Phân lớp ảnh chữ số viết tay bằng SVM

Đồ án vấn đáp môn Máy học

Hồng Thanh Hoài
Khoa Công nghệ thông tin
Đại học Khoa học Tự nhiên
Thành phố Hồ Chí Minh, Việt Nam
hthoai1006@gmail.com

Huỳnh Minh Huấn
Khoa Công nghệ thông tin
Đại học Khoa học Tự nhiên
Thành phố Hồ Chí Minh, Việt Nam
minhhuanhuynh289@gmail.com

***Tóm tắt nội dung*—Support Vector Machine (SVM) là một phương pháp học có giám sát trong các mô hình nhận dạng mẫu với cơ sở toán học chặt chẽ. Nó không chỉ hoạt động tốt với các dữ liệu được phân tách tuyến tính mà còn tốt với cả dữ liệu phân tách phi tuyến. Với nhiều bài toán, SVM mang lại kết quả tốt như mạng neuron với hiệu quả sử dụng tài nguyên tốt hơn hẳn. Ở đồ án này, nhóm tiến hành thực nghiệm SVM bằng thư viện scikit-learn trên bộ dữ liệu MNIST (phân lớp ảnh chữ số viết tay).**

I. Quá trình thực hiện

- *10/06/2019 - 15/06/2019*:
  Tìm hiểu và nắm vững kiến thức lý thuyết.
- *20/06/2019 - 25/06/2019*:
  Tìm hiểu thư viện scikit-learn và tiến hành train để chọn được các tham số tối ưu nhất với mỗi mô hình.
- *26/06/2019 - 28/06/2019*:
  Tổng hợp kết quả và viết báo cáo.
- *29/06/2019*:
  Ôn lại phần lý thuyết chuẩn bị cho vấn đáp.

GVHD: ThS. Trần Trung Kiên

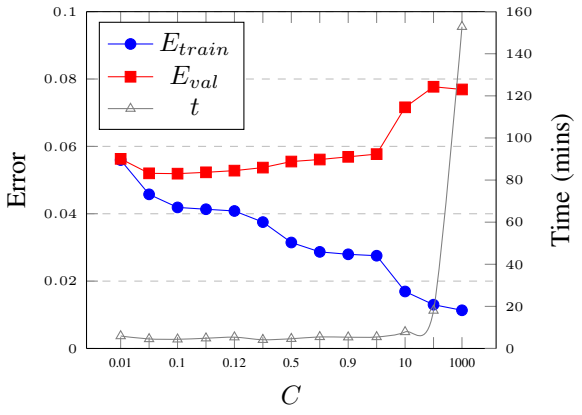
II. Huấn luyện SVM

A. Dùng linear kernel

Bảng I
Bảng độ lỗi trên tập *training* và *validation* khi thay đổi C.

C	E_train	E_val	Time
0.01	0.05594	0.0563	5min 55s
0.05	0.04576	0.0520	4min 31s
0.1	*0.04188*	*0.0519*	*4min 22s*
0.11	0.04134	0.0523	4min 52s
0.12	0.04082	0.0528	5min 22s
0.2	0.03752	0.0537	4min 9s
0.5	0.03146	0.0555	4min 37s
0.8	0.02868	0.0561	5min 30s
0.9	0.02796	0.0569	5min 21s
1	0.02754	0.0577	5min 28s
10	0.01692	0.0716	7min 51s
100	0.01300	0.0777	18min 2s
1000	0.01134	0.0769	2h 32min

Độ lỗi trên tập *training* và *validation* khi thay đổi C.



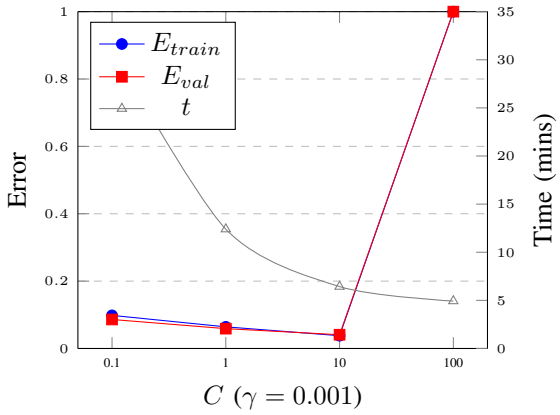
Nhận xét: Theo lý thuyết, C lớn sẽ dẫn đến trường hợp "lower bias, higher variance", tức *overfitting*. C nhỏ sẽ dẫn đến trường hợp "higher bias, lower variance", tức *underfitting*. Thật vậy, thực nghiệm cho thấy khi C càng lớn thì E_{train} càng giảm, nhưng E_{val} lại tăng (overfitting). C quá nhỏ thì E_{val} có độ lỗi còn lớn (underfitting).

B. Dùng RBF kernel

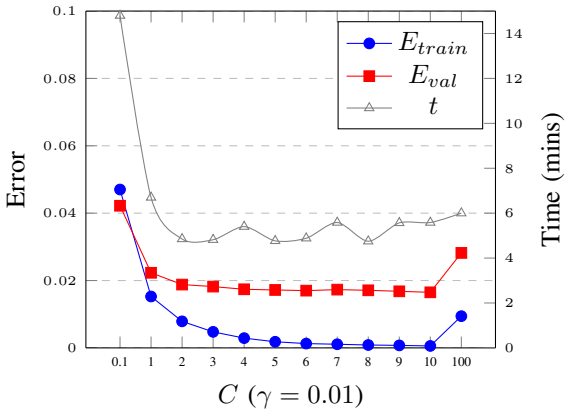
Bảng II
BẢNG ĐỘ LỖI TRÊN TẬP *training* VÀ *validation* KHI THAY ĐỔI C VÀ γ.

C	γ	E_tran	E_val	Time
0.1	0.001	0.0982	0.0861	31m12s
	0.01	0.0470	0.0422	14m48s
	0.1	0.2895	0.3125	1h14m
1	0.001	0.0642	0.0589	12m24s
	0.01	0.0152	0.0223	6m42s
	0.1	4E-05	0.0448	1h53m
2	0.01	0.0078	0.0188	4m51s
3	0.01	0.0047	0.0182	4m49s
4	0.01	0.0029	0.0174	5m24s
5	0.01	0.0018	0.0172	4m46s
6	0.01	0.0012	0.0170	4m53s
7	0.01	0.0010	0.0173	5m35s
8	0.01	0.0008	0.0171	4m45s
9	0.01	0.0007	0.0168	5m34s
10	0.001	0.0379	0.0408	6m26s
	0.01	0.0005	0.0165	5m35s
	0.1	0	0.0434	1h59m
100	0.001	1	1	4m56s
	0.01	0.0094	0.0282	
	0.1	0	0.0168	

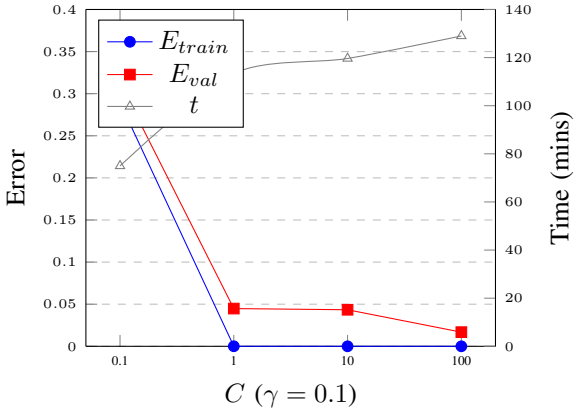
(1) Độ lỗi trên tập *training* và *validation* khi thay đổi C.



(2) Độ lỗi trên tập *training* và *validation* khi thay đổi C.



(3) Độ lỗi trên tập *training* và *validation* khi thay đổi C.



Nhận xét: Cũng tương tự như tham số C, γ lớn sẽ dẫn đến trường hợp "lower bias, higher variance", tức *overfitting*. γ nhỏ sẽ dẫn đến trường hợp "higher bias, lower variance", tức *underfitting*. Thật vậy, thực nghiệm cho thấy khi γ nhỏ (biểu đồ 1) thì E_{train} và E_{val} đều rất lớn. Khi γ lớn thì lại dẫn đến *overfitting*, E_{train} có trường hợp bằng 0 nhưng E_{val} lại lớn.

$\rightarrow$ Vậy, ta chọn hàm dự đoán cuối cùng với E_{val} nhỏ nhất là khi dùng RBF kernel, với $C = 10$, $\gamma = 0.01$, cho $E_{val} = 0.0165$ ($Accuracy = 0.9835$).

III. Đánh giá SVM

Với RBF kernel, $C = 10$, $\gamma = 0.01$, ta có kết quả sau:

Training Score	0.99942
Training Error	0.00058
Testing Score	0.982
Testing Error	0.018
Time	4min 47s

IV. Trả lời câu hỏi vấn đáp

- **Kernel**:
  - Ta sử dụng một phép biến đổi $\phi(x)$ sao cho dữ liệu ban đầu là không phân biệt tuyến tính được biến đổi sang không gian mới, nâng được chiều của dữ liệu ban đầu. Ở không gian mới này, dữ liệu trở nên khả tách tuyến tính hoặc gần khả tách tuyến tính.
  - Kernel là hàm $K(x, z) = \phi(x)^T \phi(z)$ để tính tích vô hướng giữa các điểm dữ liệu trong không gian mới.
  - Input của kernel là thể hiện (điểm) dữ liệu,x , z. Output là tích vô hướng của phép biến đổi phi tuyển.
- **Gamma**:
  γ là nghịch đảo độ lệnh chuẩn, dùng để đo độ tương quan giữa 2 điểm.
  - γ lớn thì phương sai nhỏ (bán kính Gaussian nhỏ): 2 điểm được coi là tương đồng (cùng lớp) chỉ khi chúng gần nhau.
  - γ nhỏ thì phương sai lớn (bán kính Gaussian lớn): 2 điểm vẫn có thể tương đồng với nhau dù ở xa.

Lời cảm ơn

Nhóm chúng em xin chân thành cảm ơn thầy Trần Trung Kiên đã giảng giải rất kỹ lưỡng để chúng em có thể hiểu rõ và hoàn thành được đồ án này.

Tài liệu

[1] Documentation of scikit-learn 0.21.2.
[2] Support Vector Machine, *Machine Learning cơ bản*.