Fullname: Nguyễn Ngọc Thiện

Student ID: 19110148

1. Có cần phải scale features trước khi huấn luyện các SVM models không? Tại sao?   
   *Gợi ý:* feature scaling có ảnh hưởng đến margins không?

* Cần phải scale features trước khi huấn luyện để giảm bớt các điểm bất thường vì nếu không làm vậy margins sẽ thu hẹp và thậm chí là không có đường decision boundary nào phân chia được dữ liệu với SVC và mô tả dữ liệu với SVR

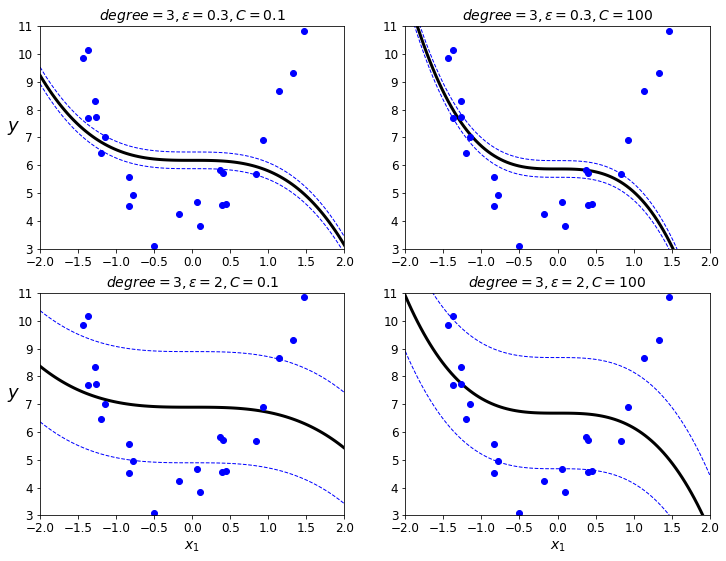
1. Primal và dual problem là gì? Tại sao cần giải dual problem mà không giải trực tiếp primal problem?

* Primal problem (bài toán gốc) là giải objective function tìm w, b
* Dual problem là bài toán có solution được cung cấp ở solution của primal problem
* Sử dụng dual problem vì đơn giản dễ tìm được solution, solution của 2 bài toán giống nhau.

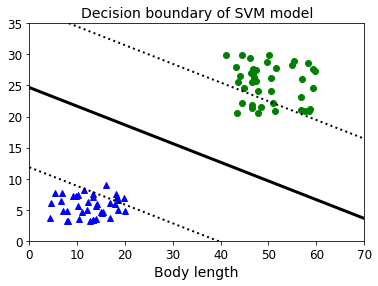
1. Khi một SVM classifier với Gaussian RBF kernel bị underfitting thì chúng ta nên tăng hay giảm tham số γ (gamma) và tham số C? Tại sao?

* Khi bị underfitting của SVM classifier với Gaussian RBF kernel khi nên tăng cả hai gamma và tham số C bởi vì khi tăng gamma để tăng độ cong của đường mô tả dữ liệu (lên bậc 3, 4, ...) và tăng C để cho các loại dữ liệu khác nhau càng nằm về 1 bên hơn -> model ít bị underfitting hơn. Tuy nhiên ta cần tăng sao cho phù hợp để tránh trường hợp ngược lại là overfitting

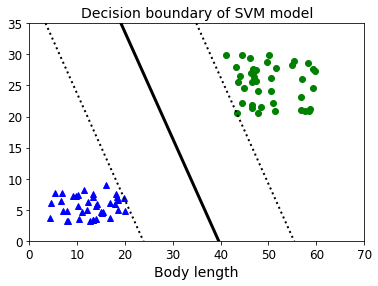
Phần Lập Trình

1. Thay đổi các phương trình tạo data giả lập, chọn các classes khác trong Iris dataset, và thực hiện chạy từng bước các thuật toán như demo trong bài học (làm tiếp theo tuần trước).
2. Huấn luyện các classifiers **LinearSVC, SVC, SGDClassifier** trên một dataset tự tạo (dataset này có thể được phân lớp bằng linear boundary) và xem y có thể tạo ra cùng một model không. Giải thích kết quả.

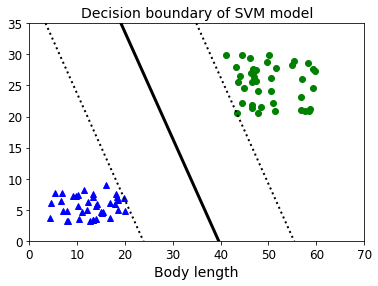
**LinearSVC**



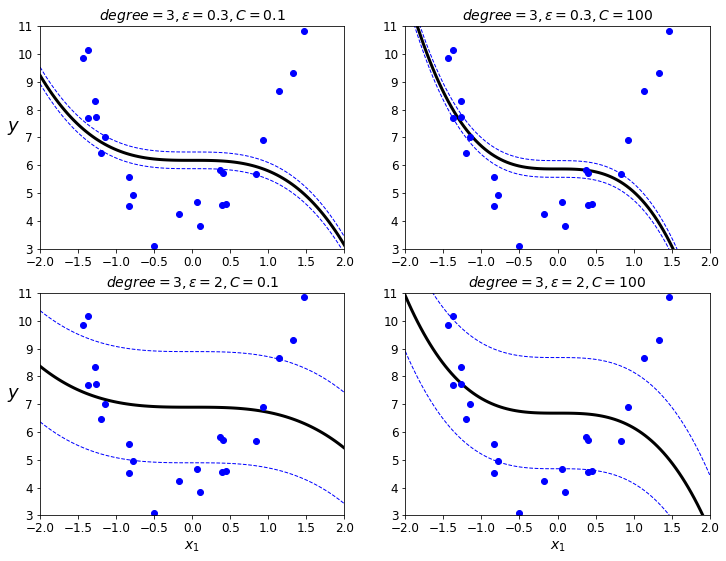
**SVC**



**SGDClassifier**

****

1. Train **polynomial SVM** regression trong mục 6.5 sử dụng hyperparameters **degree>2** và các giá trị khác nhau cho **C** và **epsilon**. Giải thích kết quả thu được.



1. Train một SVM regressor với data trong mục 6.5 sử dụng **Gaussian RBF kernel** với các giá trị khác nhau cho **C** và **epsilon**. Giải thích kết quả thu được.
2. m = 30
3. X1 = 4\*np.random.rand(m, 1) -2
4. y1 = (4 + 3\*X\*\*2 + X + np.random.randn(m, 1)).ravel()
5. *# Train 1 Gaussian SVM using Kernel trick*
6. Gaus\_kernel\_svm\_clf = Pipeline([
7. ("scaler", StandardScaler()),
8. ("svm\_clf", SVR(*kernel*="rbf", *epsilon*=0.6, *C*=0.001))  ])
9. Gaus\_kernel\_svm\_clf.fit(X1, y1)
10. Gaus\_kernel\_svm\_clf.predict(X1)
11. *# Train several Gaussian SVMs using Kernel trick*
12. epsilon1, epsilon2 = 0.2, 1, 3
13. C1, C2 = 0.01, 100, 10000
14. hyperparams = (epsilon1, C1), (epsilon1, C2), (epsilon2, C1), (epsilon2, C2)
15. svm\_clfs = []
16. *for* epsilon, C *in* hyperparams:
17. Gaus\_kernel\_svm\_clf = Pipeline([
18. ("scaler", StandardScaler()),
19. ("", SVR(*kernel*="rbf", *epsilon*=epsilon, *C*=C)) ])
20. Gaus\_kernel\_svm\_clf.fit(X1, y1)
21. svm\_clfs.append(Gaus\_kernel\_svm\_clf)
22. *[Làm bài này có điểm thưởng]* Train SVM classifiers trên **MNIST dataset** (dùng one-versus-all). Thực hiện tuning các hyperparameters để đạt accuracy cao nhất có thể.
23. from tensorflow *import* keras
24. (X\_train, y\_train), (X\_test, y\_test) = keras.datasets.fashion\_mnist.load\_data()
25. *# 1.2. Reshape to 2D array: each row has 784 (28X28 pixel) features*
26. X\_train = X\_train.reshape(60000,784)
27. X\_test = X\_test.reshape(10000,784)
28. svc\_clf = SVC(*kernel*="linear", *C*=10000)
29. svr\_clf = SVR(*kernel*="poly", *degree*=2, *epsilon*=0.7, *C*=10000)
30. from sklearn.multiclass *import* OneVsRestClassifier
31. ova\_clf = OneVsRestClassifier(SVC(*kernel*="linear", *C*=10000, *random\_state*=42))
32. from sklearn.model\_selection *import* GridSearchCV
33. from sklearn.metrics *import* accuracy\_score
34. param\_grid = {'kernel':('linear', 'rbf'), 'C':[0.01, 1, 10, 100], 'gamma': 'scale'}
35. grid\_search = GridSearchCV(ova\_clf, param\_grid, *cv*=5, *verbose*=3, *n\_jobs*=-1)
36. grid\_search.fit(X\_train[:1000], y\_train[:1000])
37. print(grid\_search.best\_params\_)
38. print(grid\_search.best\_score\_)
39. print(grid\_search.cv\_results\_)
40. y\_pred = grid\_search.predict(X\_test)
41. print(accuracy\_score(y\_test, y\_pred))