1. **SVM multi-classification**

SVM은 margin criterion을 최대화하면서 binary-class를 두 개로 분류하는 기법이다. 이를 multi-class classification으로 확장하기 위한 방법으로 multi-class 분류 문제를 여러 개의 binary-class 문제로 분해하여 표준 SVM모델을 적용하는 방법이 있다. 여기에는 대표적으로 one-vs-rest(1VR)와 one-vs-one(1V1) 접근법이 있다.

1. One-vs-Rest(1VR) 기법

이 기법은 k-class를 분류하기위해 k개의 binary 분류기(classifier)를 구성한다. 각 클래스마다 하나의 classifer를 생성한다. m번째 binary classifier은 m번째 클래스를 양(positive)으로, 나머지 k-1개의 클래스를 음으로 train하여 분류한다. 예를 들어 다음과 같이 4개의 binary classifier를 생성하여 각 classifier가 하나의 색을 구분하도록 한다. 의 경우, 초록색 클래스를 양(+), 나머지는 음(-)으로 설정하여 train을 하게 된다. 하지만 훈련한 binary classifier 중 하나라도 잘못된다면 틀린 결과를 줄 수 있다는 단점이 있다.



1. One-vs-One(1V1) 기법

여기서는 하나의 class에 대해 하나의 classifier를 생성했던 앞선 방법과는 달리, 서로 다른 class의 짝의 수에 대해 하나의 classifier를 생성한다. 따라서 k(k-1)/2 (=) 개의 binary classifier를 구성하게 된다. 즉, k(k-1)/2개의 hyperplane을 찾는 것이다. 서로 다른 두개의 클래스를 비교하여 분류한다. 다음의 예에서는 6개(=) 의 binary classifier가 생성된다. 예를 들어 의 경우, 초록색과 노란색을 가진 클래스 (를 하나의 training set으로 설정하여 초록색은 양(+), 노란색은 음(-)으로 설정하여 train하게 된다. 어떤 변수 x가 주어졌을 때, 어떤 classifier 가 x가 클래스 에 있다는 것이 확인이 되면 에 대해 한 표를 투표하게 된다. 이렇게 투표를 한 후, 가장 많은 표를 얻은 클래스에 x가 분류된다.

앞선 1VR 방법보다 training set의 크기가 작기 때문에 classifier의 수가 많아도 전반적인 training time은 1VR에 뒤처지지 않는다. 하지만 test time의 경우 1V1에서는 classifier의 수가 많기 때문에 1VR보다 느리다. 하지만 그만큼 더 정확하다고 볼 수 있다.



*1VR vs 1V1*

SVM에서는 1V1가 더 나을 확률이 높다. Classifier의 개수는 이지만, 평균적으로 각 classifier가 더 작다.

1VR은 sklearn의 OneVsRestClassifier에, 1V1은 sklearn의 OneVsOneClassifier에 구현되어 있다. 다음과 같은 방법으로 1VR과 1V1의 성능을 비교할 수 있다.

Package Import:



Data Input:



Hyper Parameter Set-Up:



1VR:



1V1:



Result:



더 높은 쪽이 더 좋은 식별 성능을 보인다.

Ex) One-aginst-rest: 0.95333 / One-aginst-One: 0.79111

🡪 여기서는 1VR가 더 나은 성능을 보인다.

*다른 접근법*

1. Error Correcting Output Codes

각 클래스를 이진 형태로 표현한다. 아래의 행렬 중, 초록색 클래스의 경우를 보면, 이진수로 10101로 나타낼 수 있다. 여기서 서로 다른 클래스의 이진수는 최소 2비트 다르다. 노란색의 경우 00111로 표현할 수 있는데, 10101와 2개의 비트에서 차이가 있다(첫번째, 네번째 비트).



각 비트를 classifier로 구성하게 되면 다음과 같이 나타낼 수 있다.



어떤 변수 x가 주어졌을 때, classifier 를 통해 비트 배열을 얻어 비트가 가장 유사한 클래스로 분류한다.

이 방법은 sklearn의 OutputCodeClassifier에 구현되어 있다.



1. Single Machine

많은 binary classifier를 하나의 최적화 문제로 동시에 train하는 방법이다.

Multi classification

SVC and [NuSVC](http://scikit-learn.org/stable/modules/generated/sklearn.svm.NuSVC.html#sklearn.svm.NuSVC) 로 멀티 classification 하는 것은 일대일 대응으로 비교하는 것에서 시작한다. 예들 들면 n개의 클래스가 있으면 n\*(n-1)/2 개의 classifiers가 만들어지고 이것들은 각각을 일대일 대응으로 비교 한다. 일대일 대응이라는 것과 조합을 이해한다면 classifier의 개수가 이해 될 것이다. 밑에는 인터넷에서 찾은 decision\_function\_shape 옵션을 이용한 결과를 보여주는 코드이다.

>>>

**>>>** X = [[0], [1], [2], [3]]

**>>>** Y = [0, 1, 2, 3]

**>>>** clf = svm.SVC(decision\_function\_shape='ovo')

**>>>** clf.fit(X, Y)

SVC(C=1.0, cache\_size=200, class\_weight=None, coef0=0.0,

decision\_function\_shape='ovo', degree=3, gamma='auto', kernel='rbf',

max\_iter=-1, probability=False, random\_state=None, shrinking=True,

tol=0.001, verbose=False)

**>>>** dec = clf.decision\_function([[1]])

**>>>** dec.shape[1] *# 4 classes: 4\*3/2 = 6*

6

**>>>** clf.decision\_function\_shape = "ovr"

**>>>** dec = clf.decision\_function([[1]])

**>>>** dec.shape[1] *# 4 classes*

4

LInearSVC도 사용되는데 이것은 위에 있던 일대일대응 방법과는 달리 one-vs-the-rest 즉 클래스가 2개면 1개만 학습하는 형태이다.

>>>

**>>>** lin\_clf = svm.LinearSVC()

**>>>** lin\_clf.fit(X, Y)

LinearSVC(C=1.0, class\_weight=None, dual=True, fit\_intercept=True,

intercept\_scaling=1, loss='squared\_hinge', max\_iter=1000,

multi\_class='ovr', penalty='l2', random\_state=None, tol=0.0001,

verbose=0)

**>>>** dec = lin\_clf.decision\_function([[1]])

**>>>** dec.shape[1]

4

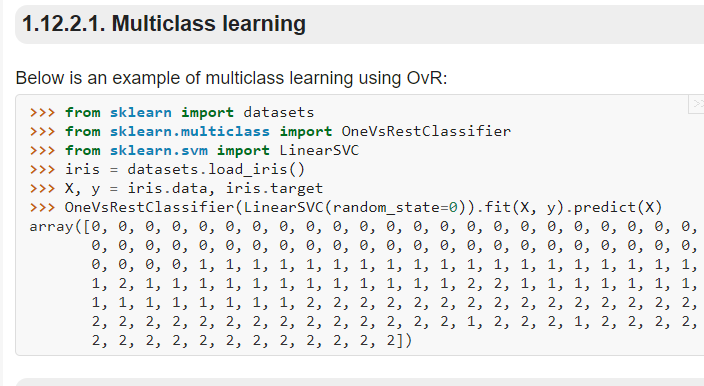
**(scikit-learn multiclass classification)**

<http://scikit-learn.org/stable/modules/classes.html#module-sklearn.multiclass>

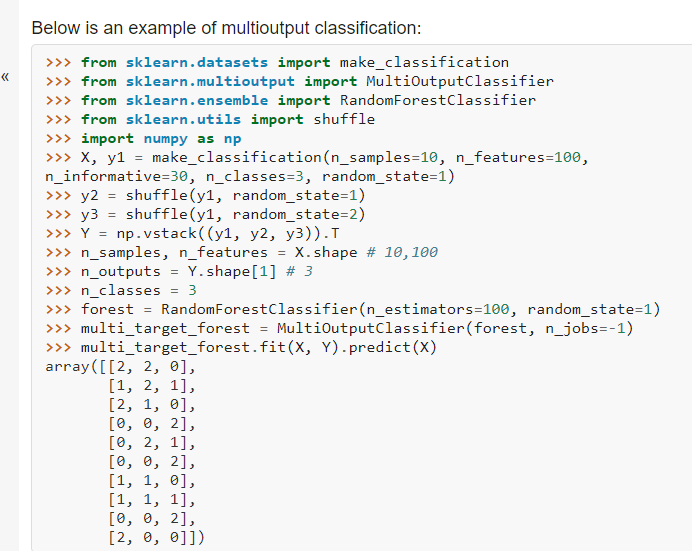
scikit-learn의 one vs the rest의 classifier 역시 기본적인 binart classifier을 기반으로 모인 unity이다(not sum이라는 것을 강조함)

그중 SVM중에서도 NuSVR이 가장 많이 사용되는 종류

one vs rest에 대한 기계학습에 대한 예시

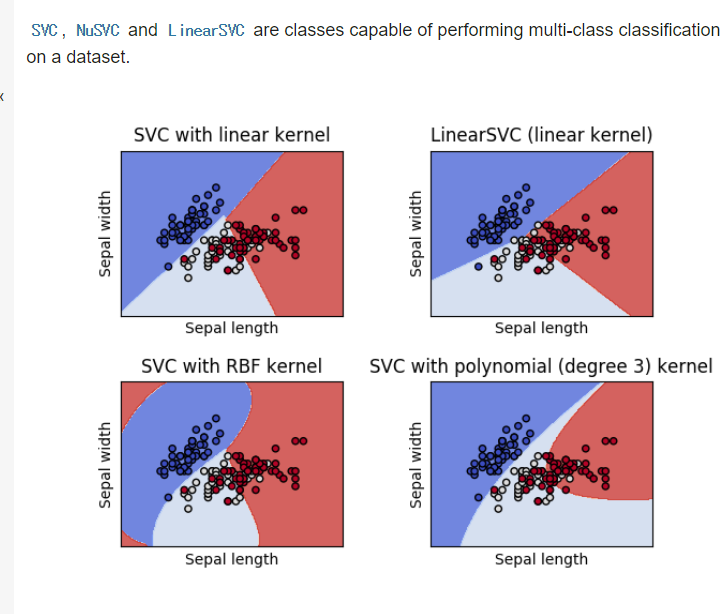


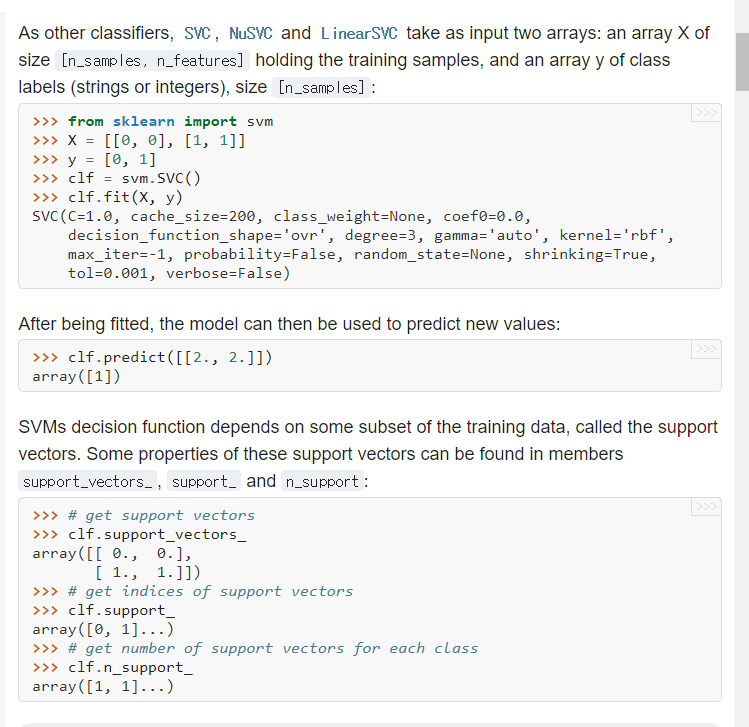
다중에 결과를 분류시켜 내보는 기계학습의 예시



SVM: a set of supervised learning methods used for classification, regression and outliers detection.

종류

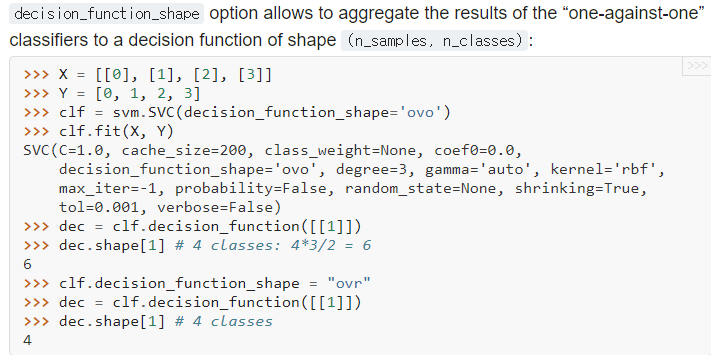




SVC를 사용하는 방법

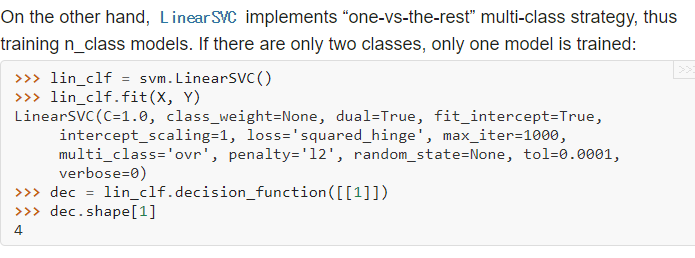
SVC와 NuSVC에서 one againset -one 방법으로 접근할 때의 모습입니다.

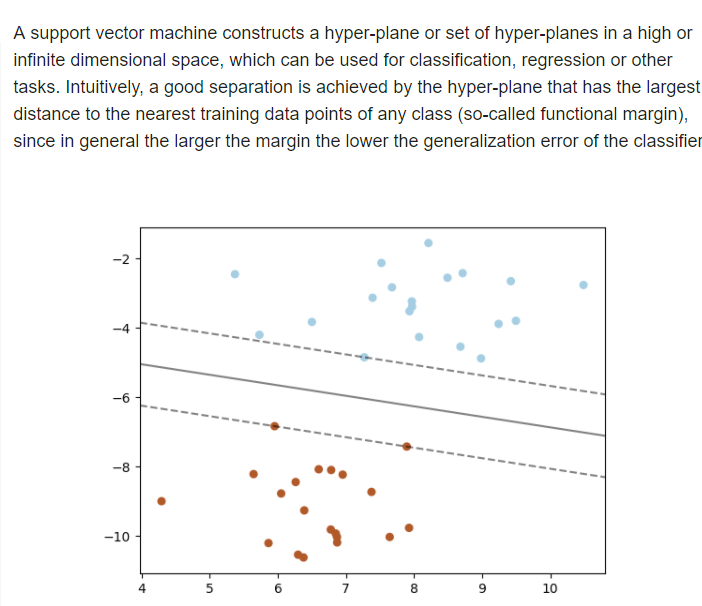
n\_class는 number of class, two classes에서 각자의 train data를 분류합니다.

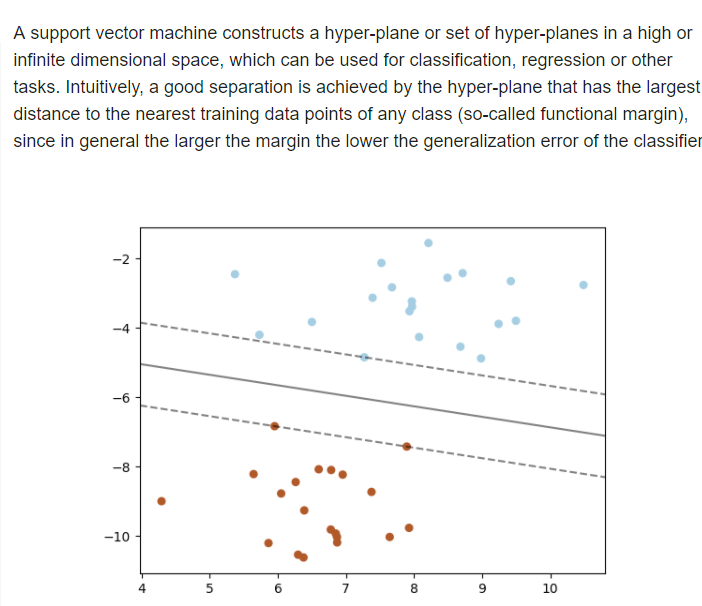


만일 "one-vs the -rest"의 방식을 구현할때는 linearSVC를 사용하는 것이 많다.

그러면 n\_class models들이 트레이닝됨







LinearSVC는 multiclass strategy에서 사용되기도 한다.