**Chapter 5. Support Vector Machines**

1. Linear SVM Classification

Soft Margin Classification

1. Nonlinear SVM Classification

Polynomial Kernel

Adding Similarity Features

Gaussian RBF Kernel

Computational Complexity

1. SVM Regression
2. Under the Hood

Decision Function and Predictions

Training Objective

Quadratic Programming

The Dual Problem

Kernelized SVM

Online SVMs

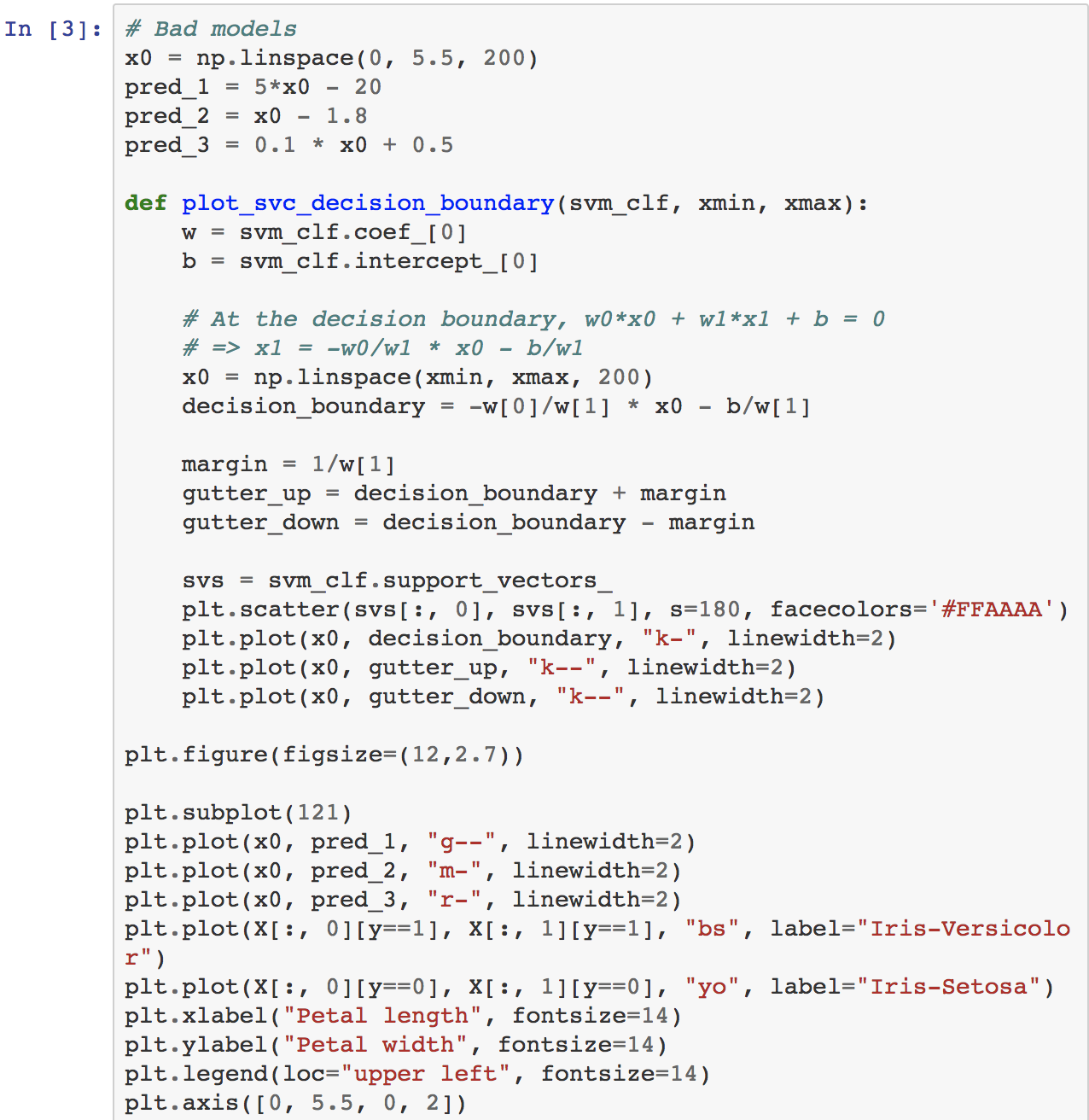
선형 또는 비선형 분류, 회귀 및 이상치 검출을 수행 할 수 있는 매우 강력하고 다양한 기계 학습 모델입니다.

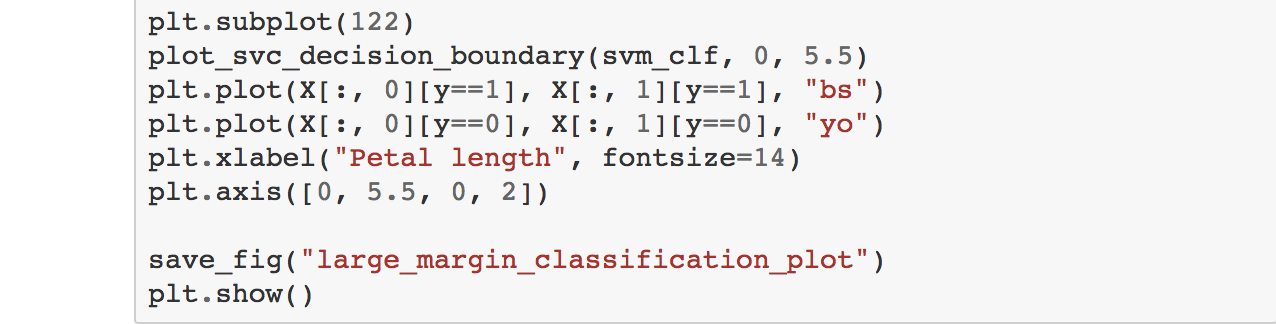
SVM은 특히 복잡하지만 중소 규모의 데이터 세트를 분류하는 데 적합합니다.

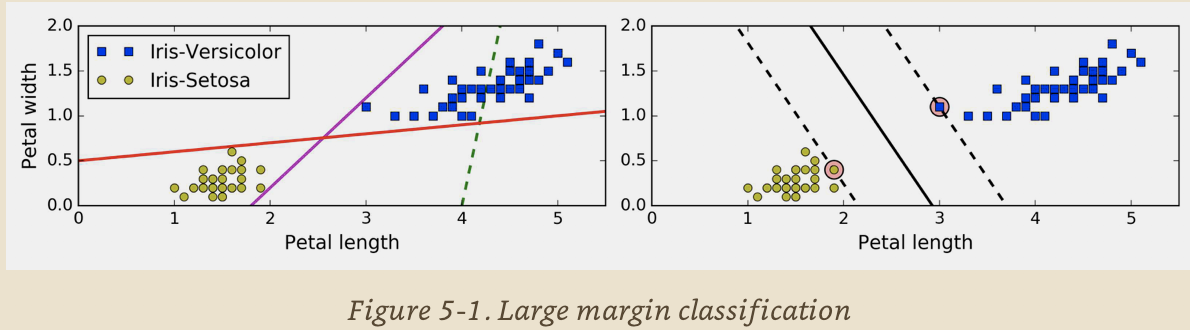
SVM의 핵심 개념, SVM을 사용하는 방법 및 SVM의 작동 방식에 대해 설명합니다.

**1. Linear SVM Classification**

4 장 끝에 소개 된 홍채 데이터 세트의 일부를 보여줍니다. 두 클래스는 직선으로 쉽게 분리 될 수 있습니다







왼쪽 그림은 세 가지 가능한 선형 분류기의 결정 경계를 보여줍니다.

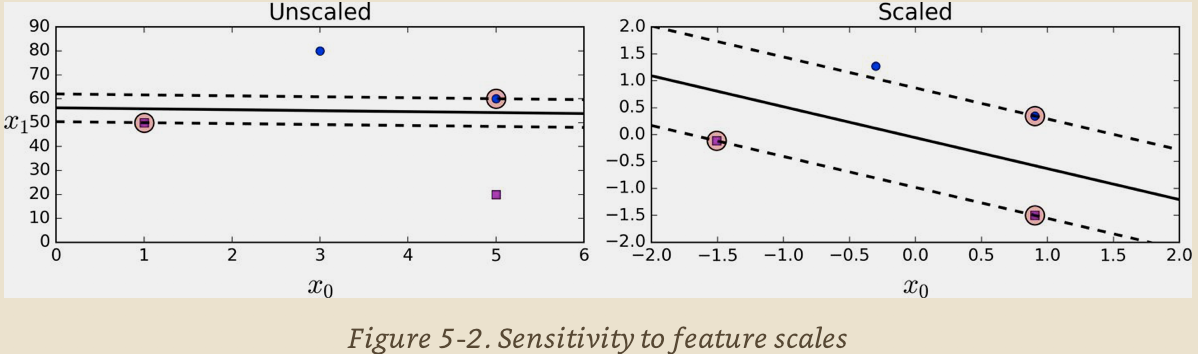
경계가 점선으로 표시되는 모델은 너무 나빠 클래스를 적절하게 분류하지 못합니다. 다른 두 모델은 이 교육 세트에서 완벽하게 작동하지만 인스턴스들과 매우 가깝게 있다. 그렇기 때문에 새로운 인스턴스들에 대해 제대로 작동하지 않을 수 있다.

대조적으로 오른쪽 그림의 실선은 SVM 분류기의 결정 경계를 나타냅니다. 가운데 실선은 두 클래스를 분리할 뿐만 아니라 가능한 한 가장 가까운 교육 인스턴스로부터 멀리 떨어져 있습니다. SVM 분류기는 클래스 사이에 평행 한 점선으로 표시된 선을 가능한 광범위하게 맞추는 것을 Large margin classification(대용량 분류)라고 합니다.

"거리 밖에서"더 많은 교육 인스턴스를 추가해도 결정 경계에는 전혀 영향을 미치지 않으며 거리 가장자리에 있는 인스턴스에 의해 완전히 결정 (또는 "지원")됩니다.

이러한 인스턴스를 Support 벡터 (그림 5-1에서 원으로 표시)라고 합니다.





SVM은 그림 5-2에서 볼 수 있듯이 feature scale에 민감합니다.

Feature 확장 후에는 결정 경계가 훨씬 더 좋아 보입니다

* 1. **Soft Margin Classification**

모든 인스턴스들을 매우 엄격하게 거리에 따라 나누는 것을 hard margin분류라고 합니다.

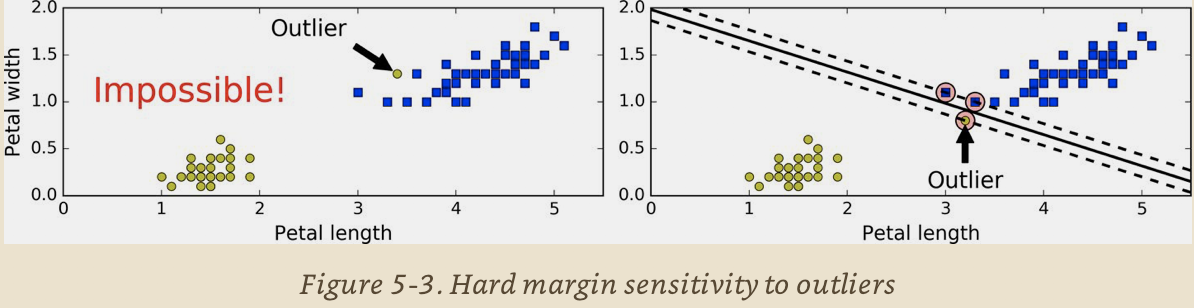
hard margin 분류에는 두 가지 주요 문제점이 있습니다.

첫째, 데이터가 선형으로 분리되어 있을 때 가능하다

두 번째, 이상치(outliers)에 민감하게 작동한다.

왼쪽에서 이상치로 인해 hard margin을 찾는 것이 불가능하며,

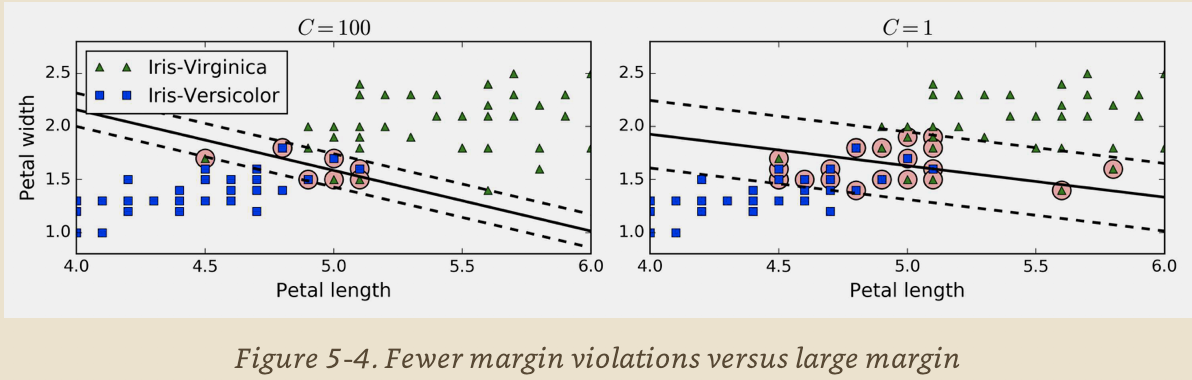
오른쪽에서는 그림 5-1에서 보았던 것과 결정 경계가 매우 다릅니다.



소프트 마진 분류는 이러한 문제를 피하기 위한 좀 더 유연한 모델.

목표는 가능한 거리를 길게 유지하는 것과 마진 위반(margin violation) (즉, 거리의 중간이나 잘못된 쪽에서 끝나는 경우)을 제한하는 것 사이의 균형을 찾는 것입니다.

Scikit-Learn의 SVM 클래스에서는 C 하이퍼 매개 변수를 사용하여 이 균형을 제어 할 수 있습니다. 하지만 C 값이 작을수록 거리가 넓지만 마진 위반은 더 커집니다



왼쪽에서 높은 C 값을 사용하면 분류 기준에서 마진 위반이 줄어들지만 마진은 줄어 듭니다.

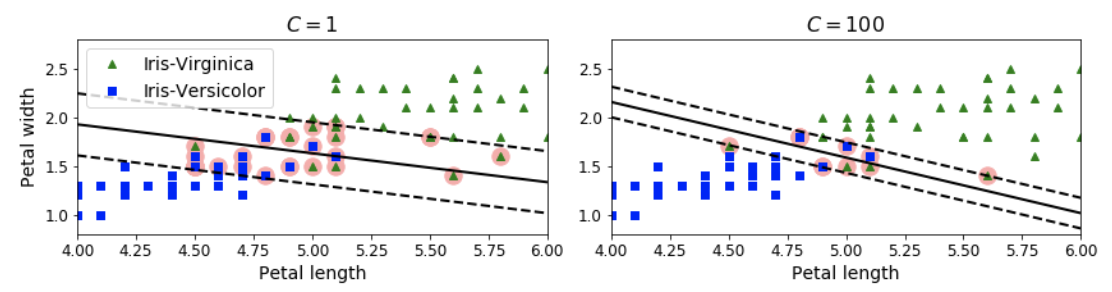
오른쪽에서 낮은 C 값을 사용하면 마진이 훨씬 커지지만 대부분의 경우 여백 안에 있습니다.

그러나 두 번째 분류기가 더 잘 일반화 될 가능성이 있습니다. 왜냐하면 이 트레이닝 셋에서도 마진 위반의 대부분이 실제로 결정 경계의 올바른쪽에 있기 때문에 예측 오류가 적습니다.

SVM 모델이 오버 피팅 (overfitting)되면 C를 줄임으로써 정규화 할 수 있습니다.

다음은 Scikit-Learn 코드는 홍채 데이터 세트를 로드하고, 기능을 스케일 한 다음, 직선 SVM 모델 (C = 1 및 힌지 loss 함수가있는 LinearSVC 클래스 사용)을 사용하여 Iris-Virginica 꽃을 감지합니다.





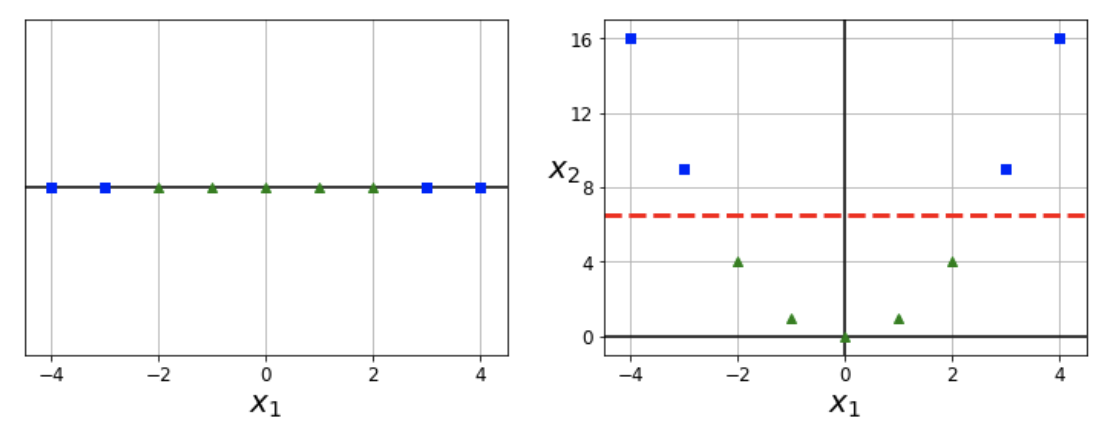
대안으로 SVC (kernel = "linear", C = 1)를 사용하여 SVC 클래스를 사용할 수도 있지만, 특히 대규모 교육 세트에서는 속도가 훨씬 느려서 권장하지 않습니다.

다른 대안은 SGDClassifier 클래스 (loss = "hinge", alpha = 1 / (m \* C))를 사용하는 것입니다. 선형 SVM 분류자를 훈련시키기 위해  Stochastic Gradient Descent이 적용됩니다 LinearSVC 클래스만큼 빠르지는 않지만 메모리에 맞지 않는 거대한 데이터 세트를 처리하거나 온라인 분류 작업을 처리하는 것이 유용 할 수 있습니다.

1. **Nonlinear SVM Classification**

선형 SVM 분류기는 효율적이고 많은 경우에 놀랍도록 잘 작동하지만 많은 데이터 세트는 선형으로 분리 될 수는 없습니다.

비선형 데이터 집합을 처리하는 한 가지 방법은 다항식 피쳐(polynomiar feature)를 더 추가하는 것입니다. 경우에 따라 선형으로 분리 가능한 데이터 셋이 생성 될 수 있습니다.

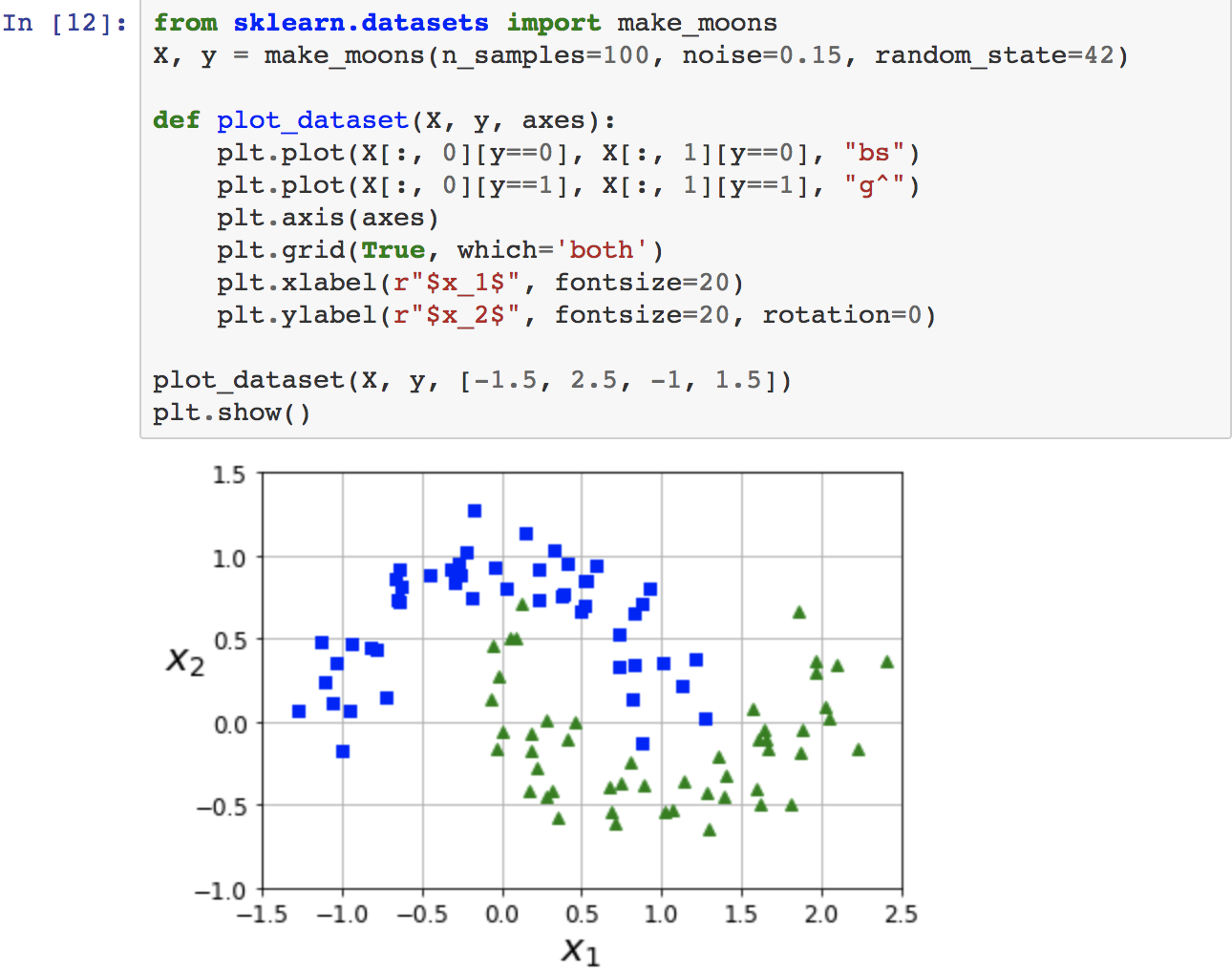


왼쪽 그림은 하나의 피쳐 x1을 가진 간단한 데이터 세트이고 선형으로 분리되지 않습니다.

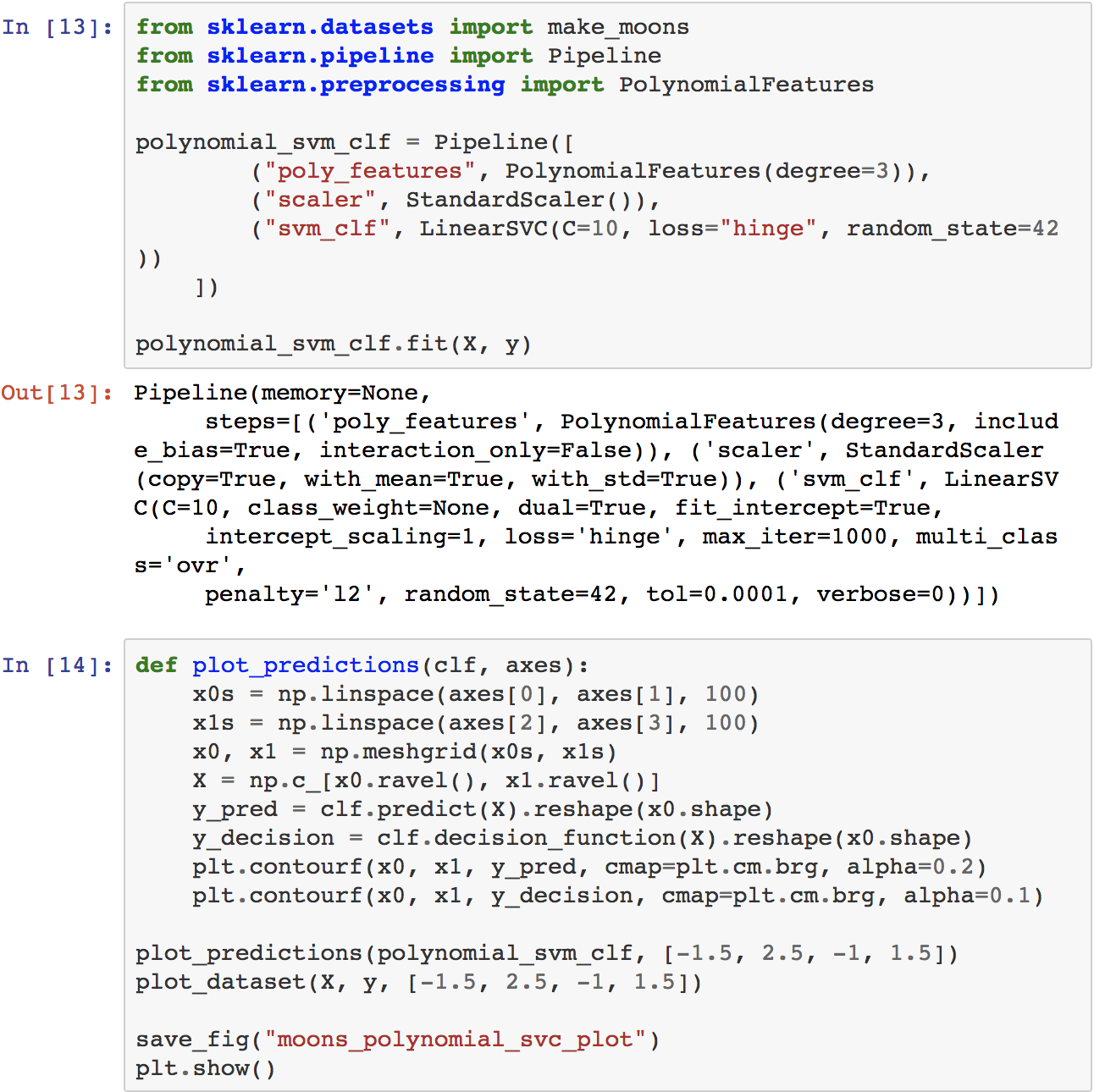
오른쪽 그림은 피쳐 x2 = (x1의 제곱) 을 추가하여 결과 2D 데이터 세트가 완벽하게 선형으로 분리됩니다.

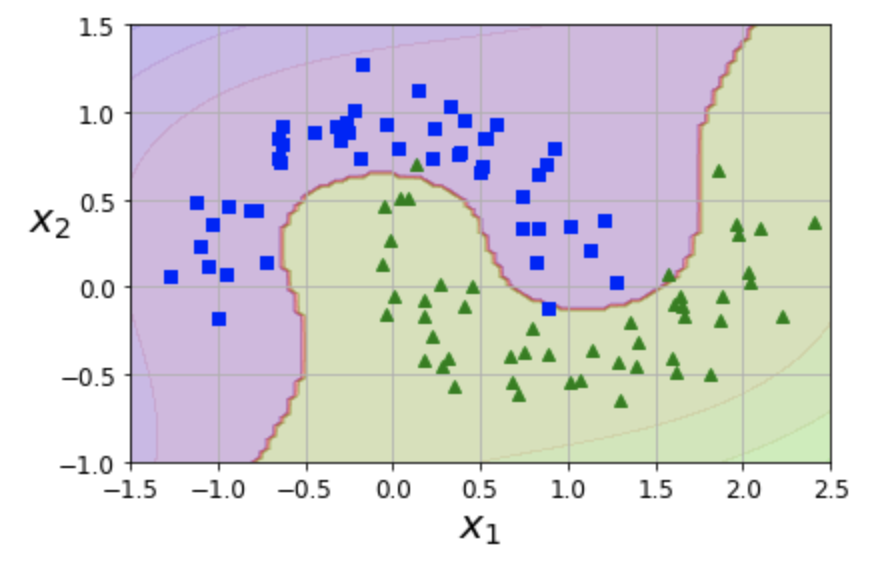
Scikit-Learn를 사용하여 구현하려면 PolynomialFeatures transformer를 포함하는 파이프 라인을 생성 한 다음 StandardScaler 및 LinearSVC를 진행 합니다.

달 데이터 세트에서 이것을 테스트 해 봅시다



샘플 수 100개, noise 0.15의 데이터 셋





파이프라인 생성 후 linear\_svc적용

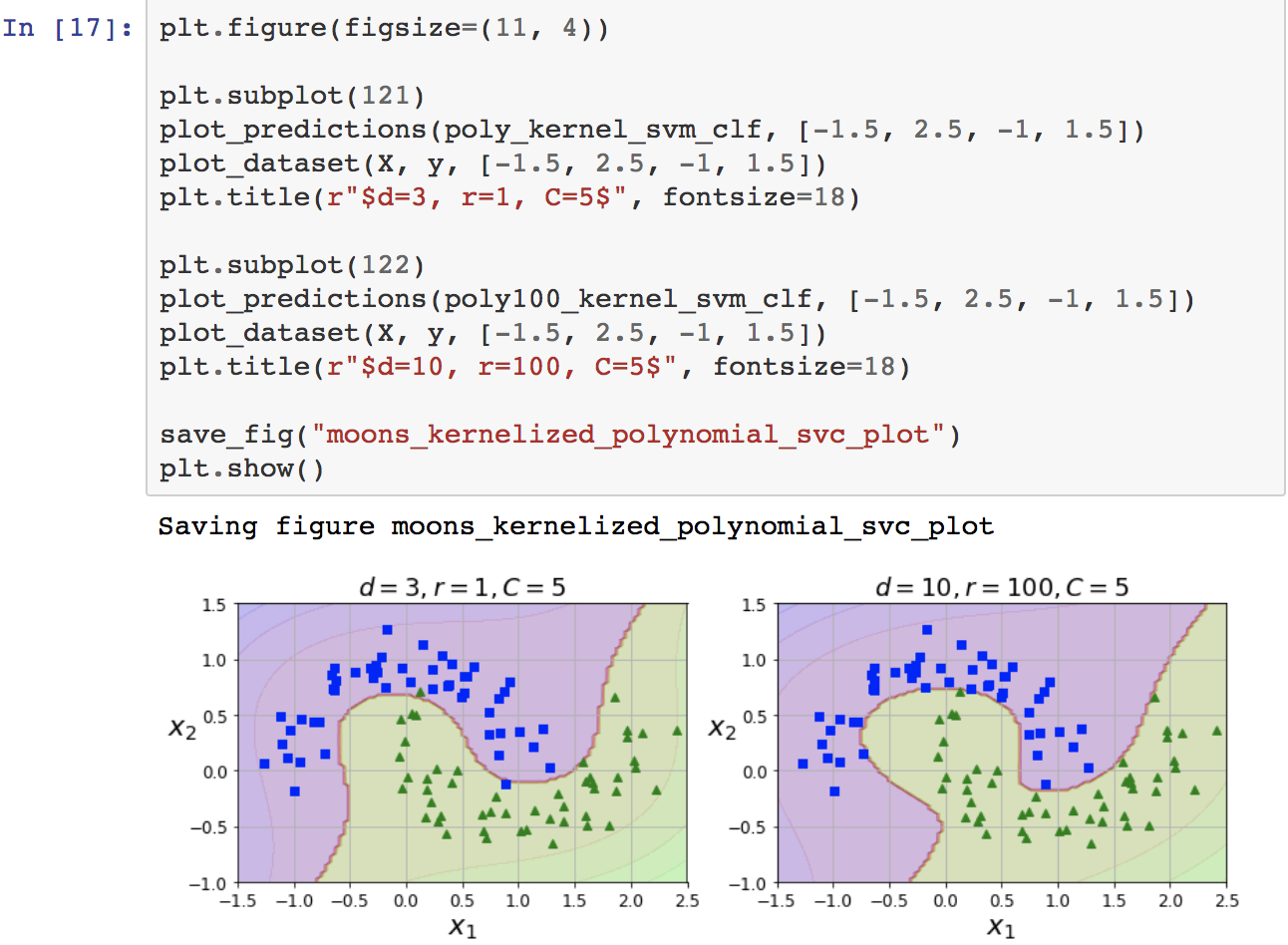
**2.1) Polynomial Kernel**

다항식 feature를 추가하는 것은 간단하며 SVM뿐만 아니라 모든 종류의 기계 학습 알고리즘에서도 훌륭하게 작동 할 수 있습니다.

그러나 다항식 차수가 낮으면 매우 복잡한 데이터 세트를 처리 할 수 ​​없으며, 다항식 차수가 높으면 많은 양의 feature가 생성되어 모델을 너무 느리게 만듭니다.

다행히 SVM을 사용할 때 커널 트릭이라고 불리는 수학 기법을 적용 할 수 있습니다

이 트릭은 SVC 클래스에 의해 구현됩니다.



달 데이터 세트에서 테스트 해 봅시다.

왼쪽에서 이 코드는 3차 다항 커널을 사용하여 SVM 분류기를 훈련시킵니다

오른쪽에는 10차 다항식 커널을 사용하는 다른 SVM 분류기가 있습니다

모델이 오버 피팅이라면 분명히 다항식 차수를 줄일 수 있습니다.

반대로 underfitting 경우 차수를 늘릴 수 있습니다.

하이퍼파라미터 인 coef0는 모델이 고차다항식과 저차다항식의 영향받는 정도를 제어합니다.

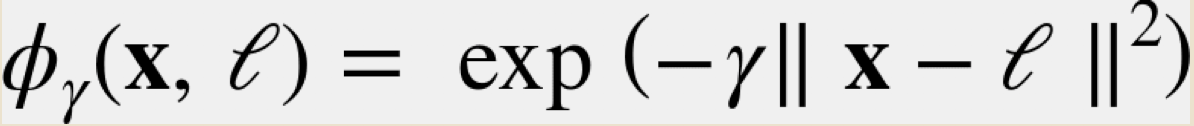
적절한 하이퍼매개 변수 값을 찾는 일반적인 방법은 그리드 검색을 사용하는 것입니다

**2.2) Adding Similarity Features**

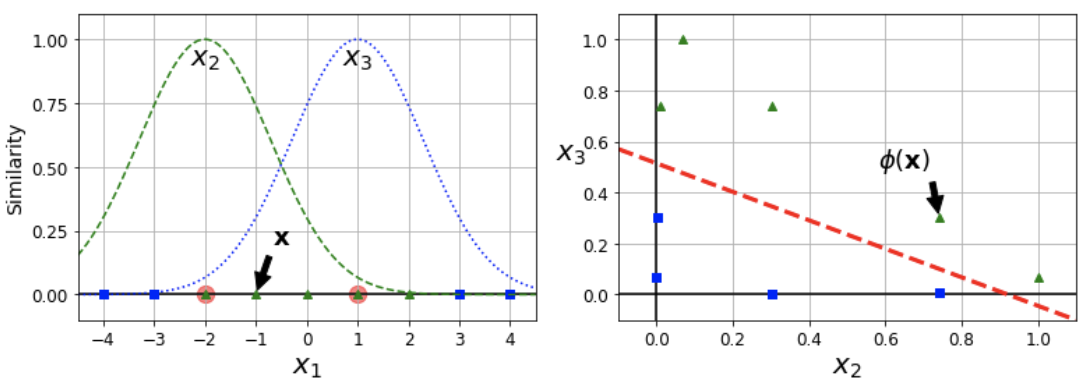
비선형 문제를 해결하는 또 다른 기술은 similarity function을 사용하여 계산된 feature를 추가하는 것입니다.

similarity function은 각 인스턴스가 특정 랜드 마크와 얼마나 비슷한지를 측정하는 함수

예를 들어 앞에서 설명한 1 차원 데이터 세트를 사용하여 x1 = -2 및 x1 = 1에서 두 개의 랜드 마크를 추가. 유사 함수를 γ = 0.3 인 Gaussian Radial Basis Function (RBF)로 정의하자



이것은 0에서 1 까지의 종 모양 함수입니다.



예를 들어 x1 = -1 인스턴스를 살펴 보겠습니다. 첫 번째 랜드 마크에서 1, 두 번째 랜드 마크에서 2의 거리에 위치합니다. 그러므로 새로운 특징은 x2 = exp (- 0.3 × 12) ≈ 0.74이고 x3 = exp (- 0.3 × 22) ≈ 0.30이다.

오른쪽의 플롯은 변환된 데이터 세트(원본 피처를 삭제)를 보여줍니다.(각각의 점들에 대해 위의 공식으로 계산한 것)

- 랜드마크를 선택하는 방법

가장 간단한 방법은 데이터 세트의 모든 인스턴스마다 위치를 획기적으로 만드는 것입니다. 많은 차원이 생성되므로 변형된 트레이닝 세트가 선형으로 분리될 가능성이 높아집니다.

단점은 m 개의 인스턴스와 n 개의 feature가 포함된 트레이닝 세트가

m 개의 인스턴스와 m 개의 feature가 있는 트레이닝 세트로 변형된다는 것입니다

(원래 기능을 삭제한다고 가정). 교육 세트가 매우 큰 경우 동일한 수의 기능으로 끝납니다.

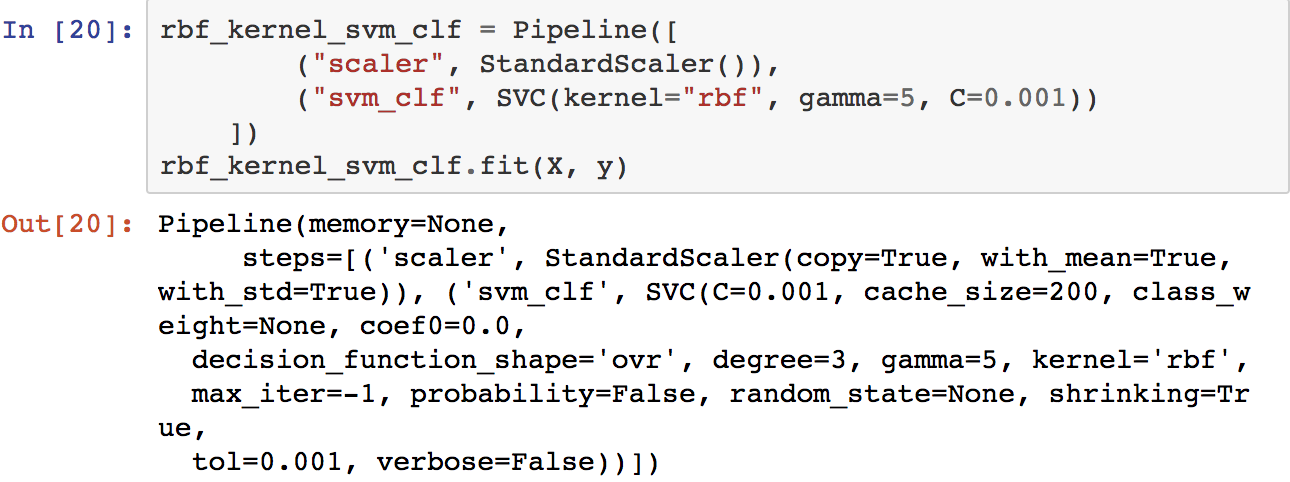
­­­­­­

**2.3) Gaussian RBF Kernel**

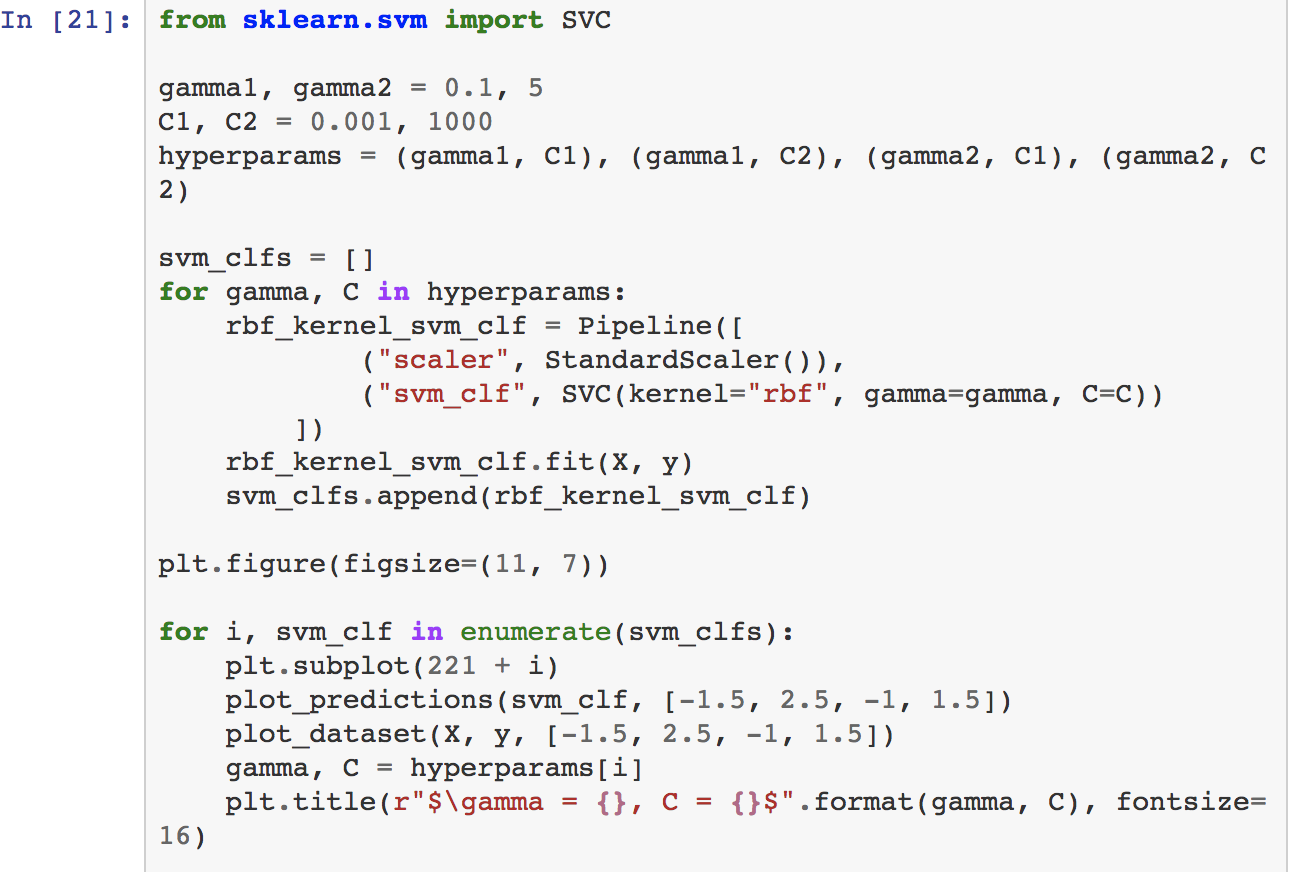
polynomial features방식과 마찬가지로 similarity features방식은 모든 기계 학습 알고리즘에 유용 할 수 있다. 하지만 모든 추가 feature를 계산할 때 특히 대규모 트레이닝 세트를 계산하는 데 많은 비용이 소요될 수 있습니다.

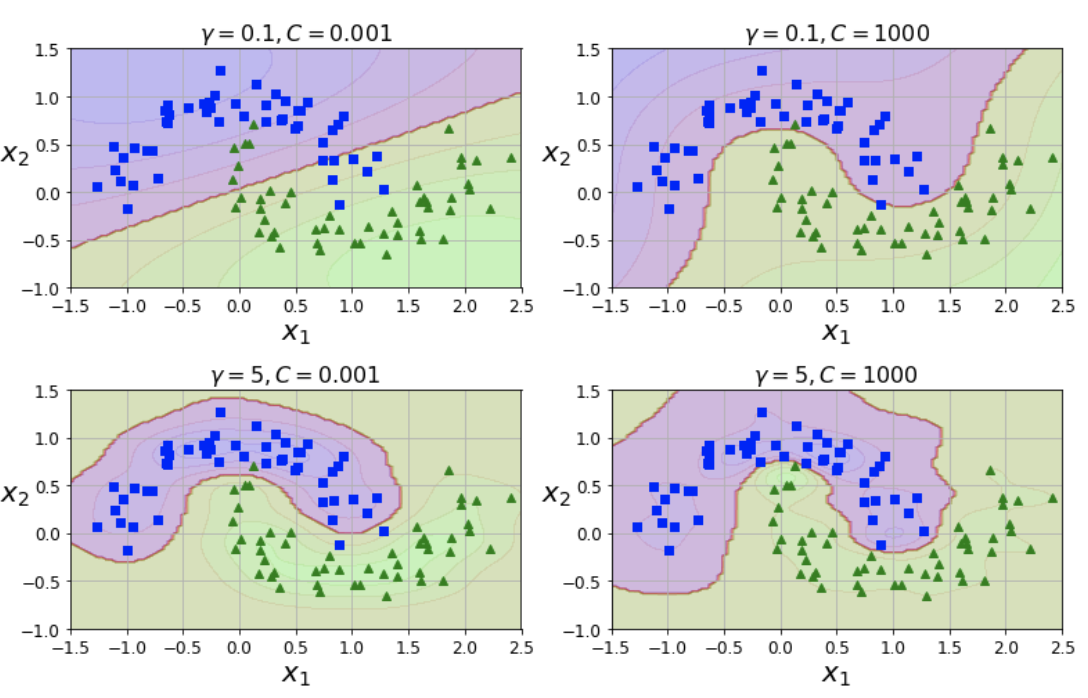
그러나 커널 트릭은 이곳에서 다시 한 번 유용하게 사용될 수 있다. 실제로 많은 추가 feature를 추가하지 않고도 비슷한 결과를 얻을 수 있습니다.

SVC 클래스를 사용하여 Gaussian RBF 커널을 사용해 봅시다.



이 모델은 다음 그림의 왼쪽 아래에 표시됩니다. 다른 플롯은 하이퍼 매개 변수 γ (γ)와 C의 다른 값으로 훈련 된 모델을 보여줍니다.





감마를 증가 시키면 가우시안 종 모양 곡선이 좁아지고 결과적으로 각 인스턴스의 영향 범위는 더 작아집니다(결정 경계가 더욱 불규칙 해지고 개별 인스턴스 주위를 움직입니다. )

반대로 감마 값이 작으면 종 모양 커브가 넓어지므로 인스턴스의 영향 범위가 넓어지고 결정 경계가 더 부드럽게 끝납니다.

그래서 γ는 정규화 hyperparameter와 같은 역할을 합니다

- overfitting이라면 감마값를 줄여야하며, underfitting 피팅이면 감마값을 증가시켜야 합니다.

다른 커널이 있지만 훨씬 드물게 사용됩니다

예를 들어, 일부 커널은 특정 데이터 구조에 특화되어 있습니다. 문자열 커널(String kernels)은 텍스트 문서 나 DNA 시퀀스를 분류 할 때 (예 : string subsequence kernel or kernels based on the Levenshtein distance) 사용됩니다.

[선택할 수있는 커널이 너무 많기 때문에 사용할 커널을 어떻게 결정할 수 있습니까? 일반적으로 LinearSVC는 SVC (kernel = "linear")보다 훨씬 빠르다는 것을 기억하십시오. 특히 트레이닝 세트가 매우 크거나 기능이 풍부하다면 항상 먼저 선형 커널을 사용해보십시오. 훈련 세트가 너무 크지 않은 경우, 가우스 RBF 커널을 시도해야합니다. 그것은 대부분의 경우에 잘 작동합니다. 그런 다음 여분의 시간과 컴퓨팅 성능이 있다면 크로스 검증과 그리드 검색을 사용하여 몇 가지 다른 커널을 시험해 볼 수 있습니다. 특히 교육 세트의 데이터 구조에 특화된 커널이있는 경우 더욱 그렇습니다.]

**2.4) Computational Complexity**

LinearSVC 클래스는 선형 SVM에 최적화 된 알고리즘을 구현하는 liblinear 라이브러리를 기반으로합니다. 커널 트릭은 지원하지 않지만 트레이닝 인스턴스 수와 기능 수에 따라 거의 선형 적으로 조정됩니다. 트레이닝 시간 복잡도는 대략 O (m × n)입니다.

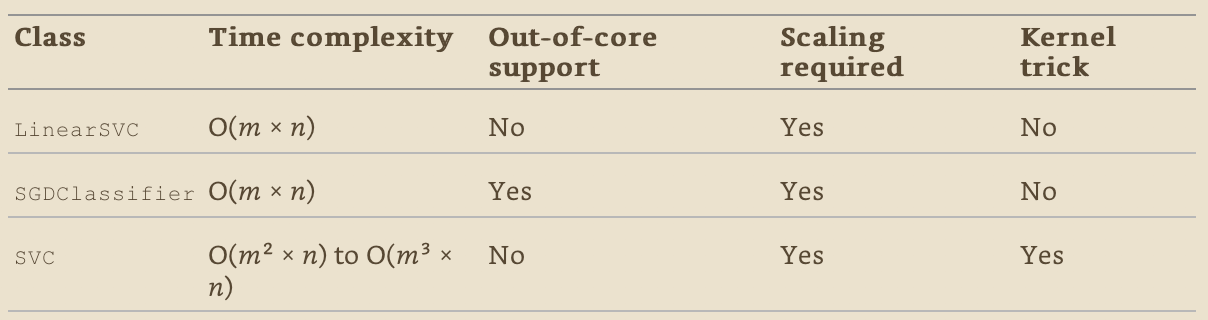
매우 높은 정밀도가 필요한 경우 알고리즘이 오래 걸립니다. 이것은 tolerance hyperparameter ε (Scikit-Learn에서 tol이라고 불린다)에 의해 제어된다.

SVC 클래스는 커널 트릭을 지원하는 알고리즘을 구현하는 libsvm 라이브러리를 기반으로합니다. 2 훈련 시간 복잡도는 일반적으로 O (m2 × n)와 O (m3 × n) 사이입니다. 이는 교육 인스턴스의 수가 커지면 (예 : 수십만 건의 인스턴스) 느려진다는 것을 의미합니다.

이 알고리즘은 복잡하지만 작거나 중간 사이즈 트레이닝 세트에 적합합니다.

그러나 기능 수가 많을 때나, sparse feature(각 인스턴스에 0이 아닌 feature가 거의 없는 경우)일 때 잘 동작합니다.

이 경우 알고리즘은 대략적으로 인스턴스당 0이 아닌 평균 수로 조정된다.

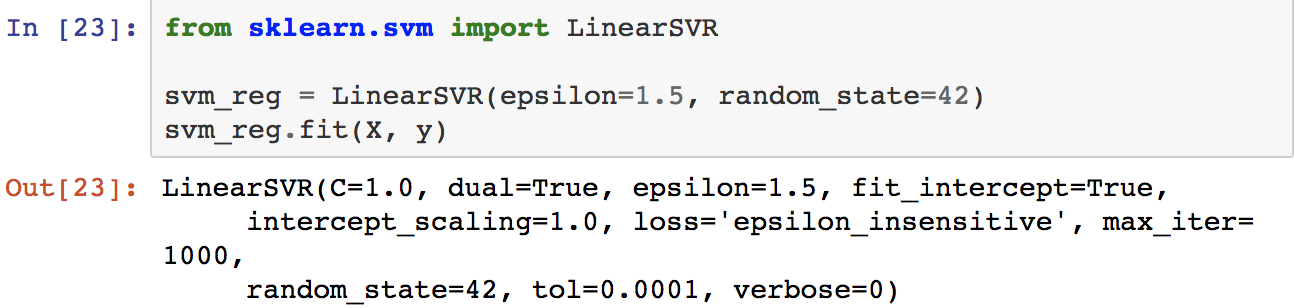


1. **SVM Regression**

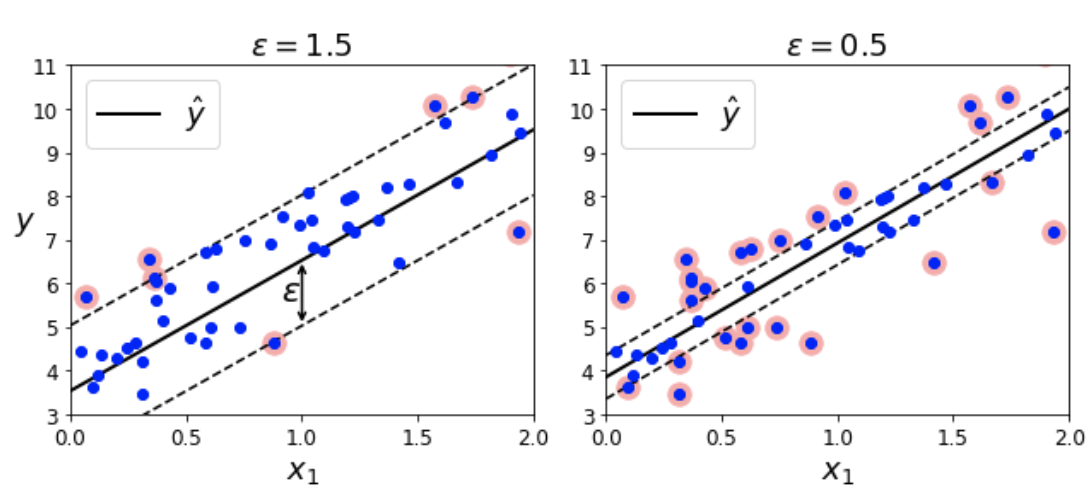
이전에 언급했듯이 SVM 알고리즘은 선형 및 비선형 분류를 지원할 뿐만 아니라 선형 및 비선형 회귀를 지원합니다.

마진위반(margin violation)을 제한하면서 두 클래스 사이에 가능한 가장 먼 거리를 맞추려 하는 대신, SVM 회귀는 마진 위반 (즉, 거리에서 벗어난 인스턴스)을 제한하면서 가능한 한 많은 인스턴스를 거리에 적용 하려고한다.

거리의 너비는 하이퍼매개 변수 ε에 의해 제어됩니다

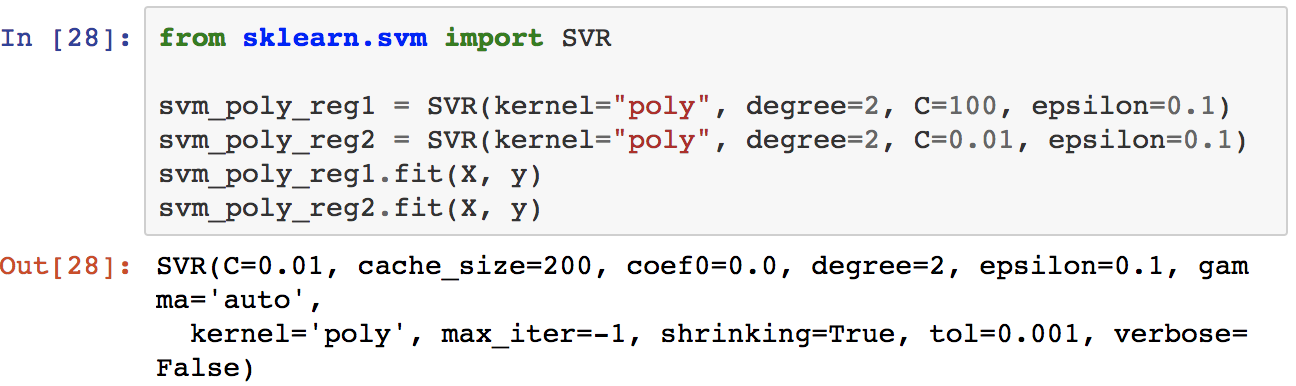


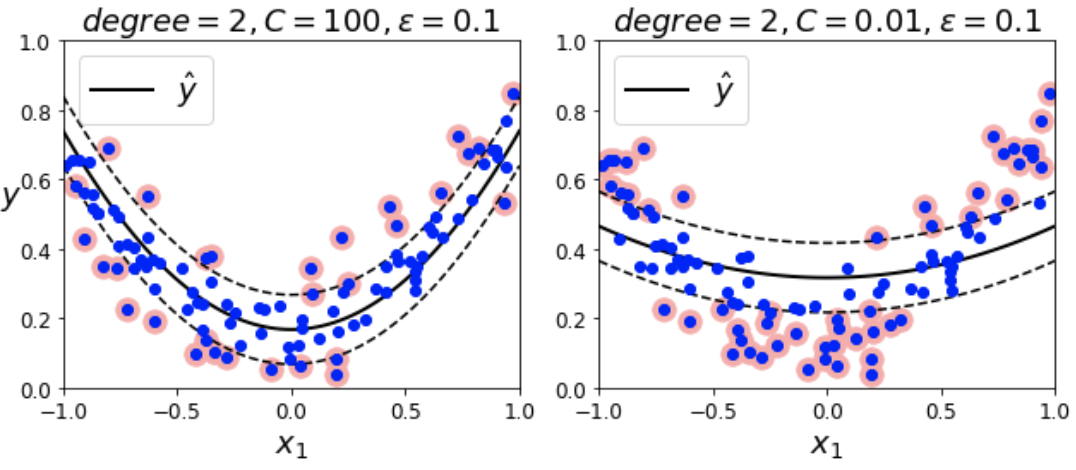
Scikit-Learn의 LinearSVR 클래스를 사용하여 선형 SVM 회귀를 수행 할 수 있습니다. 다음 코드는 위 그림의 왼쪽에 표시된 모델을 생성합니다



이 그림은 큰 마진 (ε = 1.5)과 작은 마진 (ε = 0.5)을 갖는 임의의 선형 데이터에 대해 학습 한 두 개의 선형 SVM 회귀 모델을 보여줍니다.

마진 내에 더 많은 교육 인스턴스를 추가해도 모델의 예측에는 영향을 미치지 않습니다. 따라서 모델은 ε을 구분하지 않는다.





비선형 회귀 작업을 처리하기 위해 커널화 된 SVM 모델을 사용할 수 있습니다.

2 차 다항식 커널을 사용하는 무작위 2 차 트레이닝 세트의 SVM 회귀를 보여줍니다. 왼쪽 플롯 (즉, 큰 C 값)에는 정규화가 거의없고 오른쪽 플롯 (즉, 작은 C 값)에 훨씬 많은 정규화가 있습니다.

SVR 클래스는 SVC 클래스의 회귀 값이고 LinearSVR 클래스는 LinearSVC 클래스의 회귀 값입니다. LinearSVR 클래스는 (LinearSVC 클래스와 마찬가지로) 트레이닝 세트의 크기와 선형으로 확장되지만, 트레이닝 세트가 커지면 (SVC 클래스와 마찬가지로) SVR 클래스가 너무 느려집니다.

1. **Under the Hood**

이 섹션에서는 선형 SVM 분류 자로 시작하여 SVM이 어떻게 예측을 하고 어떻게 트레이닝 알고리즘이 작동하는지 설명합니다.

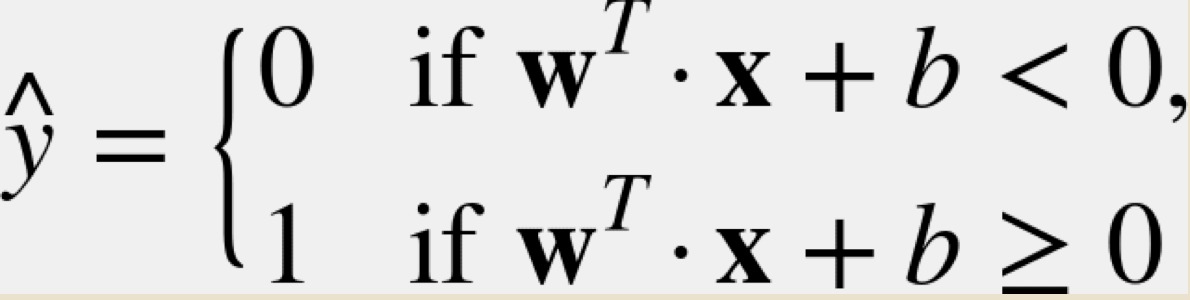
Machine Learning을 시작하는 분이라면 이 장의 마지막 부분에 있는 내용을 건너 뛰고 연습 문제로 바로 넘어갈 수 있습니다. 나중에 SVM에 대해 더 깊이 이해하고자 할 때 다시 돌아옵니다.

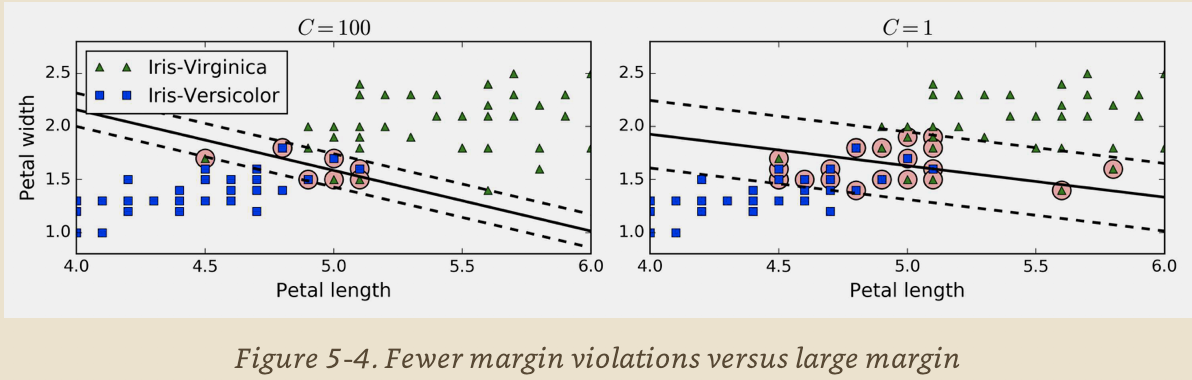
이 장에서는 SVM을 다룰 때보다 편리하고 더 일반적인 다른 규칙을 사용할 것입니다.

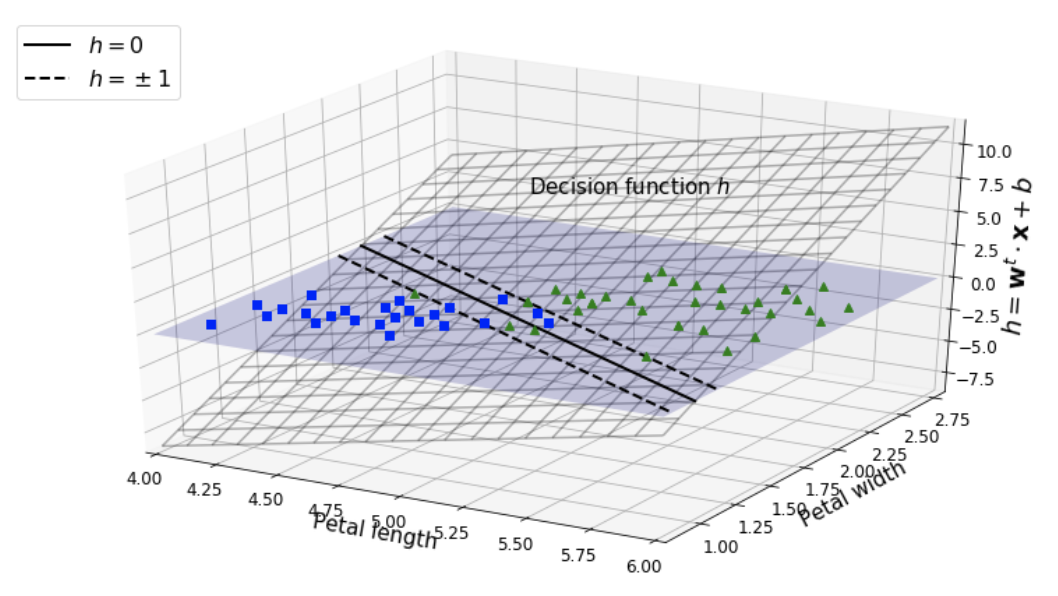
바이어스 term으로 b, feature weight벡터를 w라고 합니다.

* 1. **Decision Function and Predictions**

선형 SVM 분류기 모델은 단순히 결정 함수 wT · x + b = w1 x1 + ⋯ + wn xn + b를 계산하여 새 인스턴스 x의 클래스를 예측합니다. 예측 클래스 ŷ는 결과가 양수이면 1 그렇지 않으면 0







위 그림은 5-4의 왼쪽에 해당하는 그림이다.

이 데이터 세트에는 꽃잎 폭과 꽃잎 길이라는 두 가지 기능이 있기 때문에 이차원 평면입니다. 결정 경계(decision boundary)는 결정 함수(decision function)가 0 인 지점 집합입니다. 두 평면의 교차점이며 직선입니다 (두꺼운 실선으로 표시).

점선은 결정 함수가 1 또는 -1 인 점을 나타내며, 평행하고 결정 경계와 동일한 거리에 있으며 주변에 여백을 형성합니다.

선형 SVM 분류기를 훈련한다는 것은 마진 위반을 피하거나(하드 마진) 한계를 제한(소프트 마진)하면서 이 마진을 최대한 넓게 만드는 w 및 b의 값을 찾는 것을 의미합니다.

* 1. **Training Objective**

결정 함수의 기울기를 고려해보십시오. 이것은 가중치 벡터의 표준 인 ∥ w ∥와 같습니다.

이 기울기를 2로 나눈 경우, 결정 함수가 ± 1 인 점은 결정 경계에서 두 배 멀리 떨어집니다. 즉, 기울기를 2로 나누면 마진에 2가 곱해집니다.

가중치 벡터 w가 작을수록 마진이 커집니다.

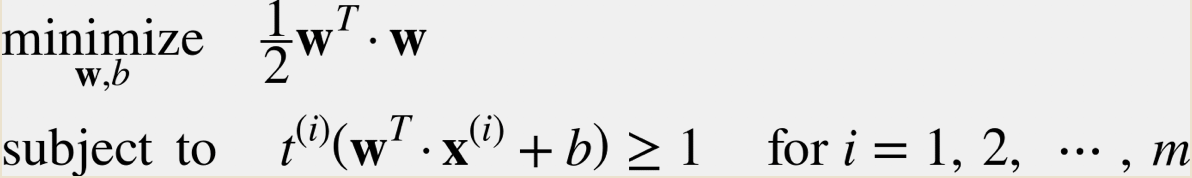


따라서 ∥ w ∥를 최소화하여 큰 마진을 얻고 싶다.

마진 위반 (하드 마진)을 피하려면 모든 긍정적 인 교육 인스턴스에 대해 1보다 커야 하고 부정적인 교육 인스턴스의 경우에는 -1보다 작아야 합니다.

제약 조건은 t(i)(wT · x(i) + b) ≥ 1이다.

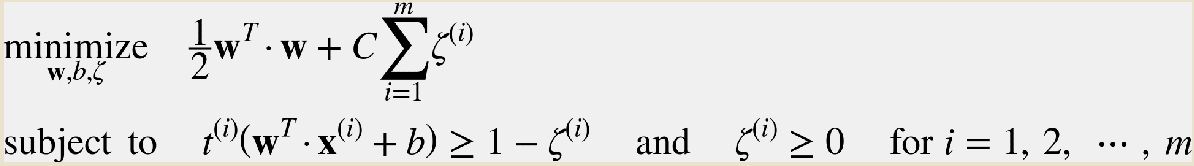
하드 마진 선형 SVM 분류기의 목적을 다음 식의 제약 최적화 문제로 표현할 수 있다.



소프트 마진으로 원하는 것을 얻으려면 각 인스턴스에 대해 여유 변수(slack variable) ζ(i) ≥ 0을 추가해야 합니다 ζ (i)는 i 번째 인스턴스가 마진을 위반할 수 있는 정도를 측정합니다.

이제는 여백 위반을 줄이기 위해 여유 변수(slack variable) ζ(i) 를 가능한 작게 만들고 여백을 늘리기 위해 (1/2)wT · w를 가능한 한 작게 만드는 두 가지 상충되는 목표가 있습니다.

이를 해결하기 위해 하이퍼 매개변수 C를 넣을 수 있다. C는 이 두 목표 사이의 절충점을 정의 할 수 있습니다.

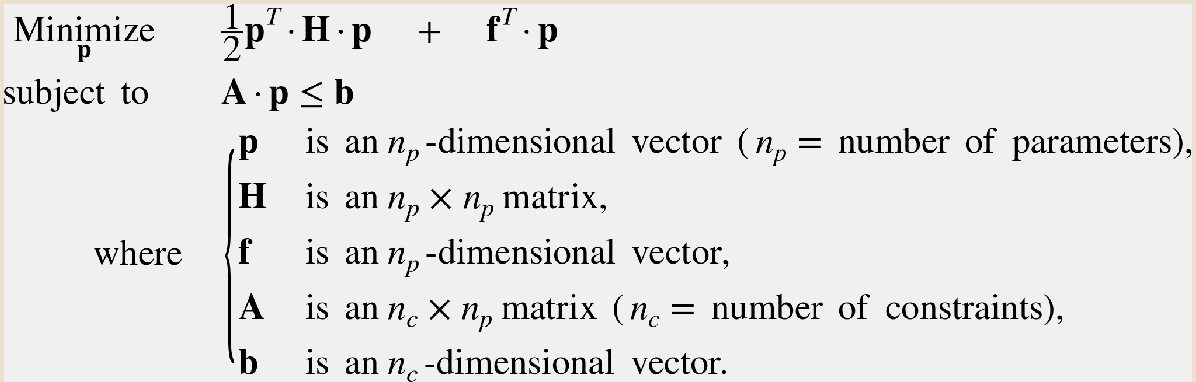


* 1. **Quadratic Programming**

하드 마진과 소프트 마진 문제는 모두 선형 제약 조건에 대한 블록 2차 최적화(convex quadratic optimization)문제입니다.

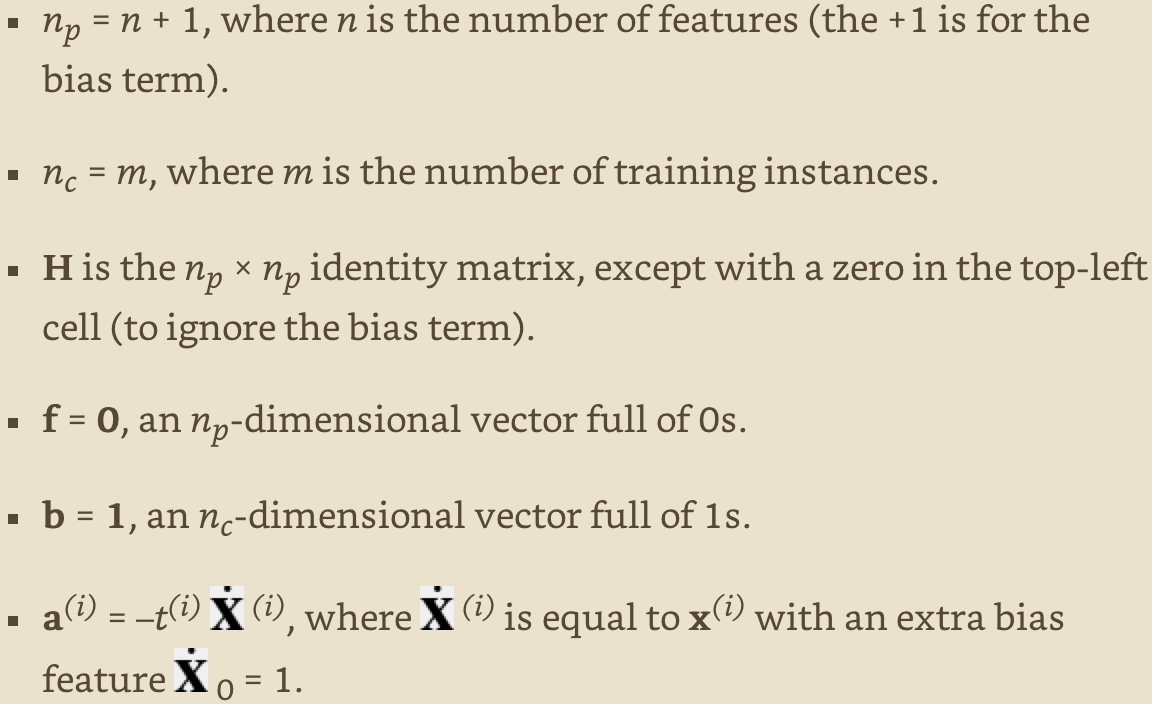
이러한 문제를 Quadratic Programming (QP) 문제라고 합니다

일반적인 문제 공식은 다음과 같다.



식 A · p ≤ b는 실제로 nc 제약 조건을 정의한다.

다음과 같은 방법으로 QP 매개 변수를 설정하면 하드 마진 선형 SVM 분류기 목표를 쉽게 얻을 수 있는지 확인할 수 있습니다.



np = n + 1, 여기서 n은 지형지물의 수입니다 (+1은 바이어스 항을 나타냄).

nc = m, m은 트레이닝 인스턴스의 수입니다.

H는 np × np 단위 행렬이며, 왼쪽 위의 셀에 0이있는 것을 제외하고는 (바이어스 항을 무시).

f = 0, np 차원 벡터가 0으로 채워짐.

b = 1, nc 차원 벡터가 1로 채워짐.

a(i) = -t(i)X(i) 여기서 X(i)는 여분의 바이어스 특성 X0 = 1 인 x(i)와 동일하다.

따라서 하드 마진 선형 SVM 분류기를 훈련하는 한 가지 방법은 이전 매개 변수를 전달하여 기성품 QP 솔버를 사용하는 것입니다. QP 솔버를 사용하여 소프트 마진 문제를 해결할 수 있습니다

* 1. **The Dual Problem**

Primal problem으로 알려진 제한된 최적화 문제가 주어지면 다른 문제이지만 밀접하게 관련된 문제를 표현할 수 있습니다. 이를 Dual problem이라 한다.

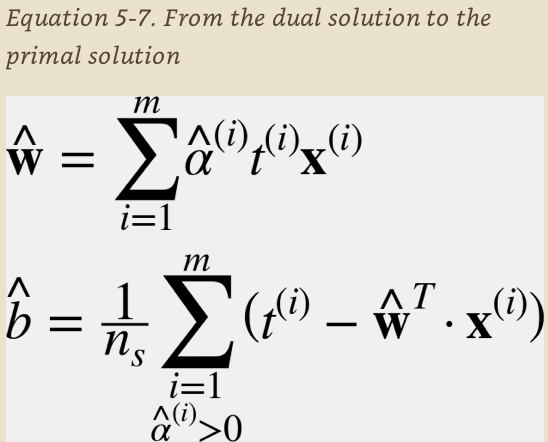
Dual problem에 대한 해답은 전형적으로 Primal problem의 해법에 대한 하한을 제공하지만, 어떤 조건에서는 Primal problem와 같은 해를 가질 수 있다.

다행히도 SVM 문제는 이러한 조건을 충족 시키므로 Primal problem 또는 Dual problem를 해결할 수 있습니다. 두 가지 모두 동일한 솔루션을 갖게 됩니다.

다음 식은 선형 SVM objective의 이중 형태를 보여줍니다



이 방정식을 최소화하는 벡터 α(QP 해석기 사용)를 찾으면 식 5-7을 사용하여 원 문제를 최소화하는 W와 b를 계산할 수 있습니다.

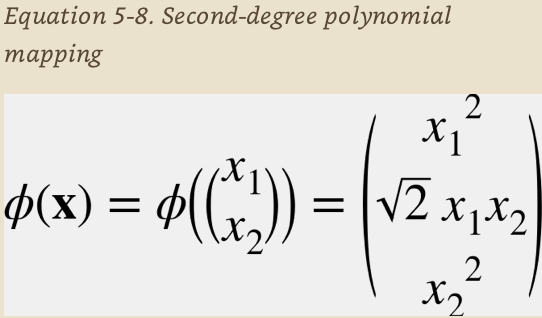


dual problem은 트레이닝 인스턴스의 수가 feature의 수보다 작을 때 primal보다 더 빨리 풀 수 있습니다. 더 중요한 것은 dual problem은 커널 트릭을 가능하게하는 반면, primal은 가능하지 않다는 것입니다.

* 1. **Kernelized SVM**

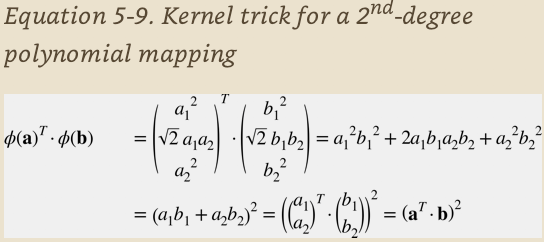
2 차원 다항식 변환을 2 차원 트레이닝 세트 (예 : 위성 트레이닝 세트)에 적용한 다음 변형 된 트레이닝 세트에서 선형 SVM 분류기를 학습한다고 가정합니다.

다음 식은 적용하고자하는 2 차 다항식 매핑 함수 φ를 보여줍니다.



변환 된 벡터는 2 차원이 아닌 3 차원이다.

이제 이 2차 다항식 매핑을 적용하고 변환 된 벡터의 내적을 계산하면 2차원 벡터 2개에 어떤 일이 발생하는지 살펴본다.



변환 된 벡터의 내적은 원래 벡터의 내적의 제곱과 동일하다 : φ (a)T · φ (b) = (aT · b)2

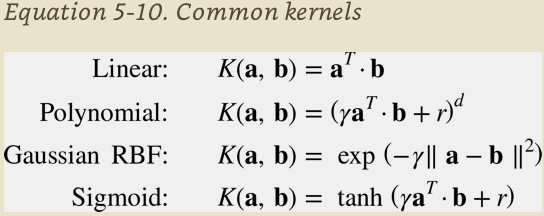
모든 훈련 인스턴스에 변환 φ를 적용하면 dual problem(식5-6)식은 내적 (φ) (x(i)) T · φ (x(j)) 를 포함하게 된다.

그러나 φ가 식 5-8에 정의 된 2 차 다항식 변환이면 변환된 벡터의 내적을 간단히 바꿀 수 있습니다. 따라서 실제로는 트레이닝 인스턴스를 전혀 변형할 필요가 없습니다.

결과는 실제로 훈련 세트를 변환한 다음 선형 SVM 알고리즘에 맞추는 문제를 겪었을 때와 똑같습니다. 그러나 이 트릭은 전체 프로세스를 훨씬 더 효율적으로 계산합니다.

함수 K (a, b) = (aT · b) 2는 2 차 다항 커널로 불린다. 기계 학습에서 커널은 변환 φ를 계산할 필요없이 (또는 심지어 알기 위해) 원본 벡터 a와 b에만 기초하여 내적을 계산할 수 있는 함수입니다.

다음 식은 가장 일반적으로 사용되는 커널입니다.



[머서의 정리에 따르면, 함수 K (a, b)가 머서의 조건 (K는 연속적이고 대칭인 인수 K (a, b) = K (b, a))이라고 하는 몇 가지 수학적 조건을 존중하면, K (a, b) = φ (a)T · φ (b)가 되도록 다른 공간에 a와 b를 매핑하는 함수 φ가 존재한다.

따라서 φ가 무엇인지 모를지라도 φ가 존재한다는 것을 알기 때문에 K를 커널로 사용할 수 있습니다.

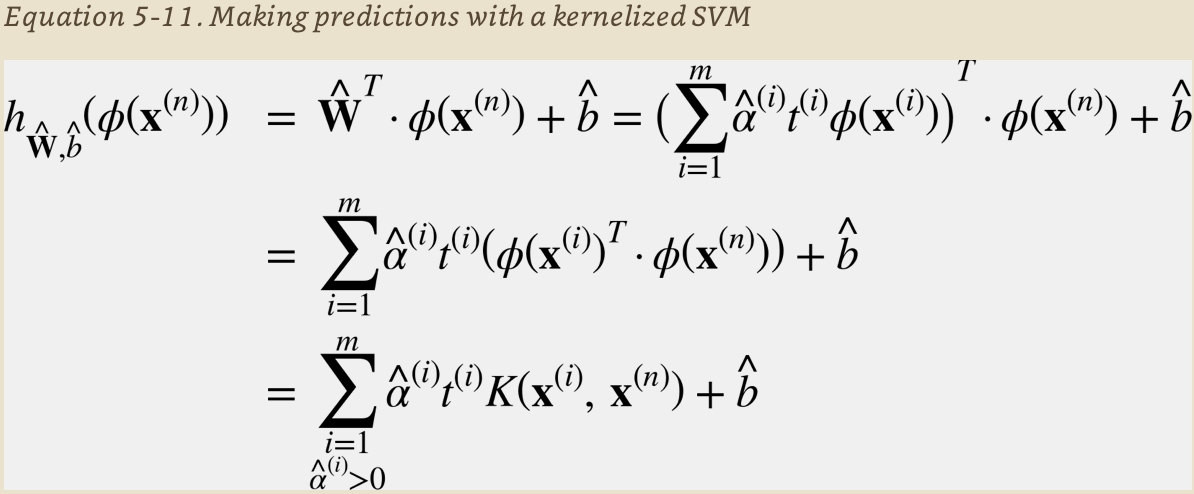
Gaussian RBF 커널의 경우, φ가 실제로 각 학습 인스턴스를 무한 차원 공간에 매핑한다는 것을 알 수 있으므로 매핑을 실제로 수행 할 필요가 없습니다.

시그모이드(Sigmoid) 커널과 같이 자주 사용되는 커널은 머서의 모든 조건을 준수하지 않지만 일반적으로 실제로 잘 작동합니다.]

식 5-7은 선형 SVM 분류 자의 경우 이중 해에서 근원 해법으로 이동하는 방법을 보여줍니다. 그러나 커널 트릭을 적용하면 φ (x (i))가 포함 된 방정식으로 끝납니다. 사실 φ (x (i))와 같은 차원 수를 가져야합니다.

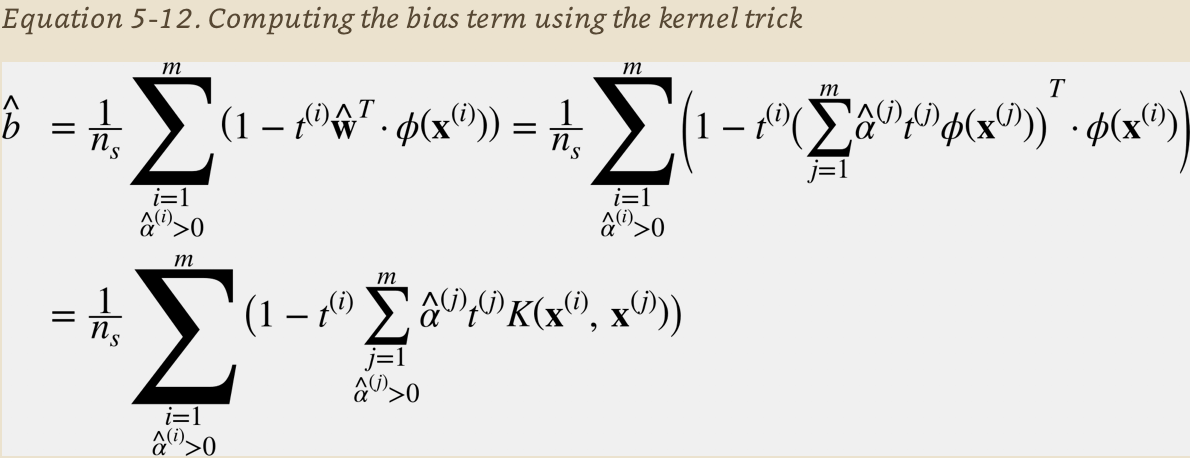
방정식 5-7에서 W에 대한 공식을 새로운 인스턴스 x (n)에 대한 의사 결정 함수에 적용할 수 있으며, 입력 벡터 사이에 내적만 있는 방정식을 얻을 수 있습니다

이렇게 하면 커널 트릭을 다시 사용할 수 있습니다



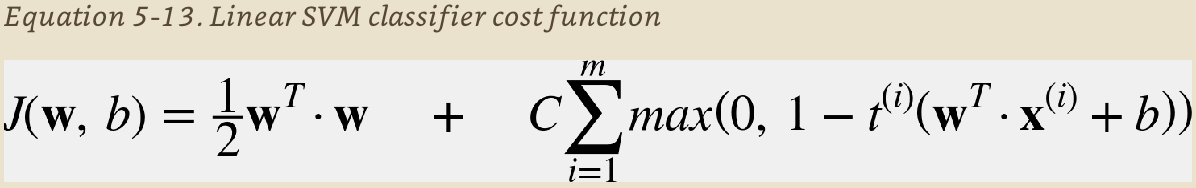
지원 벡터에 대해서만 α (i) ≠ 0이므로, 예측을 하는 것은 모든 학습 인스턴스가 아닌 지원 벡터만으로 새로운 입력 벡터 x (n)의 내적을 계산하는 것을 포함한다.

물론 동일한 트릭을 사용하여 바이어스 항 b를 계산해야 합니다



* 1. **Online SVMs**

온라인 SVM 분류기를 간략히 살펴 보도록 하겠습니다. 온라인 학습은 일반적으로 새 인스턴스가 도착할 때 점진적으로 학습한다는 것을 의미합니다.



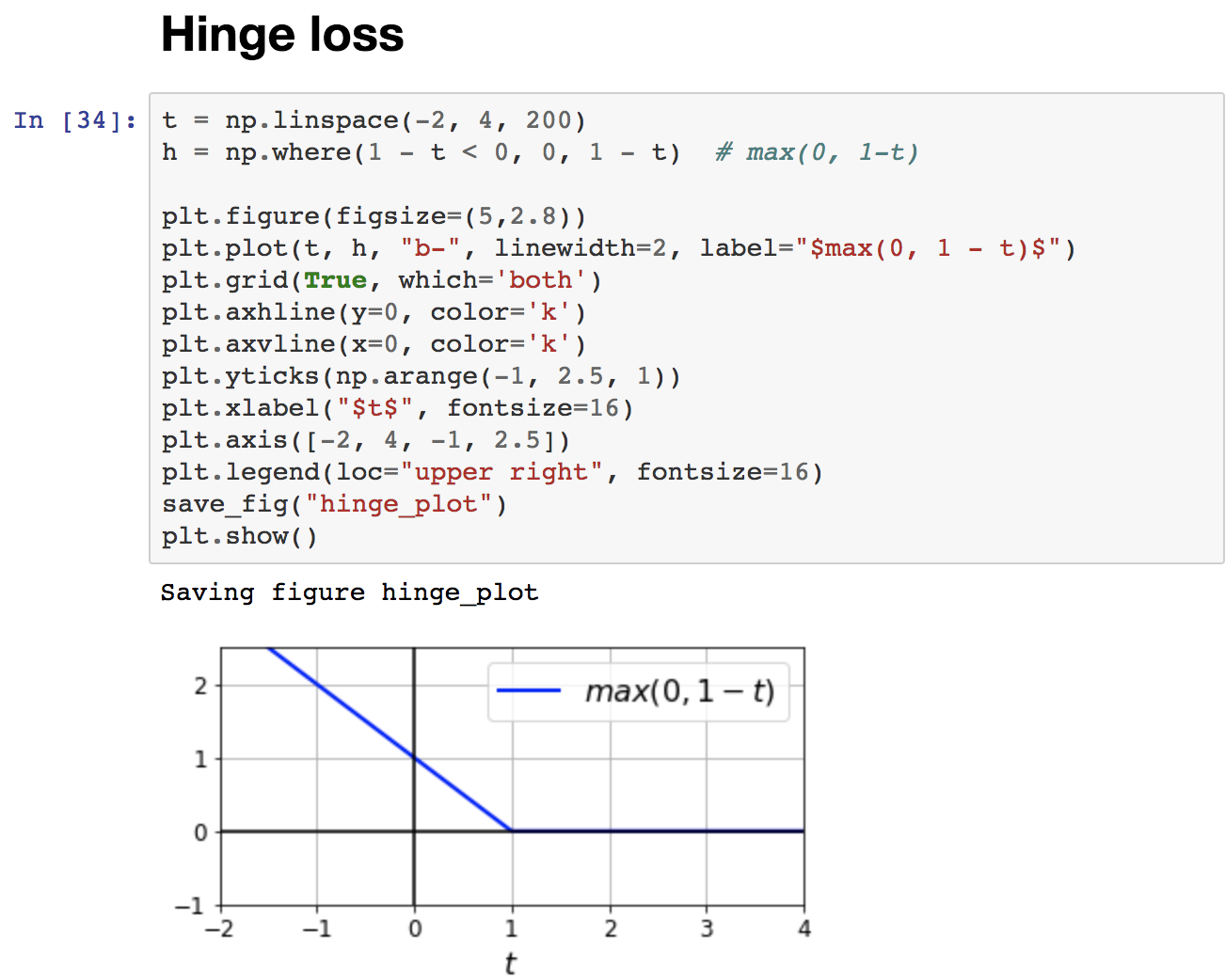
선형 SVM 분류 자에 대한 하나의 방법은primal problem으로부터 도출 된 식 5-13의 비용 함수를 최소화하기 위해 Gradient Descent(예를 들어 SGDClassifier)를 사용하는 것이다.

불행히도 QP를 기반으로 하는 방법보다 훨씬 느리게 수렴합니다.

비용 함수의 첫 번째 항은 모델이 작은 가중치 벡터 w를 갖도록 넣어 큰 여백을 만듭니다.

두 번째 항은 모든 마진 위반의 합계를 계산합니다.

거리에서 벗어나 올바른 위치에 있는 경우 인스턴스의 여백 위반은 0과 같거나 그렇지 않으면 거리의 올바른 면까지의 거리에 비례합니다.



함수 max (0, 1 - t)는 힌지 손실 함수 (hinge loss function)라고 부릅니다.

t < 1 일 때 그 기울기는 -1 이고. t > 1 일 경우 0입니다.

t = 1에서는 미분 할 수 없지만 Lasso 회귀와 마찬가지로 여전히 ​​t = 1 (즉, -1과 0 사이의 모든 값)에서 임의의 Gradient Descent를 사용할 수 있습니다.

온라인 커널화 된 SVM을 구현하는 것도 가능합니다. 예를 들어 "Incremental and Decremental SVM Learning" 또는 “Fast Kernel Classifiers with Online and Active Learning.”를 사용할 수 있습니다. 그러나 Matlab 및 C ++에서는 이러한 기능이 구현됩니다.

대규모 비선형 문제의 경우 Neural network(신경망)을 대신 사용하는 것이 좋습니다