ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH
TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN

SUPPORT VECTOR MACHINE

NHẬP MÔN HỌC MÁY

21KHMT2

GIẢNG VIÊN
PHẠM TRỌNG NGHĨA
BÙI DUY ĐĂNG
NGUYỄN NGỌC ĐỨC

SINH VIÊN
NGUYỄN HỒNG HẠNH 21127503

Mục lục

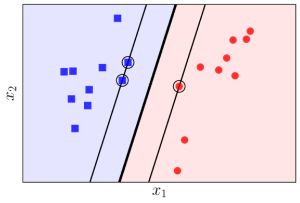
I Lý thuyết mô hình SVM (Support Vector Machine)

1 SVM

- **Bài toán:** giả sử hai lớp dữ liệu là linearly separable, tìm được vô số mặt phẳng phân chia hai lớp dữ liệu, vậy đâu là mặt phân chia tốt nhất?

- **Ý tưởng chính:** tìm mặt phân cách mà khoảng cách từ mặt phân cách tới các điểm gần nhất của mỗi lớp (margin) là như nhau và margin là lớn nhất.

- **Support vectors:** là những điểm nằm gần mặt phân chia nhất. Mặt phân cách có thể được xác định chỉ dựa trên những điểm support vectors. Số lượng những điểm support vectors chiếm số lượng rất nhỏ trong số N điểm. Việc mô hình chỉ tập trung vào số lượng hữu hạn các support vectors sẽ giúp giảm khả năng bị overfitting, giảm độ phức tạp của mô hình và tăng khả năng tổng quát hoá.



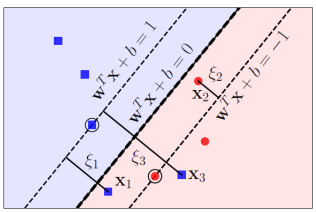
SVM với dữ liệu linearly separable

2 Soft-margin SVM

- **Bài toán:** làm sao để tìm tìm mặt phân cách trong trường hợp dữ liệu gần linearly separable (không tồn tại mặt phẳng nào phân chia hoàn toàn hai lớp dữ liệu), hoặc dữ liệu linear separable nhưng xuất hiện một vài điểm nhiễu khiến cho margin rất nhỏ, đường phân lớp bị lệch một phía?

- **Ý tưởng chính:** hy sinh một vài điểm ở gần khu vực biên giới giữa hai lớp để tạo được mặt phân chia tốt, hy sinh các điểm nhiễu để được margin tốt hơn. Mục tiêu là tối đa margin và tối thiểu sự hy sinh.

- **Siêu tham số C:** là một hằng số dương quyết định mức độ hy sinh. Nếu C nhỏ, việc hy sinh cao hay thấp không ảnh hưởng nhiều tới giá trị của hàm mục tiêu, thuật toán sẽ tập trung tìm margin lớn nhất. Nếu C lớn, thuật toán sẽ tập trung giảm sự hy sinh và cho ra margin nhỏ hơn.



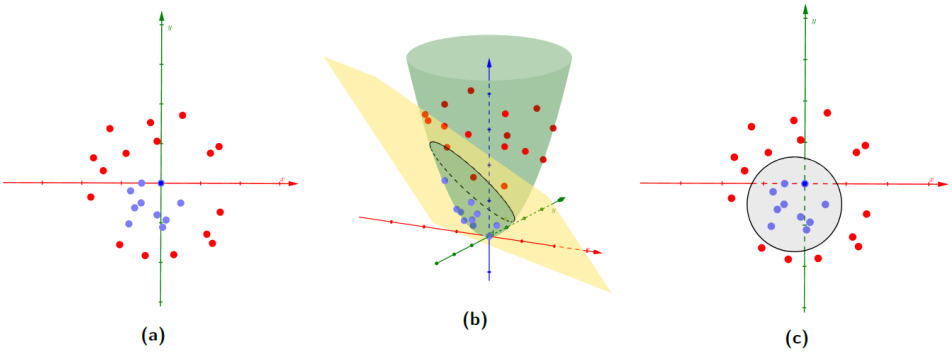
Soft-margin SVM với dữ liệu gần linearly separable

3 Kernel SVM

- **Bài toán:** thực tế có trường hợp dữ liệu không linearly separable, làm thế nào để tìm được mặt phân cách?

- **Ý tưởng chính:** Biến đổi dữ liệu không linearly separable trở nên linearly separable ở một không gian mới nhiều chiều hơn. Tuy nhiên, thay vì chuyển đổi dữ liệu sang một không gian mới nhiều chiều hơn, ta sử dụng hàm kernel để tính toán, mô tả mối quan hệ giữa hai điểm dữ liệu khi chúng ở trong không gian mới.

- **Siêu tham số γ trong RBF Kernel:** γ càng nhỏ, sự ảnh hưởng lên mặt phân chia sẽ bao gồm thêm các điểm ở phía xa, cho ra kết quả đơn giản nhưng mặt phân chia sẽ không đủ tốt, không đủ phức tạp. γ càng lớn, mặt phân chia sẽ càng chỉ phụ thuộc vào những điểm gần nó nhất, mô hình sẽ cho ra kết quả phức tạp và có xu hướng bị overfitting.

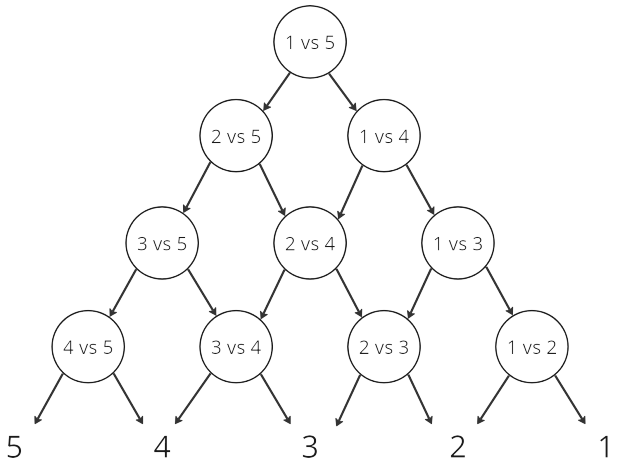


Ví dụ về Kernel SVM

4 Multiclass SVM

- **One-against-one:** xây dựng $\frac{k(k-1)}{2}$ bộ binary classifiers cho từng cặp classes, kết quả x sẽ được dự đoán vào class có nhiều lượt vote nhất. Trường hợp có nhiều hơn một class có số vote lớn nhất bằng nhau, chọn class có index bé nhất.

- **DAGSVM:** ý tưởng giống với one-against-one, nhưng ở testing phase DAGSVM sẽ sử dụng rooted binary directed acyclic graph (Đồ thị không chu trình có hướng gốc nhị phân) bao gồm $\frac{k(k-1)}{2}$ nodes với mỗi node là một binary classifier.



Rooted binary directed acyclic graph

Video trình bày đồ án

https://youtu.be/V5y-aCGGLjQ

II Huấn luyện SVM để phân lớp ảnh thời trang

1 Thư viện sử dụng

- **numpy**: Hỗ trợ thực hiện các phép toán số học và xử lý mảng nhanh chóng trong quá trình làm việc với dữ liệu số và ma trận.

- **matplotlib.pyplot**: Tạo đồ thị và trực quan hóa dữ liệu, hiển thị kết quả của các mô hình.

- **gzip, os**: Hỗ trợ đọc dữ liệu từ các tập tin.

- **time**: Cung cấp hàm để đo thời gian chạy của đoạn mã.

- **itertools.product**: Tạo ra các tổ hợp của các phần tử, được sử dụng để tạo các bộ tham số để thử nghiệm trong quá trình điều chỉnh mô hình.

- **sklearn.model_selection.train_test_split**: Được sử dụng để chia tập dữ liệu thành tập huấn luyện và tập kiểm tra, giúp đánh giá hiệu suất của mô hình máy học.

- **sklearn.svm.SVC**: Thực hiện mô hình Support Vector Machine (SVM) trong thư viện scikit-learn.

- **sklearn.metrics**: Cung cấp các hàm để đánh giá mô hình như tính độ chính xác, ma trận nhầm lẫn.

2 Chuẩn bị dữ liệu

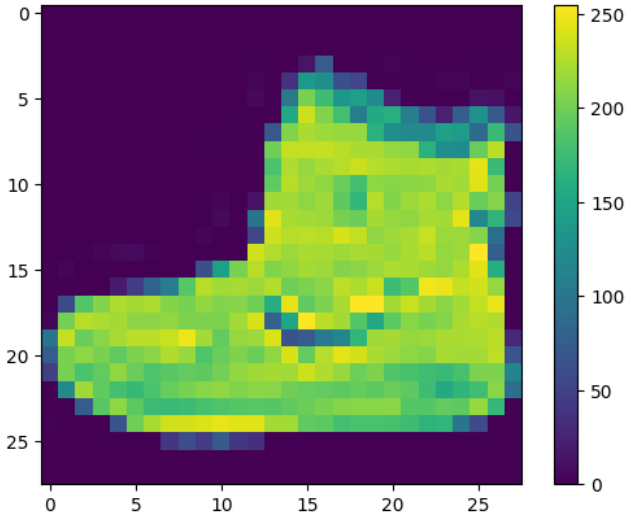
- Bộ dữ liệu được sử dụng là bộ Fashion MNIST. (Tải dữ liệu)

- Bộ dữ liệu bao gồm 60000 mẫu huấn luyện và 10000 mẫu thử nghiệm. Mỗi mẫu là ảnh có kích thước 28x28 tương ứng với một trong các nhãn sau:

Label	Description
0	T-shirt/top
1	Trouser
2	Pullover
3	Dress
4	Coat
5	Sandal
6	Shirt
7	Sneaker
8	Bag
9	Ankle boot

- Dữ liệu sẽ được đọc từ 4 file sau:

- train-images-idx3-ubyte.gz: training set images, bao gồm 60,000 mẫu
- train-labels-idx1-ubyte.gz: training set labels, bao gồm 60,000 mẫu
- t10k-images-idx3-ubyte.gz: test set images, bao gồm 10,000 mẫu
- t10k-labels-idx1-ubyte.gz: test set labels, bao gồm 10,000 mẫu



Mẫu đầu tiên trong tập train sau khi đọc dữ liệu từ file

- Có thể thấy trong hình, các giá trị pixel nằm trong phạm vi từ 0 đến 255. Ta sẽ chuẩn hoá các mẫu ảnh trong tập train và tập test trong phạm vi từ 0 đến 1 bằng cách chia các giá trị cho 255 và sử dụng hàm *reshape* để chuyển đổi định dạng của hình ảnh từ mảng hai chiều (28, 28) thành mảng một chiều (784).

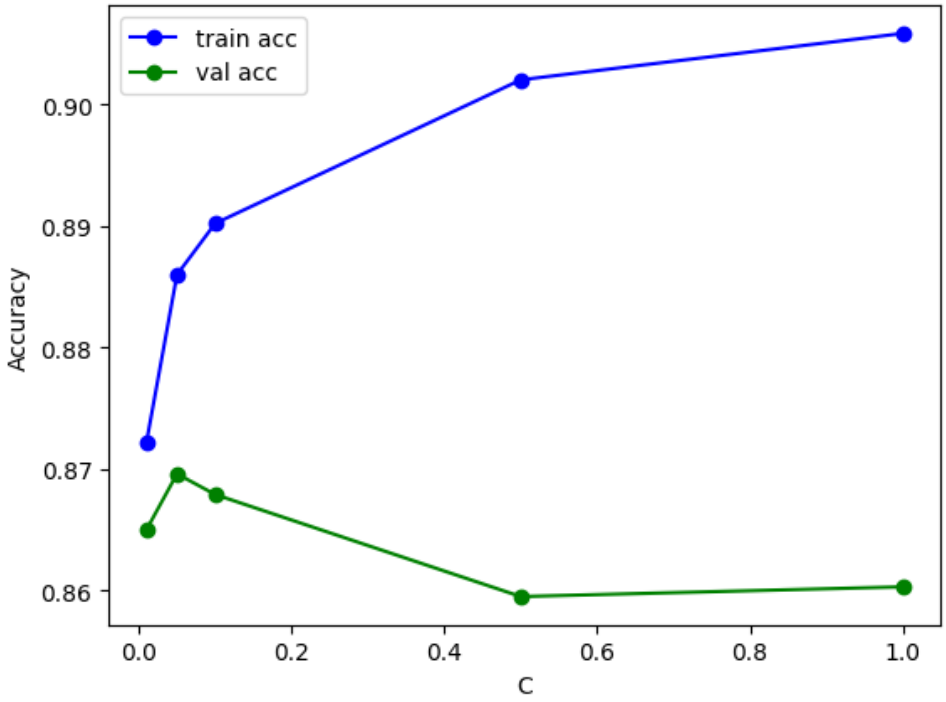
Kiểm tra 16 mẫu đầu tiên trong tập train sau khi chuẩn hoá dữ liệu

3 Huấn luyện SVM

Linear kernel

- Mô hình sẽ chạy lần lượt với 5 giá trị khác nhau của C: 0.01, 0.05, 0.1, 0.5, 1

C	Thời gian (giây)	Train accuracy	Validation accuracy
0.01	239.27	0.8722	0.865
0.05	155.08	0.886	0.8696
0.1	246.02	0.89022	0.8679
0.5	267.72	0.90204	0.8595
1	206.10	0.90586	0.8603



Linear kernel

- **Nhận xét:** Giá trị C = 0.05 cho kết quả tốt nhất với validation accuracy cao nhất (0.8696). Bên cạnh đó ta có thể thấy với các giá trị C > 0.05, tuy train accuracy tăng nhưng validation accuracy lại giảm, đây là dấu hiệu cho thấy mô hình đang bị overfitting.

RBF kernel

- Mô hình sẽ chạy lần lượt với các giá trị C = [0.01, 0.05, 0.1, 0.5, 1] và γ = [0.001, 0.005, 0.01, 0.05, 0.1]

C	Thời gian (giây)	Train accuracy	Validation accuracy
0.01	1414.80	0.70786	0.708
0.05	649.31	0.76846	0.767
0.1	546.16	0.79092	0.7887
0.5	475.65	0.83604	0.8347
1	345.09	0.85282	0.8524

$\gamma = 0.001$

C	Thời gian (giây)	Train accuracy	Validation accuracy
0.01	732.17	0.76298	0.7608
0.05	471.92	0.81674	0.8166
0.1	512.92	0.83736	0.8349
0.5	349.46	0.87756	0.8725
1	197.92	0.89022	0.8809

$\gamma = 0.005$

C	Thời gian (giây)	Train accuracy	Validation accuracy
0.01	660.51	0.7802	0.779
0.05	366.50	0.83322	0.831
0.1	412.73	0.85394	0.8546
0.5	377.28	0.89396	0.8844
1	291.16	0.91042	0.8918

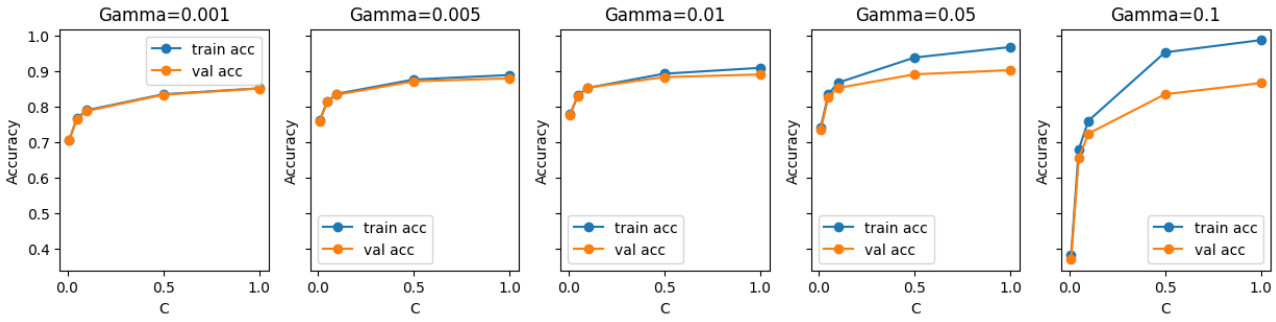
$\gamma = 0.01$

C	Thời gian (giây)	Train accuracy	Validation accuracy
0.01	1263.53	0.74172	0.7351
0.05	982.77	0.8357	0.828
0.1	633.57	0.86862	0.8537
0.5	658.31	0.93954	0.8922
1	631.05	0.9691	0.9041

$\gamma = 0.05$

C	Thời gian (giây)	Train accuracy	Validation accuracy
0.01	1970.74	0.38196	0.371
0.05	1626.66	0.68094	0.6558
0.1	1331.05	0.76128	0.7251
0.5	1453.21	0.95406	0.8361
1	1389.64	0.98866	0.8677

$\gamma = 0.1$



RBF kernel

- **Nhận xét:** Giá trị γ càng lớn, mô hình càng phức tạp nên thời gian train mô hình càng lâu. Giá trị $\gamma = 0.1$ cho mô hình có dấu hiệu bị overfitting, đặc biệt là khi với C = 0.5 và C = 1 thì train accuracy rất cao (trên 95%) nhưng validation accuracy lại dưới 90%. Với giá trị $\gamma = 0.001$ và $\gamma = 0.005$, train accuracy và validation accuracy không chênh lệch nhiều, nhưng nhìn chung accuracy của hai giá trị γ này không được cao. Validation accuracy khá tốt với $\gamma = 0.01$ và $\gamma = 0.05$, validation accuracy cao nhất là 0.9041 với C = 1 và $\gamma = 0.05$.

4 Đánh giá SVM

- Dựa vào kết quả huấn luyện SVM, ta sẽ sử dụng mô hình có validation accuracy là RBF kernel với C = 1 và $\gamma - 0.05$ để chạy trên tập test.

Test Accuracy: 0.8912

	precision	recall	f1-score	support
0	0.84	0.85	0.85	1000
1	1.00	0.96	0.98	1000
2	0.81	0.82	0.81	1000
3	0.88	0.90	0.89	1000
4	0.83	0.83	0.83	1000
5	0.96	0.97	0.97	1000
6	0.73	0.69	0.71	1000
7	0.95	0.96	0.95	1000
8	0.94	0.97	0.96	1000
9	0.97	0.95	0.96	1000
accuracy			0.89	10000
macro avg	0.89	0.89	0.89	10000
weighted avg	0.89	0.89	0.89	10000

Test accuracy

- Mô hình cho ra kết quả test accuracy là 0.8912, chênh lệch không quá nhiều với train accuracy và validation accuracy.

zalando research | Benchmark | Fashion MNIST | Original MNIST | Side-by-Side | Repository

Filter 129 results by Name, Parameter

Name	Parameter	Accuracy (mean)	Accuracy (std)	Training time	Repeats	Job start	Job Done
SVC	{"C":10,"kernel":"poly"}	0.897	0.000	1:12:39	2	6 years ago	6 years ago
SVC	{"C":100,"kernel":"poly"}	0.896	0.000	1:12:30	2	6 years ago	6 years ago
SVC	{"C":10,"kernel":"rbf"}	0.896	0.000	1:15:25	2	6 years ago	6 years ago
SVC	{"C":100,"kernel":"rbf"}	0.893	0.000	1:18:01	2	6 years ago	6 years ago
GradientBoostingClassifier	{"loss":"deviance","max_depth":10,"n_estimators":100}	0.888	0.001	17:38:22	5	6 years ago	6 years ago
SVC	{"C":1,"kernel":"rbf"}	0.884	0.000	1:15:54	2	6 years ago	6 years ago
GradientBoostingClassifier	{"loss":"deviance","max_depth":10,"n_estimators":50}	0.880	0.001	16:42:47	2	6 years ago	6 years ago
RandomForestClassifier	{"criterion":"entropy","max_depth":50,"n_estimators":100}	0.879	0.000	0:08:39	2	6 years ago	6 years ago

Một số kết quả bằng phương pháp khác được ghi nhận trên bộ dữ liệu Fashion MNIST

- Các mô hình kernel SVM có accuracy cao hơn đều sử dụng giá trị C = 10 và C = 100. Với mô hình cùng có giá trị C = 1 thì mô hình với $\gamma = 0.05$ có accuracy cao hơn accuracy được ghi nhận.

- Nhìn chung, mô hình cho ra kết quả khá tốt với accuracy chỉ thấp hơn 4 trong tổng số kết quả được ghi nhận trên bộ dữ liệu Fashion-MNIST.

III Tài liệu tham khảo

1. Convex optimization and Duality for convex optimization problems
   (a) Convexity and The Principle of Duality
   (b) The Karush–Kuhn–Tucker (KKT) Conditions and the Interior Point Method for Convex Optimization
   (c) Lagrange.pdf
2. Lý thuyết mô hình SVM
   (a) Support Vector Machine
   (b) Soft Margin Support Vector Machine
   (c) Kernel Support Vector Machine
   (d) RBF Kernel
   (e) Multiclass Kernel