ thị 2D biểu diễn 2 thành phần chính:




Kết quả sử dụng (SVC) của Scikit-Learn để huấn luyện mô hình SVM và vẽ các biên quyết định của SVM với kernel = Linear trên không gian 2D:



Kết quả sử dụng (SVC) của Scikit-Learn để huấn luyện mô hình SVM và vẽ các biên quyết định của SVM với kernel = rbf trên không gian 2D:




Tổng kết:

```
Số lượng support vectors (SVM tuyến tính): 317
Số lượng support vectors (SVM kernel RBF): 324

Độ chính xác trên toàn bộ đặc trưng:
SVM tuyến tính: 0.9883
SVM kernel RBF: 1.0000
```

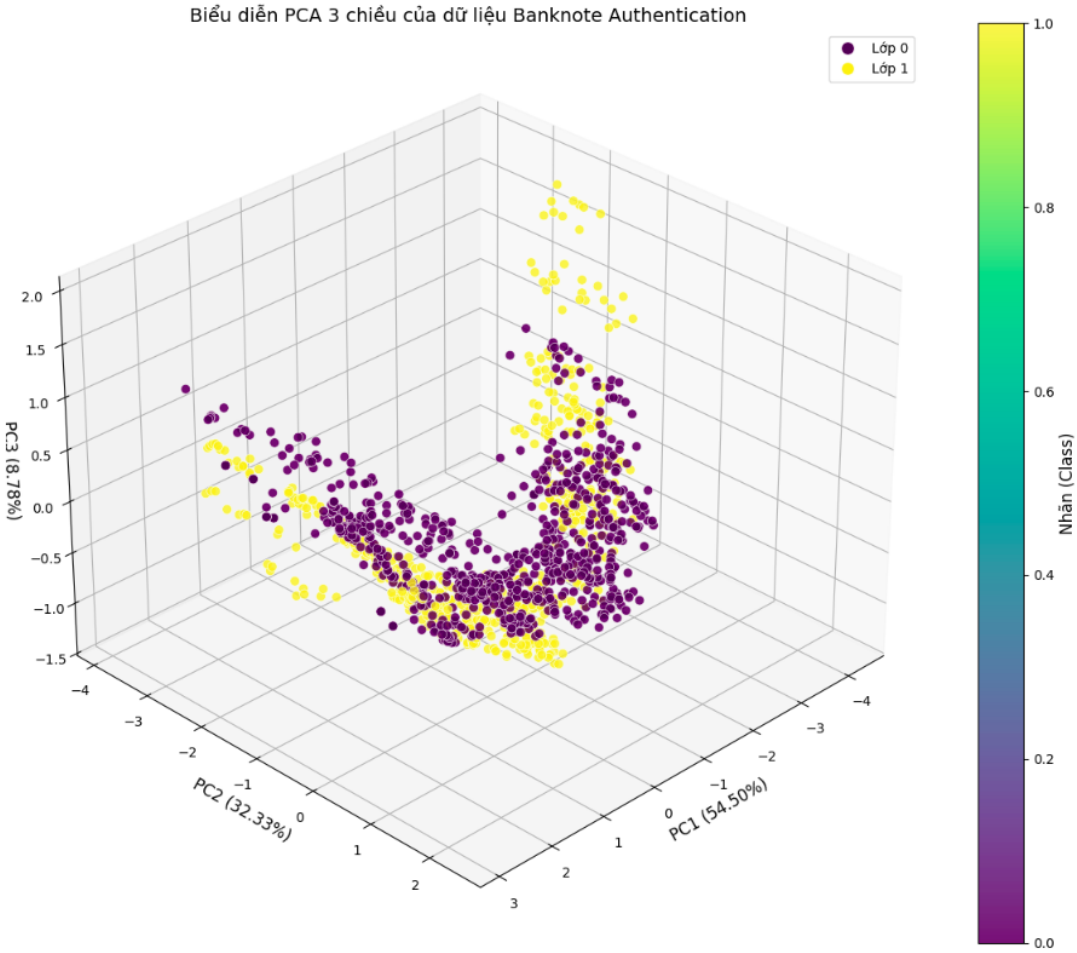
**Dữ liệu trong không gian 2D không phân tách tuyến tính tốt:** Sự chênh lệch đáng kể về độ chính xác giữa SVM tuyến tính (74.93) và SVM kernel RBF (82.51) cho thấy dữ liệu có cấu trúc phi tuyến tính trong không gian 2D.

**Đường biên quyết định:** Đường biên tuyến tính không thể bao quát được sự phân bố phức tạp của dữ liệu, trong khi đường biên phi tuyến tính của kernel RBF thích nghi tốt hơn với cấu trúc dữ liệu.

Đồ thị 3D biểu diễn 3 thành phần chính:

Phương sai giải thích được bởi từng thành phần: [0.54497602 0.32328872 0.08784561]

Tổng phương sai giải thích được: 0.9561




Vẫn có vùng chồng lấn, đặc biệt ở khu vực trung tâm.

Cấu trúc dữ liệu không tuyến tính, giải thích vì sao SVM kernel RBF hiệu quả hơn SVM tuyến tính.

Kết quả sử dụng (SVC) của Scikit-Learn để huấn luyện mô hình SVM và vẽ các biên quyết định của SVM với kernel = linear và rbf trên không gian 3D: