해당 논문의 주된 구성은 위와 같다. 지금은 여기서 빨간색 테두리 부분에 대해 중점적으로 알아 볼 것이다. 이를 위해서 우선 기본적인 개념 몇 개에 대해 정리하고, 뒤에서 Python Code 구현을 진행해 보기로 한다. 설명에 있어서 고려대 강필성 교수님의 자료의 내용이 포함되어 있는 부분이 있음을 사전에 말씀드립니다.

An Introduction to Kernel-Based Learning Algorithms

1. Introduction
2. Learning to Classify – Some Theoretical Background
   - A. VC Dimension in Practice
   - B. Margins and VC Dimension
3. Nonlinear Algorithms in Kernel Feature Spaces
   - A. Wrapping Up
4. Supervised Learning
   - A. Support Vector Machines
     - 1) Sparsity
     - 2) nu-SVMs
     - 3) Computing the Threshold
     - 4) A Geometrical Explanation
   - B. Kernel Fisher Discriminant
     - 1) Optimization
   - C. Connection between Boosting and Kernel Methods
   - D. Wrapping Up
5. Unsupervised Learning
   - A. Kernel PCA
   - B. Single-Class Classification
6. Model Selection
   - 1) Bayesian evidence framework
   - 2) PAC
   - 3) Cross Validation
7. Applications
   - A. Supervised Learning
     - 1) OCR
     - 2) Analyzing DNA Data
   - B. Benchmarks
   - C. Unsupervised Learning
     - 1) Denoising
8. Conclusion and Discussion

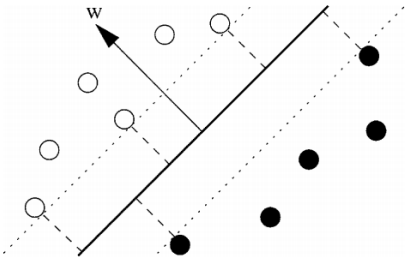
Support Vector Machines - Supervised Learning

◆ 정리

⊙ 기본 개념

기본적인 Support Vector Machine은 Linear Classification 중 하나라고 볼 수 있다. 아래 그림과 같이 동그라미와 점이 있을 경우, 점과 동그라미를 가장 잘 구분하는 선인 boundary 를 찾게 해주는 알고리즘이다. 이는 고차원 공간에서는 boundary라는 용어보다는 hyperplane이라고 부르게 된다.
training 샘플이 위 그림과 같이 hyperplane 상에 표현된다고 할 때, 다음 형식의 함수를 선택할 수 있다.

$$f(x) = (w \cdot x) + b$$



$$R[f] \leq R_{emp}[f] + \sqrt{\frac{h\left(ln\frac{2n}{h}+1\right) - ln\left(\frac{\delta}{4}\right)}{n}}$$

아래에서는 SVM의 몇가지 용어와 개념에 대해 정리하고 갈 것이다.

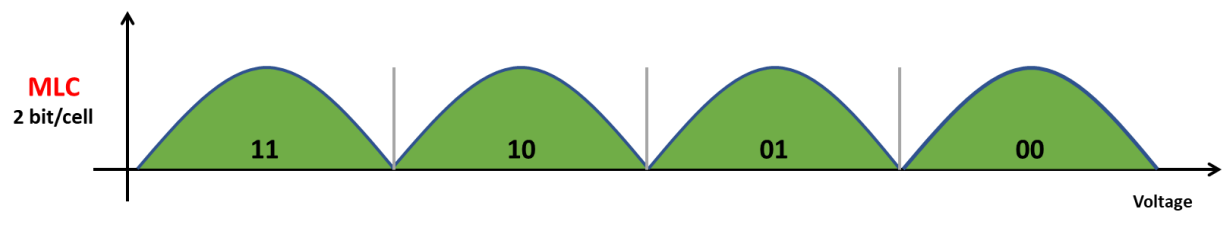
- Dichotomy
- Shatter
- VC Dimension
- Hard Margin
- Soft Margin

⊙ Dichotomy

0 or 1 의 값만 가질 수 있는 두 변수가 A, B가 있다고 해보자. 그렇다면 해당 조합의 가능한 경우는 (0. 0), (0, 1), (1, 0), (1, 1) 이렇게 4가지 경우 중 하나로 존재할 것이다. 해당 경우의 가능한 조합의 수는 $2^2 = 4$로 표현된 것이다. SVM에서는 계산의 편리함을 위해 0 or 1대신 -1/+ 1로 사용하고 있다.

이를 일반적으로 두개 중 하나의 값을 가질 수 있는 변수의 개수가 n개라고 한다면 그 가능한 조합의 수는 2^n라고 표현할 수 있다. 그리고 가능한 모든 종류로 나누는 것을 'Dichotomy' 라고 한다.

☞ 해당 개념을 내가 알고 있는 도메인 지식으로 이해해 본다면, NAND Flash Memory의 Cell과 유사한 개념으로 생각할 수 있다. 아래 그림처럼 Cell 당 2bit을 저장하기 위해서 4개의 State를 존재해야 함을 알 수 있다. Cell당 3bit 을 저장하기 위해서는 {0,0,0}, {0,0,1}, … {1,1,1}까지 8가지의 경우가 존재해야 한다는 것을 알 수 있다. 데이터를 저장한다는 것은 그것을 구분하여 읽기도 가능함을 의미한다.



다음으로는 Shatter 의 대한 개념을 살펴본다.

⊙ Shatter

공간 상에서 선형 분류기는 한 개의 직선으로 앞서 확인한 Dichotomy를 모두 표현할 수 있느냐 를 의미한다고 볼 수 있다. 아래 그림으로 확인해 보면 빨간색 테두리 부분에 대해서는 구분 할 수 없는 것을 알 수 있다. 이를 일반적으로 우리는 XOR 경우를 말하고 선형으로는 해당 조건에 대해 분류가 불가능 함을 알 수 있다.

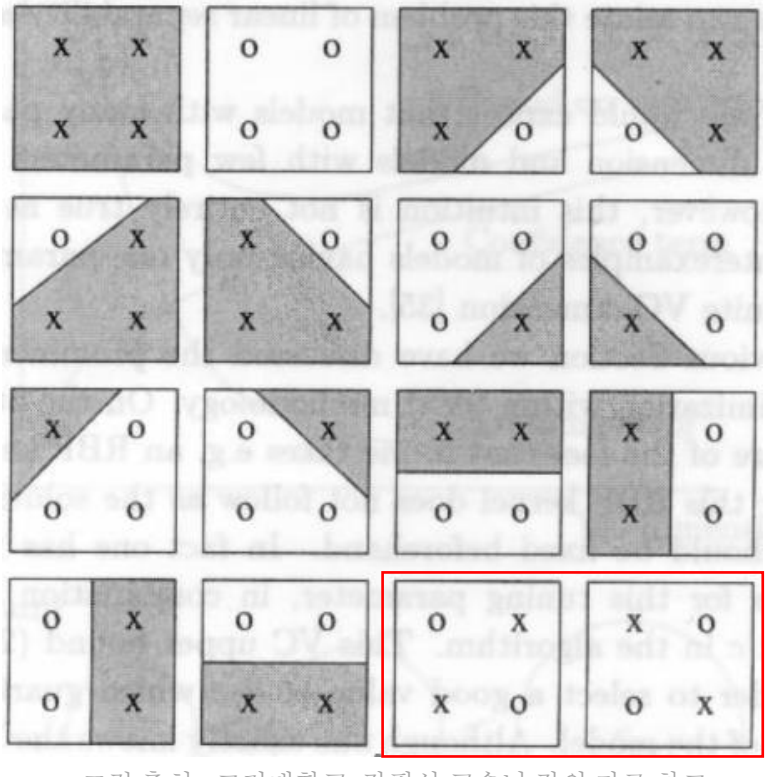


그림 출처 : 고려대학교, 강필성 교수님 강의 자료 참고

다음으로는 VC Dimension 의 대한 개념을 살펴본다.

⊙ VC Dimension

Vapnik-Chervonekis(VC) Dimension이라고 불리우는 용어는 hypothesis space 의 복잡도라고 할 수 있다. 이는 SVM 분류기가 얼마나 복잡한 데이터를 표현할 수 있느냐의 여부는 Shattering 할 수 있는 가장 많은 데이터의 개수라고 생각할 수 있다.

일반적으로 N 차원의 hyperplane 의 VC Dimension 은 N+1 이라고 한다. Margin 을 최대화 한다는 말은 VC Dimension 을 줄인다는 의미와 같다고 할 수 있다.

다음으로는 Margin에 대해 알아 본다.

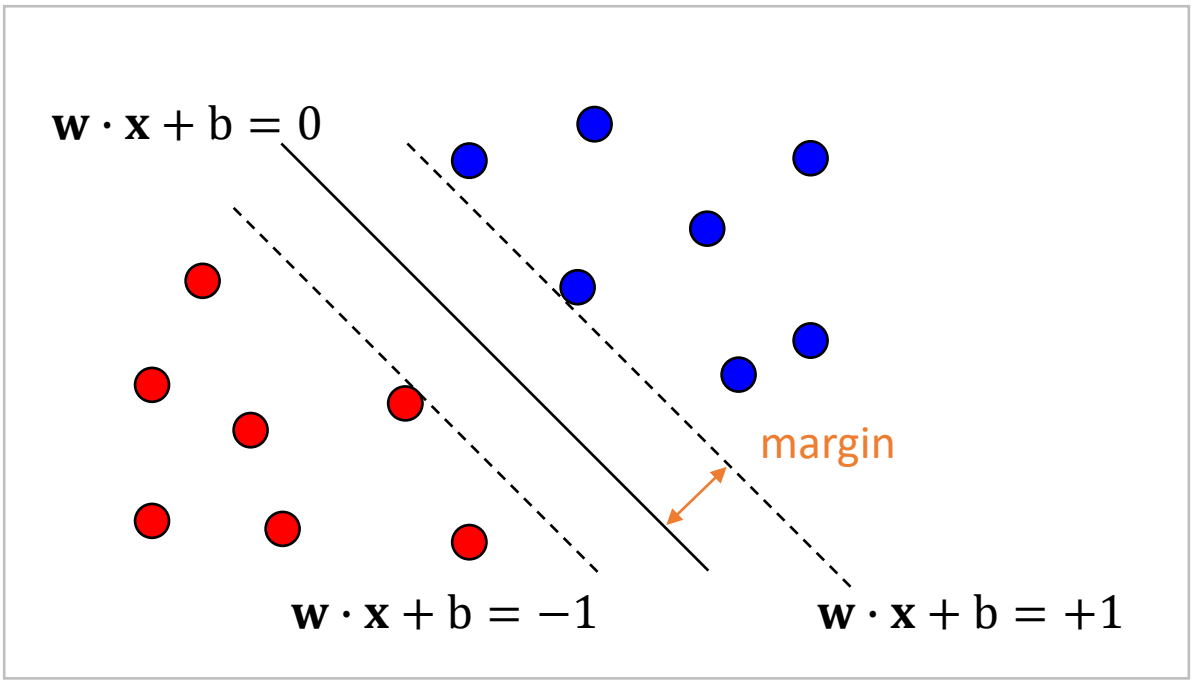
일반적으로 Margin이라 하면 안정성이 유지되는 여분이라고 생각할 수 있다. 제품 수명에 대한 스펙이 5년이라고 한다면 이는 최소한의 보장되어야 하는 기간을 의미하고 추가 기간에 대해서는 우리는 Margin 이라고 할 수 있다. 이처럼 SVM에서는 Margin에 대한 개념을 두가지고 나누어 설명하고 있다. Hard Margin과 Soft Margin에 대해서 알아보자.

⊙ Margin

아래와 같이 두개의 Class에 대해 분류 문제가 있다. 두 Class를 분류해주는 직선 $\mathbf{w} \cdot \mathbf{x} + b = 0$ 이 존재하고 그 직선을 기준으로 두 Class는 분류가 잘 됨을 확인할 수 있다. 여기서 점선으로 표시된 $|\mathbf{w} \cdot \mathbf{x} + b| = 1$ 직선을 사이에 실전이 존재하고 점선과 실선 사이의 거리를 Margin이라고 표현한다. Margin의 의미는 두 class를 분류함에 있어서 hyperplane 상의 Class의 관측치가 Margin 거리만큼은 침범하더라도 Class 분류에는 오류가 없음을 의미한다. SVM은 이 Margin을 최대화하는 분류 경계면을 찾는 알고리즘이라고 할 수 있다.

⊙ Hard Margin

Hard Margin은 Margin안에 관측치가 들어오는 것을 허용하지 않는다. 그렇게 되면 Margin은 상대적으로 줄어들게 될 것이다. 하지만 그만큼 tight하게 구분하는 방법을 의미한다. 아래 그림을 보면 알 수 있듯이, hyperplane상의 관측치는 Margin인 점선을 넘지 않도록 boundary를 찾는 것을 알 수 있다.

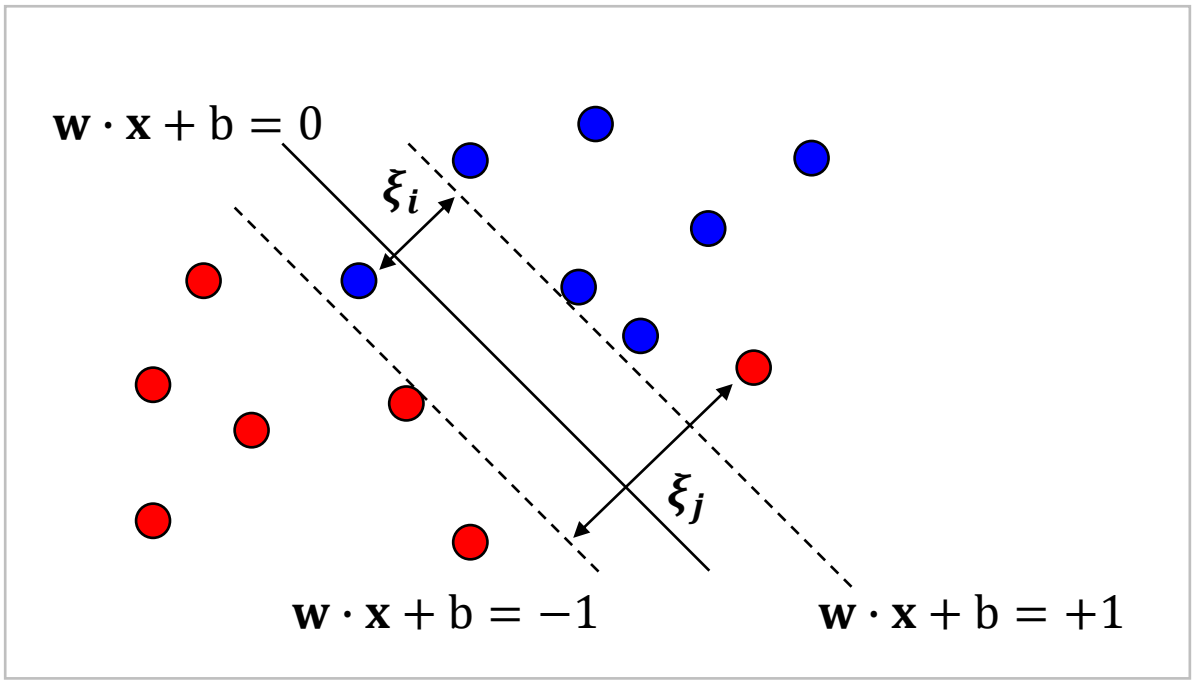


⊙ Soft Margin

Soft Margin은 관측치를 허용하지 않던 Hard Margin과는 달리 Margin 안에 관측치를 허용하고 침투한 관측에 대해 penalty 를 부여하는 방식이다. C-SVM이 Soft Margin이 적용된 알고리즘이고 목적식을 표현하면 아래와 같다.

$$min \frac{1}{2}||w||^2 + C\sum_{i=1}^{n}\xi_i$$

C는 사용자가 설정 가능한 hyperparameter이고, C값이 커질수록 margin의 폭이 줄어들고, 반대로 C값이 작아질수록 margin의 폭이 커짐을 수식을 통해 확인이 가능하다.



☞ Soft와 Hard Margin에 대한 개념을 반도체 제품에서 빗대어 생각해보겠다. Margin의 관측치를 허용하지 않는 개념은 DRAM에서의 Error Bit을 하나도 허용하지 않는 것과 유사하다. 그리고 Margin의 관측치를 허용하는 Soft Margin은 NAND Flash처럼 Error Bit을 어느정도 허용하는 개념과 유사하다고 볼 수 있다.

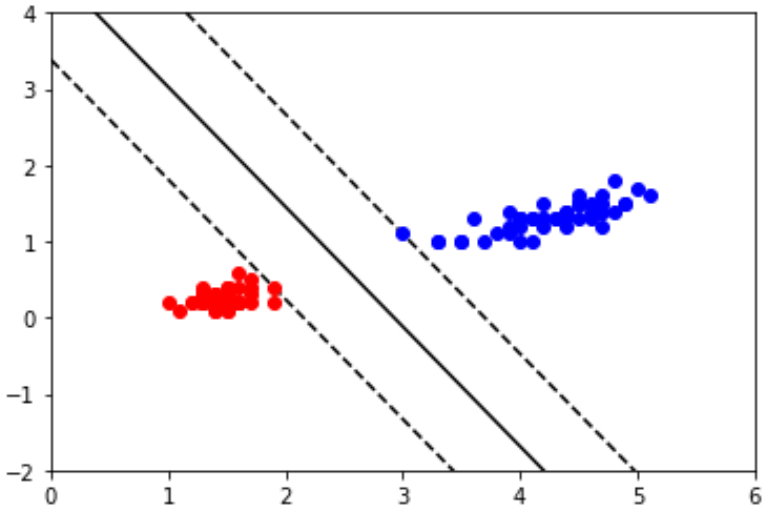
Support Vector Machines - Supervised Learning

◆ Hard Margin – SVM Code 구현

아래 SVM의 sklearn을 통해 구현한 코드이다. SVC라고 모델 이름이 되어 있는 이유가 궁금해서 찾아본 결과, SVC의 C는 “Classification이기 때문이다” 라고 확인하였다. 같은 이유로 SVR은 “Regression의 R을 의미”한다. 데이터 셋은 iris 데이터 셋을 불러와서 사용하였고, data에서 2는 꽃잎 길이를 의미하는 데이터이고, 3은 꽃잎 너비를 의미하는 변수로 2차원 데이터를 생성 하였다.
Hard Margin 확인을 위해서 Linear와 C 값을 inf로 설정하였다. 또한 decision boundary와 upper, lower 의 margin을 함께 수식을 통해 구하였다.

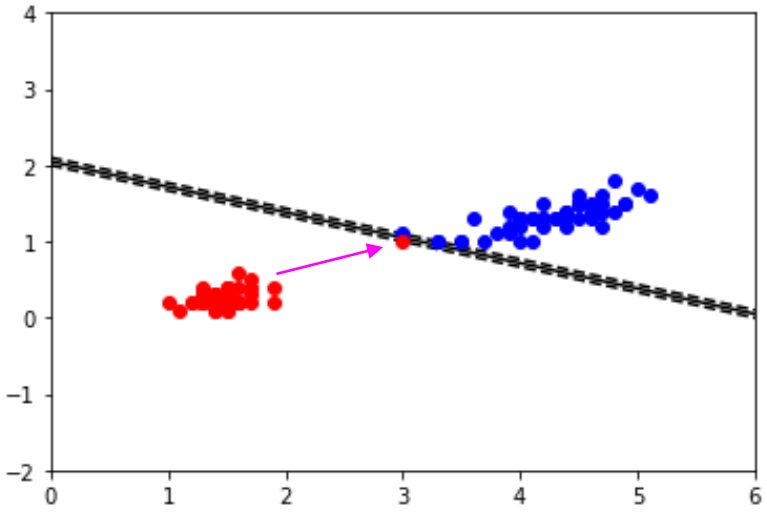
```
from sklearn.svm import SVC
from sklearn import datasets
import numpy as np
import matplotlib
import matplotlib.pyplot as plt

iris = datasets.load_iris()
X = iris["data"][:, (2, 3)]
y = iris["target"]
X = X [ (y==0) | (y==1) ]
y = y[ (y==0) | (y==1)]


# SVC Training
linear_svc = SVC(kernel="linear", C=float("inf"))
linear_svc.fit(X, y)

# Boundary 계산
w = linear_svc.coef_[0]
b = linear_svc.intercept_[0]
x0 = np.linspace(0, 7, 100)
decision_boundary = -w[0]/w[1] * x0 - b/w[1]

# Margin 계산
margin = 1/w[1]
upper = decision_boundary + margin
lower = decision_boundary - margin

# 그래프 표현
plt.plot(x0, decision_boundary, "k-")
plt.plot(x0, upper, "k--")
plt.plot(x0, lower, "k--")
plt.plot(X[:, 0][y==1], X[:, 1][y==1], "bo")
plt.plot(X[:, 0][y==0], X[:, 1][y==0], "ro")
plt.xlim(0, 6)
plt.ylim(-2, 4)
plt.show()
```

해당 SVC training 결과를 그래프로 확인한 결과는 다음과 같다.

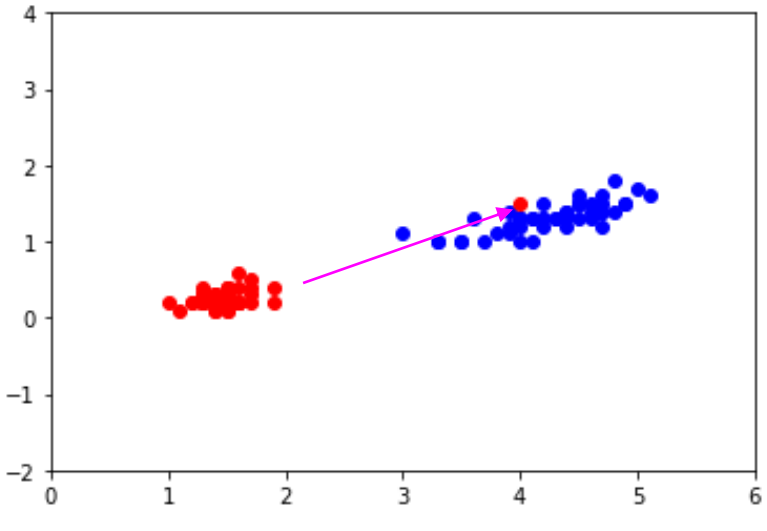


Margin의 upper와 lower에 겹쳐 있는 관측치가 support vector이다. 그리고 해당 결과에서 확인할 수 있듯이, margin은 어떤 관측치의 침투도 허용하지 않는 결과를 확인 할 수 있었다.

이번에는 관측치를 임의로 변화를 주어 Hard Margin의 경우의 SVC는 어떻게 구성되는지 확인해볼 것이다. 빨간 포인트 하나를 파란 영역으로 붙여 보았다. 아래 그래프에서 확인 되듯이 SVC는 동작하고 margin이 엄청 작아졌지만 구분이 가능한 것을 확인할 수 있다.



이번에는 빨간 포인트를 더욱 옮겨서 파랑 영역 속으로 침투 시킨 결과 해당 SVC는 동작하지 않았다. 이는 예상했던 결과처럼 Hard Margin의 경우에는 어떠한 관측치 섞임도 허용하지 않기에 어쩌면 당연한 결과일지도 모른다.

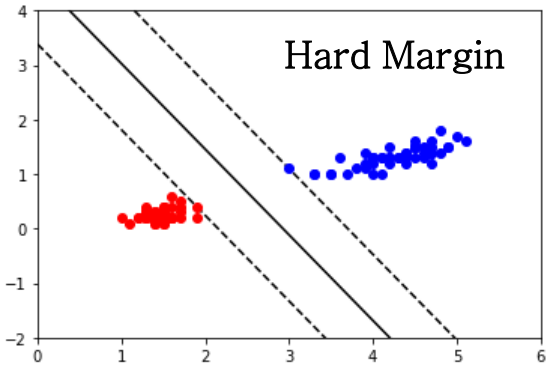


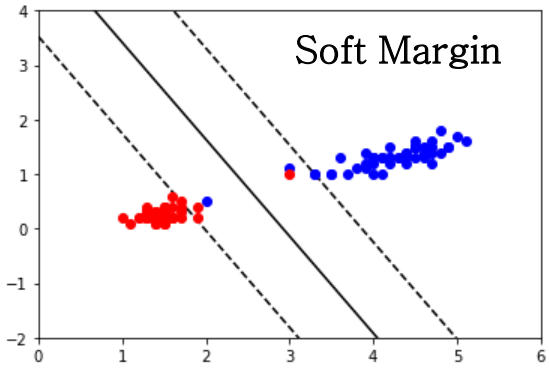
◆ Soft Margin – SVM Code 구현

아래 SVM의 sklearn을 통해 구현한 코드이다. SVC라고 모델 이름이 되어 있는 이유가 궁금해서 찾아본 결과, SVC의 C는 “Classification이기 때문이다” 라고 확인하였다. 같은 이유로 SVR은 “Regression의 R을 의미”한다. 데이터 셋은 iris 데이터 셋을 불러와서 사용하였고, data에서 2는 꽃잎 길이를 의미하는 데이터이고, 3은 꽃잎 너비를 의미하는 변수로 2차원 데이터를 생성 하였다.
Soft Margin 확인을 위해서 Linear와 C 값을 0.5, 10으로 실험을 진행하였다.

```
from sklearn.svm import SVC
from sklearn import datasets
import numpy as np
import matplotlib
import matplotlib.pyplot as plt

iris = datasets.load_iris()
X = iris["data"][:, (2, 3)]
y = iris["target"]
X = X [ (y==0) | (y==1) ]
y = y[ (y==0) | (y==1)]

X[0,] = [3,1]
X[50,] = [2,0.5]

# SVC Training
# linear_svc = SVC(kernel="linear", C=float("inf"))
linear_svc = SVC(kernel="linear", C=0.5)
# linear_svc = SVC(kernel="linear", C=10)
linear_svc.fit(X, y)

# Boundary 계산
w = linear_svc.coef_[0]
b = linear_svc.intercept_[0]
x0 = np.linspace(0, 7, 100)
decision_boundary = -w[0]/w[1] * x0 - b/w[1]

# Margin 계산
margin = 1/w[1]
upper = decision_boundary + margin
lower = decision_boundary - margin

# 그래프 표현
plt.plot(x0, decision_boundary, "k-")
plt.plot(x0, upper, "k--")
plt.plot(x0, lower, "k--")
plt.plot(X[:, 0][y==1], X[:, 1][y==1], "bo")
plt.plot(X[:, 0][y==0], X[:, 1][y==0], "ro")
plt.xlim(0, 6)
plt.ylim(-2, 4)
plt.show()
```

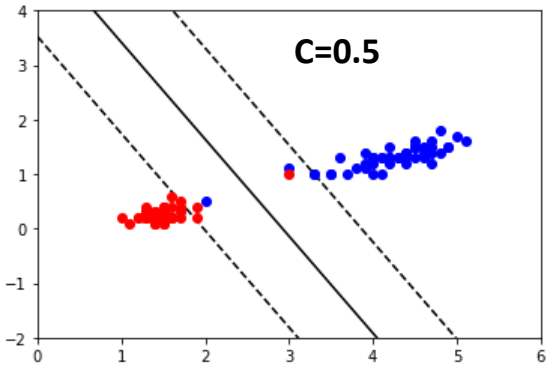
해당 SVC training 결과를 그래프로 확인한 결과는 다음과 같다.

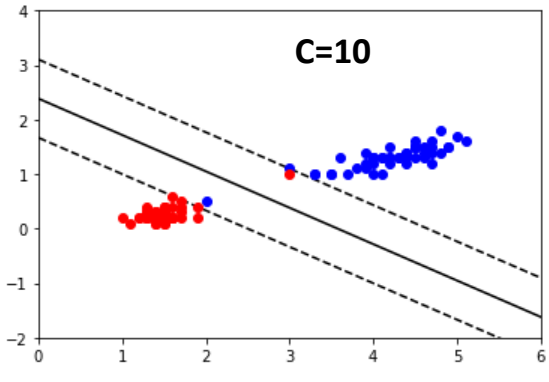




왼쪽 그래프는 앞에서 확인했던 Hard Margin의 결과이다. 관측치를 임의로 변화시켜서 margin 안쪽으로 옮겼음에도 SVC 모델 구성이 가능한 것을 확인할 수 있었다. 앞의 식에서도 알 수 있듯이 C에 대한 수식 Term에 의해 허용이 가능해진 것이다.

이번에는 margin을 조절할 수 있는 하이퍼파라미터인 C를 변화시키면서 margin이 어떻게 변화되는지 확인해 볼 것이다. 해당 실험에서는 c를 0.5와 10으로 진행하였다.





하이퍼파라미터 C는 margin 폭을 줄이거나 넓히는 역할을 한다는 것을 위의 결과를 통해 알 수 있다. 그리고 오른쪽 그래프를 통해 C가 클수록 라그랑지안 문제에서 ξ_i의 역할이 커지면서 margin이 작아지는 것을 알 수 있다. 반대로 C가 작아지면 margin이 커지는 것을 알 수 있다.

◆ Soft Margin – 선형 분리가 불가능한 경우

이번 코드 구현에서는 선형 분리가 가능한 경우에 대해서만 확인해 보았다. 앞서 논문에서도 확인할 수 있듯이 XOR 처럼 선형으로 분리가 안되는 경우는 비선형으로 구분해야 하는데, 이때는 kernel을 'rbf'나 가우시안 커널로 변경하여 커널 트릭을 통해 구분할 수 있도록 해주면 된다.

```
SVC(kernel="rbf", C=1, gamma = 0.1)
```

여기서 C 파라미터는 여유변수에 관한 파라미터이고, gamma는 커널 자체에 대한 파라미터로, Gaussian sphere 같은, 훈련 샘플의 영향력을 증가시키는 등의 역할을 하는 파라미터 계수이다. gamma를 증가시키면 일반적으로 훈련 샘플의 영향력이 증가하고, 결정경계가 더욱 부드러워지게 된다.