수학 2팀

이선민, (김현우)

목차

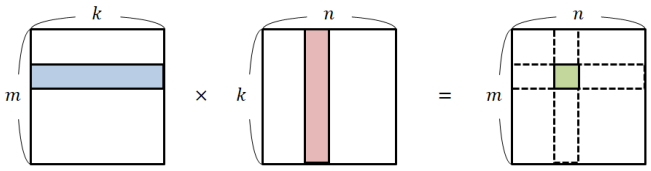
---

1. 행렬의 기하학적 의미

1. 행렬곱의 의미

(1) 행벡터와 열벡터의 내적

우리가 아는 행렬곱의 과정에 대해서 잘 살펴보자. 행렬곱은 하나의 행벡터와 열벡터의 내적 (in
ner product)으로 이루어진다. 아래 수식을 보면 잘 이해가 될 것이다.



$$\begin{bmatrix}1 & 2\\3 & 4\end{bmatrix}\begin{bmatrix}a & b\\c & d\end{bmatrix}=\begin{bmatrix}1\cdot a+2\cdot c & 1\cdot b+2\cdot d\\3\cdot a+4\cdot c & 3\cdot b+4\cdot d\end{bmatrix} \quad (1)$$

위 행렬곱에서 1행 1열의 원솟값을 살펴보자. 이는 두 행렬이 계산되기 전 **1) 왼쪽 행렬의 1행의 행벡터와 2) 오른쪽 행렬의 1열의 열벡터의 곱**으로 이루어진 것임을 알 수 있다.

$$\begin{bmatrix}1 & 2\end{bmatrix}\begin{bmatrix}a\\c\end{bmatrix}=1\cdot a+2\cdot c \quad (2)$$

즉 두 행렬의 곱은 **행벡터와 열벡터의 내적(곱)**으로 이해할 수 있는 것인데, 행벡터와 열벡터의 내적을 함수의 관점에서 이해해 볼 수 있다.

열벡터 $[a,\ c]^T$를 '변화의 대상'으로, 행벡터 $[1,\ 2]$를 '변화 시키는 행위자'로 생각해봤을 때, 위의 수식 (2)는 다음과 같이 함수의 표기로 나타낼 수 있다.

$$\begin{bmatrix}1 & 2\end{bmatrix}\left(\begin{bmatrix}a\\c\end{bmatrix}\right) = 1\cdot a\ +\ 2\cdot\ c \qquad (3)$$

즉, 행벡터는 열벡터를 입력으로 받아 스칼라(상수)로 출력하는 함수인 것이다.

(2) 열벡터의 선형 결합

또 다른 방식으로 행렬의 곱을 이해하는 방법은 **열벡터의 선형 결합**으로 이해하는 방법이다.

행렬과 행렬의 곱에 대해 생각했던 위 예시 수식(1)과 달리, 이번에는 행렬과 벡터의 곱에 대해 생각해보자.

$$\begin{bmatrix}1 & 2\\3 & 4\end{bmatrix}\begin{bmatrix}a\\c\end{bmatrix} = \begin{bmatrix}1\cdot a + 2\cdot c\\3\cdot a + 4\cdot c\end{bmatrix}$$

오른쪽 결과를 다시 나타내면 아래와 같다.

$$\begin{bmatrix}1 & 2\\3 & 4\end{bmatrix}\begin{bmatrix}a\\c\end{bmatrix} = a\cdot\begin{bmatrix}1\\3\end{bmatrix} + c\cdot\begin{bmatrix}2\\4\end{bmatrix}$$

즉, 행렬과 벡터의 곱은 행렬을 구성하고 있는 열벡터 $[1,\ 3]^T$와 $[2,\ 4]^T$ 의 **선형 결합**으로 이루어진 것으로 볼 수 있는 것이다.

이를 정리하면, 행렬곱은 행벡터와 열벡터의 내적이자 열벡터의 선형 결합으로 이해할 수 있는 것!

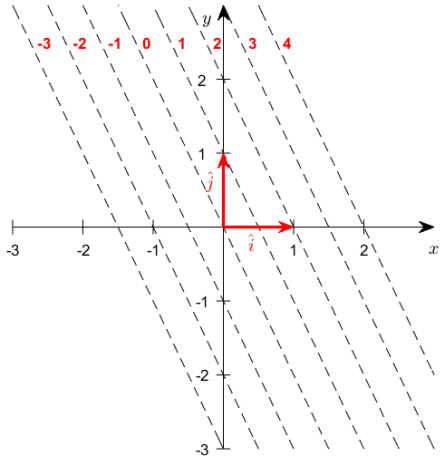
2. 행벡터의 시각화

위에서 우리는 행벡터는 열벡터를 입력으로 받고 스칼라를 출력하는 함수라고 정의하였다. 그렇다면, 이 함수로서의 행벡터를 시각화해보기로 하겠다.

예를 들어, $\begin{bmatrix}2 & 1\end{bmatrix}$ 라는 행벡터를 시각화한다고 하면, 임의의 벡터 $\begin{bmatrix}x & y\end{bmatrix}^T$ 에 대해 함수의 출력을 좌표계에 나열함으로써 행벡터 $\begin{bmatrix}2 & 1\end{bmatrix}$ 를 시각화 할 수 있을 것이다.

$$\begin{bmatrix}2 & 1\end{bmatrix}\left(\begin{bmatrix}x\\y\end{bmatrix}\right) = 2x + y$$

즉, 위와 같은 함수에 대한 여러가지 출력 값들을 x, y 평면에 나타낼 수 있을 것. 예를 들어, $2x + y = 1$ 을 만족하는 $x,\ y$ 쌍을 나타내면 $y = -2x + 1$이라는 선 위에 모두 표시할 수 있을 것이다. 즉, 동일한 스칼라 출력값을 갖는 x, y 순서쌍들을 하나의 선으로 연결할 수 해볼 수 있다. 이러한 시각화를 하는 이유는? 벡터 $\vec{v}$ 가 $\begin{bmatrix}2 & 1\end{bmatrix}$ 이라는 함수를 통과했을 때의 출력값을 눈으로 쉽게 알아볼 수 있기 때문!



위 그림은 $2x + y$ 의 출력값이 각각 -3부터 4에 해당되는 $x,\ y$순서쌍들을 한선에 표시한 것을 확인할 수 있다.

3. 행벡터와 벡터의 내적

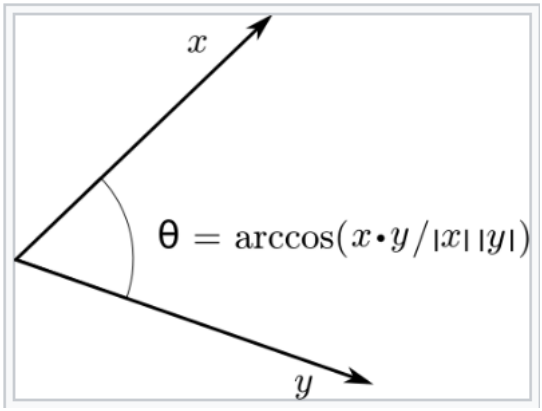
여기서 벡터 간의 내적이 갖는 기하학적 의미 전에, 내적 자체가 갖는 기하학적 의미에 대해 이해해보자.

(1). 내적의 기하학적 의미

유클리디안 공간에서, 유클리디안 벡터는 기하학적 객체로 크기와 방향을 모두 가진다. 한 벡터는 화살표처럼 생겼는데, 그것의 크기는 길이가 되고, 방향은 화살표의 방향이 된다. 벡터 x 의 크기는 $|x|$ 라고 표현하고, 이때 두 벡터 $x,\ y$ 에 대한 내적은 다음과 같이 정의된다.

$$x \cdot y = |x|\ |y| \cos\theta$$

여기서 세타는 두 벡터의 사잇각이다.



정리하자면 한 벡터를 다른 벡터위로 정사영 시킨 길이와 그 다른 벡터의 길이의 곱셈으로 이해할 수 있는 것.

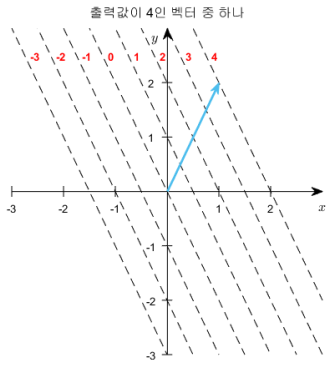
(2). 벡터 간의 내적의 기하학적 의미

이제 우리는 임의의 두 벡터 $\vec{v}_1$ 와 $\vec{v}_2$ 에 대해 두 벡터의 사잇각이 θ 라면 두 벡터의 내적이 다음과 같이 계산된다는 것을 알고 있다.

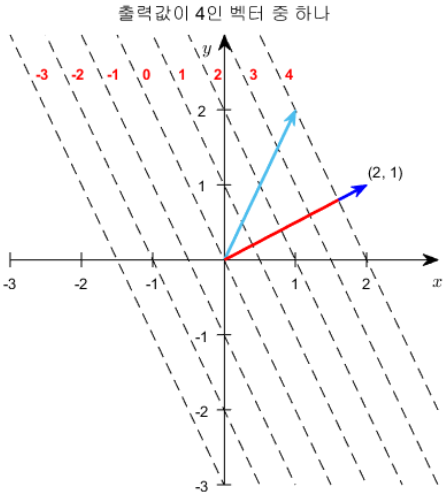
$$\vec{v}_1 \cdot \vec{v}_2 = |\vec{v}_1||\vec{v}_2| \cos\theta$$

위 식을 통해 가져가야할 점은 $|\vec{v}_2| \cos\theta$ **는** $\vec{v}_2$ **를** $\vec{v}_1$ **방향으로 정사영시킨 것**이라는 사실이다.

예를 들어, $2x + y$ 의 출력값이 4가 되게 하는 경우를 생각해보자. 이 때, 출력 스칼라 값이 4가 되게 하는 임의의 벡터를 하나 그려보면 다음과 같다.



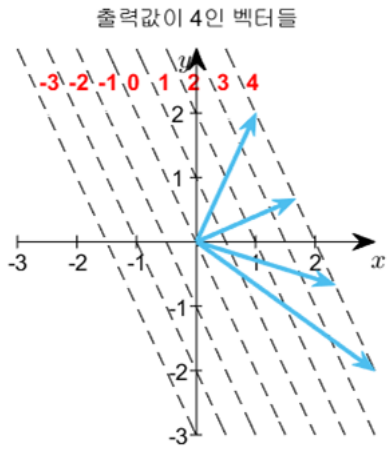
그러고나서, 행 벡터에 해당하는 $[2,\ 1]$ 을 그려본 뒤, $2x + y = 4$ 라는 점까지의 거리를 생각해보자.



하늘색 벡터를 $\vec{v}_1$ 으로, 파란색 실선으로 표현된 행벡터 $[2,\ 1]$ 를 $\vec{v}_2$ 으로 생각해보자. 이때 빨간색 실선은 **1) 벡터 $\vec{v}_1$ 를 벡터 $\vec{v}_2$ 위에 정사영시킨 벡터**이자 **2) $2x + y = 4$ 라는 점까지의 거리**를 의미한다.

빨간색 실선의 길이는 직각 삼가형의 넓이를 구하는 방법을 이용해 구할 수 있는데, 자세한 내용은 reference 를 참고하자.

아무튼 벡터 $\vec{v}_1$ 와 벡터 $\vec{v}_2$ 의 내적은 열벡터 $\vec{v}_1$ 의 정사영 길이 (빨간색 실선)과 행벡터 $\vec{v}_2$ 의 길이와의 곱과 같다는 결론이 나오게 된다. 이때, 두 벡터의 내적을 계산할 때에는 $2x + y = 4$ 의 점선 위에 있는 어떤 벡터를 고르더라도, 행벡터 $[2,\ 1]$ 과의 내적 계산은 $2x + y$ 의 출력 스칼라값인 4가 되는 것을 알 수 있다. 다시 말해, 열벡터의 정사영 길이가 내적 계산에 이용되기 때문임을 기하학적으로 확인할 수 있는 것! 이를 그림으로 정리해보자.



위 그림은 함수로써의 행벡터 $[2,\ 1]$ 의 출력 스칼라 값이 4일 때, 4가 나오게 하는 다양한 열벡터들의 예시를 나타내었다. 이렇게 스칼라 값 4를 출력하는 다양한 열벡터들은, 행벡터 $[2,\ 1]$ 에 대해 모두 동일한 정사영 벡터(이전 그림에서 빨간색 실선)를 가지므로, 어떠한 열벡터들과 행벡터 $[2,\ 1]$ 의 내적은 모두 동일한 결과인 4를 갖게 된다는 것이다.

2. 기저(Basis)

1. 기저란

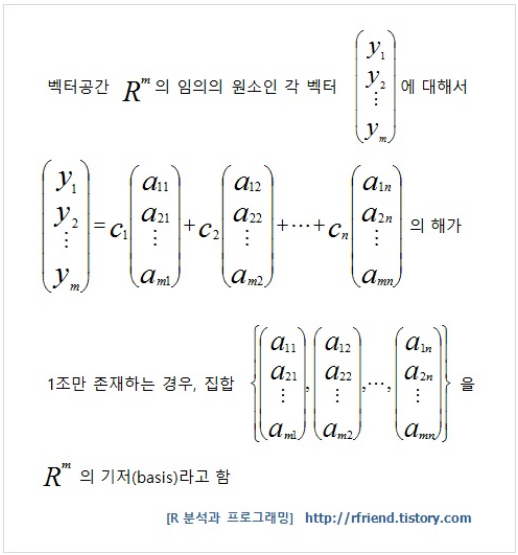
기저란 벡터들의 집합인 벡터 공간 V에 대해 그 V를 span 하는 가장 작은 (minimal) 집합을 말한다.

> 💡 span이란
> 어떤 벡터 공간 V가 있을 때 V의 원소들의 선형 결합들에 의해 만들어진 공간을 의미한다. 즉 V = $\{v_1, v_2, ..., v_n\}$ 일 때 임의의 실수 $a_1, a_2, ..., a_n$ 에 대하여 $a_1v_1, a_2v_2, ..., a_nv_n$ 이 만들어내는 공간을 span $\{v_1, v_2, ..., v_n\}$ 이라고 하는 것.
> 다시 말해 span{V}는 벡터 공간 V의 원소들의 선형 결합으로 형성되는 또 다른 벡터 공간인 셈이다.

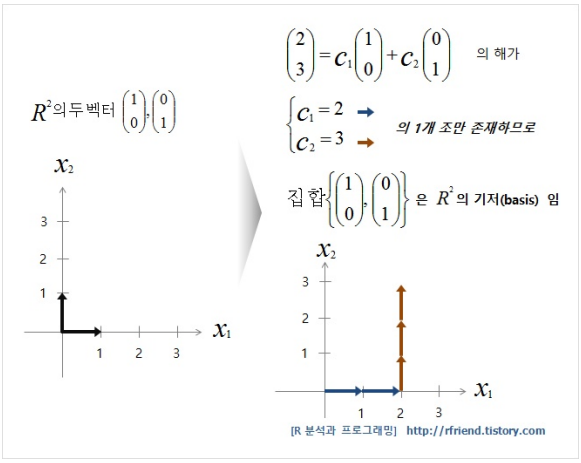
두 개의 벡터는 그들이 독립이라 하더라도 $\mathbb{R}^3$ 전체를 생성할 수는 없다. 네 개의 벡터는 $\mathbb{R}^3$ 을 생성한다하더라도 선형 독립일 수 없다. 우리는 공간을 생성하기 위한 충분한 수의 독립된 벡터를 원하되, 그 이상을 원하지 않는다. 기저는 이러한 요구에 정확히 부합하는 개념이다.

기저가 되려면 다음과 같은 조건을 만족시켜야 한다.

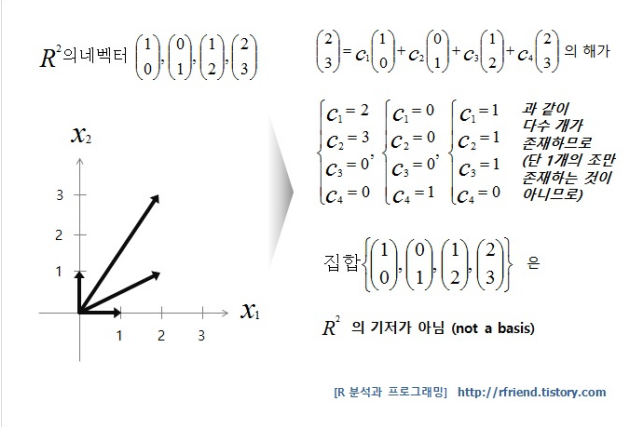
- 기저에 속한 벡터들이 선형 독립
- 기저에 속한 벡터들이 벡터 공간 V를 생성



다음 그림을 통해 기저인 경우와 기저가 아닌 경우를 구분해보자.

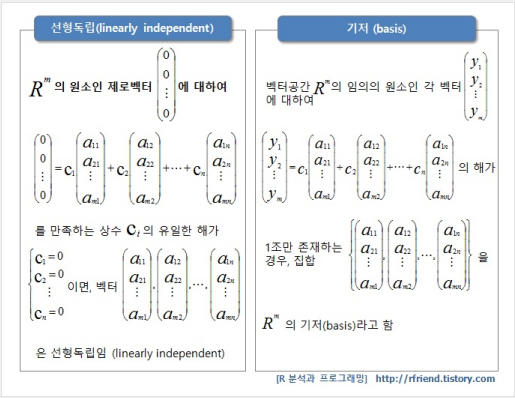


벡터 공간 $\mathbb{R}^2$ 의 임의의 원소인 벡터 $\begin{bmatrix} 2 \\ 3 \end{bmatrix}$ 에 대해 $\begin{bmatrix} 2 \\ 3 \end{bmatrix} = c_1 \begin{bmatrix} 1 \\ 0 \end{bmatrix} + c_2 \begin{bmatrix} 0 \\ 1 \end{bmatrix}$ 을 만족하는 해가 단 한 개만 존재하므로, 집합 $\{ \begin{bmatrix} 1 \\ 0 \end{bmatrix}, \begin{bmatrix} 0 \\ 1 \end{bmatrix} \}$은 $\mathbb{R}^2$ 의 기저라고 할 수 있다.

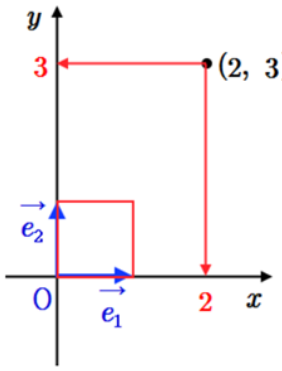


위 그림은 기저의 조건을 만족하지 못하는 경우.

💡 선형 독립 vs 기저
선형 독립은 기저의 개념이 제로벡터로 한정된 개념. 기저는 $\mathbb{R}^n$ 의 모든 벡터를 대상으로 하는 개념.



지금까지 기저에 대해 개념적으로 알아보았는데, 기저의 개념을 그래프 상에서 이해해보겠다.

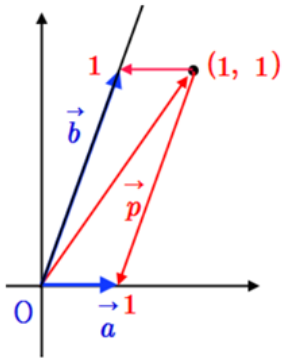


위 예시에서 P(2, 3)이란 좌표를 말할 때는 어떤 과정이 내포되어있다. 먼저 원점 O를 두고 원점을 지나는 서로 수직인 두 개의 축을 생각했을 때, 우리는 흔히 x축 y축이라고 말한다. 점 P의 좌표가 (2, 3)이라는 것은 점 P에서 x축에 내린 수선의 발의 눈금을 읽으면 2이고, y축에 내린 수선의 발의 눈금은 3이라는 뜻이다.

여기서 말한 눈금 2와 3을 읽기 위해서는 먼저 읽는 단위를 먼저 결정해야하는데, 이 단위를 결정하는 것이 바로 기저(basis)인 것이다. 위 그림의 좌표평면에서 원점 O에서 (1, 0)으로 가는 벡터를 $\vec{e}_1$, (0, 1)로 가는 벡터를 $\vec{e}_2$ 로 정하고, 이를 기준으로 해서 좌표 P(2, 3)을 $\vec{p} = 2\vec{e}_1 + 3\vec{e}_2$ 로 표시한다. 즉, 벡터 $\vec{p}$ 는 **두 기저 벡터의 선형 결합**으로 표현되는 것.

크기가 1이고, x축, y축의 양의 방향으로 향하는 벡터 $\vec{e}_1, \vec{e}_2$ 를 기저라고 하면 우리가 알고 있는 좌표가 벡터와 일대일 대응된다. 이 때 우리가 흔히 아는 일반적인 표준 좌표계에서의 좌표라고 해서, $\vec{e}_1, \vec{e}_2$ 를 **표준 기저**라 한다.

물론, 기저를 위의 그림과 같이 $\vec{e}_1, \vec{e}_2$ 로 잡을 필요는 없다. 우리가 흔히 아는 좌표평면에 평행/수직하지 않고 길이도 1이 아닌 두 벡터를 잡아도, 여전히 기저가 될 수 있다.



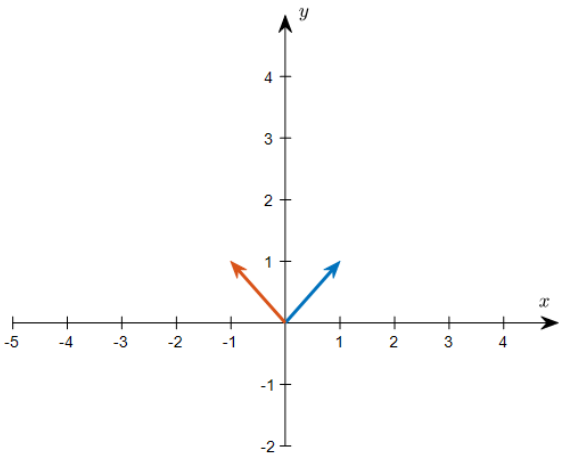
위의 그림에서, $\vec{a}$ = $(1, 0)$ 와 $\vec{b}$ = $(1, 3)$ 를 기저로 잡으면, 앞서 $2\vec{e}_1 + 3\vec{e}_2$ 로 표현되었던 $\vec{p}$ 가 이제는 $1\vec{a} + 1\vec{b}$ 로 표현되는 것이다. 즉 점 P는 a, b 의 측면에서 $(1, 1)$ 을 가지는 것인데, 이 $(1, 1)$ 을 $\vec{a} = (1, 0)$ 와 $\vec{b} = (1, 3)$ 에 대한 **상대적 좌표**라고 한다. 여기서 기저의 변환이 나오는데 이는 아래에서 다루도록 하겠다.

2. 기저의 변환

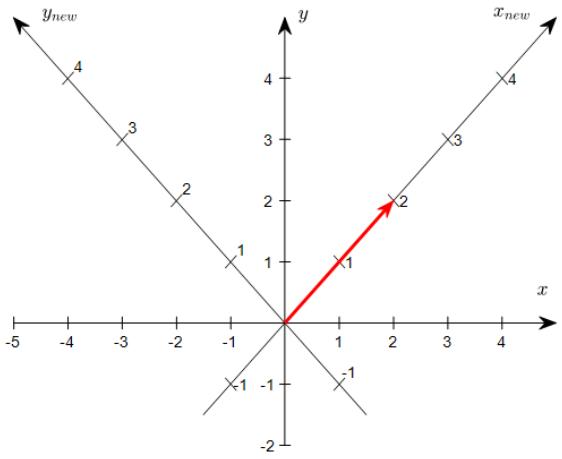
표준 기저가 아닌 전혀 새로운 기저를 이용해 좌표계를 구성할 수 있는데, 새로운 좌표계를 이용하여 임의의 벡터를 어떻게 다시 표현할 수 있을까?

아래와 같은 새로운 기저 벡터들의 집합을 생각해보자.

$$\mathcal{B} = \left\{\vec{b_1}, \vec{b_2}\right\} = \left\{\begin{bmatrix}1\\1\end{bmatrix}, \begin{bmatrix}-1\\1\end{bmatrix}\right\}$$



위 그림은 새로운 기저 벡터를 표현한 것인데, 이 기저 집합 $\mathcal{B}$ 를 이용한 새로운 좌표계는 아래 그림과 같다.



표준 기저에서는 $(2,2)$ 로 표현되던 빨간색 좌표가 새로운 좌표계에서는 $(2,0)$ 으로 관찰되는 것을 볼 수 있다.

이렇게 기저가 새로운 좌표계에 의해 변경되는 경우 표기는 어떻게 할 수 있을까? 기저가 변경되는 경우, 벡터를 표현할 때 아랫첨자로 기저가 되는 벡터 집합의 이름을 기재할 수 있다.

예를 들어, 표준 기저에서 표현되는 $(2,2)$ 벡터는

$$\begin{bmatrix}2\\2\end{bmatrix}_{\mathcal{E}}$$

라고 표현하고, 새로운 기저 $\mathcal{B}$ 를 기반으로 하여 표현되는 벡터 $(2,0)$ 은

$$\begin{bmatrix}2\\0\end{bmatrix}_{\mathcal{B}}$$

라고 표현된다.

$$\begin{bmatrix}2\\2\end{bmatrix}_{\mathcal{E}}, \begin{bmatrix}2\\0\end{bmatrix}_{\mathcal{B}}$$

위에서 보았듯이, 두 벡터는 표현만 다를 뿐 동일한 벡터이다. 그렇다면 위 식은 어떻게 해석될 수 있을까? 위 식은 표준좌표계에서는 $(2,2)$ 의 값을 가지던 좌표가 $\mathcal{B}$ 라는 기저를 가지고 만든 새로운 좌표계에서 $(2,0)$ 로 표현될 수 있다.

즉, 표준 좌표계에서의 좌표인 $(2,2)$ 가 $\mathcal{B}$ 의 기저들을 이용하여 나타낼 수 있다는 뜻인데, 이를 관계식으로 나타내보면 다음과 같다.

$$\begin{bmatrix}2\\2\end{bmatrix} = k_1\begin{bmatrix}|\\b_1\\|\end{bmatrix} + k_2\begin{bmatrix}|\\b_2\\|\end{bmatrix} \tag{4}$$

표준 좌표계의 $(2,2)$ 를 기저 $\mathcal{B}$ 를 이용해 표현되는 좌표인 (k_1,k_2) 로 나타낼 수 있다는 뜻이다.

우리는 앞서 행렬 곱에 대한 기하학적 의미 파트에서 행렬과 벡터의 곱이 '**열벡터의 선형 결합**'으로 해석할 수 있다는 것을 알아보았다. 이러한 관점에서 위의 식을 아래와 같이 다시 쓸 수 있다.

$$\begin{bmatrix}2\\2\end{bmatrix} = \begin{bmatrix}| & |\\b_1 & b_2\\| & |\end{bmatrix}\begin{bmatrix}k_1\\k_2\end{bmatrix}$$

그리고, 우리에게 주어진 $\mathcal{B}$ 의 기저는 아래와 같으므로,

$$\mathcal{B} = \left\{ \begin{bmatrix} 1 \\ 1 \end{bmatrix}, \begin{bmatrix} -1 \\ 1 \end{bmatrix} \right\}$$

(4)번 식을 다시 써보면

$$\begin{bmatrix} 2 \\ 2 \end{bmatrix} = \begin{bmatrix} 1 & -1 \\ 1 & 1 \end{bmatrix} \begin{bmatrix} k_1 \\ k_2 \end{bmatrix} \tag{5}$$

이고, 위 관계식을 만족하는 k_1, k_2 를 구하면 앞서 새로운 기저 $\mathcal{B}$ 에서 얻은 좌표였던, $(2, 0)$ 이라는 답을 얻을 수 있는 것이다.

$$\therefore \begin{bmatrix} k_1 \\ k_2 \end{bmatrix} = \begin{bmatrix} 1 & -1 \\ 1 & 1 \end{bmatrix}^{-1} \begin{bmatrix} 2 \\ 2 \end{bmatrix} = \begin{bmatrix} 2 \\ 0 \end{bmatrix} \tag{6}$$

즉, 새로운 기저는 기존의 기저의 **선형 결합** 형태로 표현한 것.

3. SVM

1. Support Vector Machine 이란

(1) Support Vector Machine

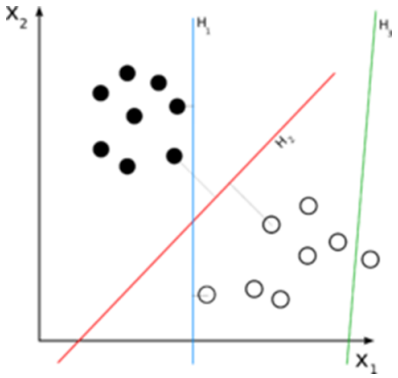
서포트 벡터 머신(SVM)은 분류 과제를 수행하는 지도학습 중 하나로, 분류를 위한 기준선, 즉 **결정 경계(Decision Boundary)를** 정의하는 모델이다. 두 카테고리 중 어느 하나에 속한 데이터의 집합이 주어졌을 때, SVM 알고리즘은 주어진 데이터 집합을 바탕으로 하여 새로운 데이터가 어느 카테고리에 속할지 판단하는 분류 모델을 만든다. Decision boundary의 형태에 따라 선형 모델, 비선형모델로 구분할 수 있는데, 유클리디안 평면에서 선형 모델을 가지고 설명하도록 하겠다.

(2). 목표 및 아이디어

SVM은 p 차원의 벡터(즉, p 개의 숫자 리스트)가 주어졌을 때, 이러한 데이터 점들을 $(p-1)$ 차원의 초평면으로 분류할 수 있는지를 확인하고 싶은 것이다. 데이터를 분류하는 초평면은 여러가지가 나올 수 있는데, 수많은 초평면들 중에서 가장 타당한 초평면을 선택하는 기준은 무엇일까? 결론부터 얘기하자면, 두 클래스 사이에서 가장 큰 분류 또는 마진(margin, 즉 두 클래스 간의 차이)을 가지는 초평면을 선택하는 것이다. 따라서 우리는 **초평면에서 가장 가까운 각 클래스의 데이터 점들 간의 거리가 최대**가 되는 초평면을 선택한다.

> 💡 초평면(Hyperplane)이란?
> 초평면이란 어떤 N차원 공간에서 한 차원 낮은 N-1 차원의 subspace를 말한다. 예를 들어, 3차원의 경우 2차원인 면이 되는 것이고, 2차원의 경우 선이 되는 것이다.

SVM의 목표를 그림을 통해 이해해보자.

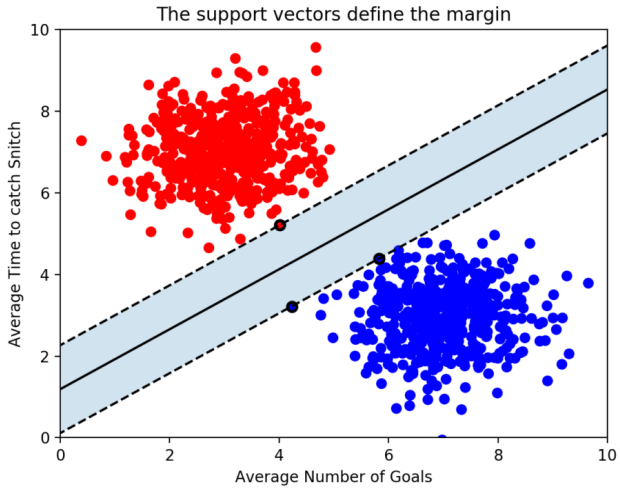


다음과 같은 그림이 있을 때, 결정 경계의 후보가 되는 세 개의 초평면(해당 예시에서는 1차원 직선)을 확인할 수 있다. 가장 먼저 H_3 은 두 클래스를 제대로 분류하지 못하므로 결정 경계가 될 수 없다. 그렇다면 앞서 설명한 초평면을 선택하는 기준에 맞춰 H_1 과 H_2 중 적절한 결정 경계를 선택해보도록 하겠다. 두 직선 모두 클래스의 점들을 잘 분류하고 있지만, H_2 가 H_1 보다 더 큰 마진을 갖고 분류하는 것을 확인할 수 있다. 즉, 각 초평면에서 가장 가까운 클래스의 점들이 결정 경계를 정의하는 결정적인 역할을 하게되는 것을 확인할 수 있다.

(3). 서포트 벡터와 마진

정리해보자면, 최적의 결정 경계는 데이터 군으로부터 최대한 멀리 떨어져야 하며, 각 결정 경계와 가까이 있는 데이터 포인트들을 **서포트 벡터(Support Vector)**라고 한다.

결정 경계와 서포트 벡터 사이의 거리를 **마진(Margin)**이라고 한다. 각 개념들을 아래 그림을 통해 확인해보자.

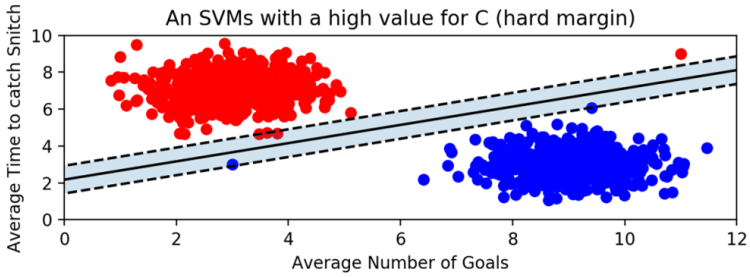


가운데 실선이 결정 경계가 되며, 검은 점선 위에 있는 각 클래스의 점들이 서포트 벡터가 된다. 그리고 하늘색 영역이 마진이 되는 것.

서포트 벡터의 역할에서 SVM의 장점을 볼 수 있는데, SVM에서는 결정 경계를 정의하는 게 결국 서포트 벡터이므로, 여러 데이터 포인트들 중에서 소수의 서포트 벡터만 잘 골라내면 분류를 효율적으로 진행할 수 있는 것.

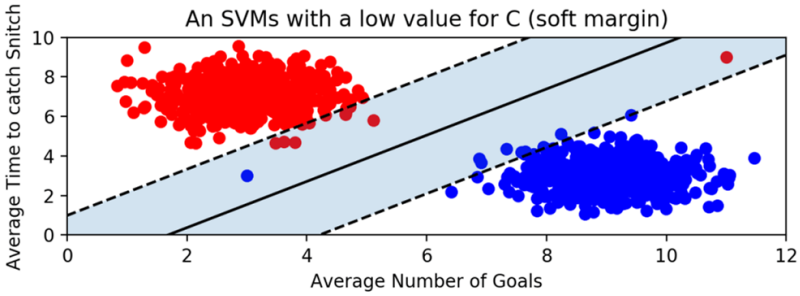
(4). Outlier

우리는 앞서 SVM의 결정 경계가 소수의 서포트 벡터에 의해 결정되는 것을 확인했다. 그러나 결정 경계를 결정해야하는 서포트 벡터가 이상치일 경우 분류가 제대로 되었다고 할 수 있을까?



위 그림에서 왼쪽에 혼자 튀어 있는 파란 점과 오른쪽 상단에 빨간 점은 누가 봐도 아웃라이어다. 즉 SVM은 이러한 아웃라이어들을 잘 다루는 게 관건. 위 그림은 아웃라이어 즉 잘못 분류되는 것을 허용하지 않고 기준을 까다롭게 세운 경우이다. 이를 **하드 마진(Hard Margin)**이라고 하는데, 서포트 벡터와 결정 경계 사이의 거리가 매우 좁은 것을 확인할 수 있다. 이렇게 개별적인 학습 데이터들을 놓치지 않고 아웃라이어를 허용하지 않는 기준으로 결정 경계를 정해버리면 오버피팅 문제가 발생할 수 있다.

반대의 경우도 살펴보자.

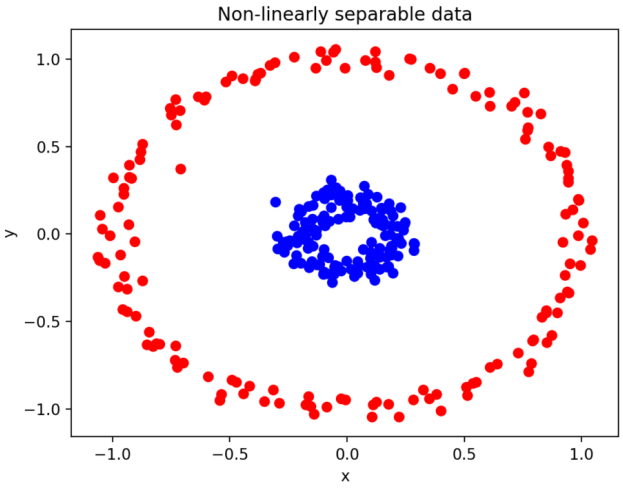


아래 그림은 각 클래스의 아웃라이어들을 마진 안에 포함되도록 너그럽게 기준을 잡았다. 이를 **소프트 마진(Soft Margin)**이라고 한다. 하드 마진보다 서포트 벡터와 결정 경계 사이의 거리, 즉 마진이 커진 것을 확인할 수 있다. 이 경우 언더피팅 문제가 발생할 수 있다. 소프트 마진에 대한 자세한 내용은 reference 참고!

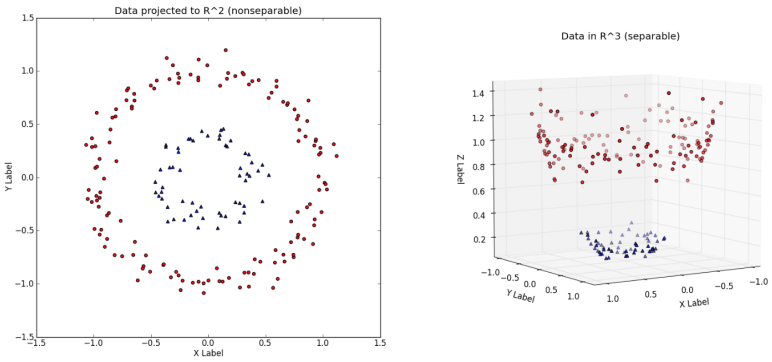
2. Kernel SVM

(1). Kernel SVM의 원리

지금까지는 SVM의 선형 이진 분류 모델을 예시로 설명했는데, 만약 선형으로 분리되지 않는 데이터 셋이 있다면 어떻게 해야 할까?



위 데이터 셋이 주어졌을 경우 직선 형태의 결정 경계로는 빨간색 점과 파란색 점을 구분할 수 없을 것이다.



커널 기법의 기본적인 아이디어는 **저차원의 데이터를 높은 차원으로 이동시켜 고차원 공간에서 데이터를 분류**하고자 함이다. 예를 들어, 분류가 어려운 2차원의 데이터 셋에 비선형 특성을 추가하여 3차원으로 옮김으로써 2차원의 평면 경계로 3차원 데이터 셋을 구분하는 것이다.

(2). Kernel

커널은 연속함수 $K(x, y)$ 로 실수, 함수나 벡터 등을 x, y 인자로 받고 실수를 내뱉는 symmetric 함수이다. 저차원의 데이터를 고차원으로 mapping 시킬 때 발생하는 연산 비용 문제나, 어떤 특성을 추가해야할 지 모를 때 커널 기법을 사용할 수 있다.

커널은 결국 두 벡터의 내적이며, 기하학적으로는 코사인 유사도를 의미하기 때문에 유사도 함수(Similarity Function)이라고도 불린다. 즉, 두 표본의 **유사도 측정의 기준**으로 볼 수 있는 것!

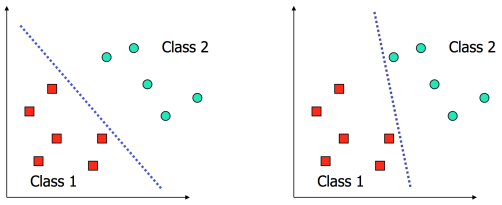
커널의 종류는 가우시안 커널, 시그모이드 커널 등 여러 가지가 있는데, 이는 reference를 참고하도록 하자.

3. SVM 수식으로 이해하기

(1). Decision Boundary

앞선 결정 경계를 수식으로 정의해보자. 결정 경계(결정 초평면)를 정의하기 위해서는 결정 초평면에 수직하는 가중치 벡터 $\vec{w}$ 와 상수 b 가 필요하다.

n차원의 평면의 방정식은 어떻게 정의할까? 2차원을 예로 들어 보자. 직선에 수직한 벡터 $\vec{w} = (a, b)$ 에 대한 직선의 방정식은 $ax + by + c = 0$ 이다.



다시, 위에 나타나 있는 결정 경계를 $\vec{w}^T\vec{x} + b = 1$ 로 나타내 보겠습니다. 이 직선보다 위에 있으면 class 2, 아래에 있으면 class 1으로 분류할 수 있는데, 이를 수식으로 나타내면 아래와 같다.

class 1을 negative x_-, class 2를 positive x_+ 이라고 보면 각 데이터는

$$y_i = +1 \; when \; \vec{w}\vec{x_+} + b \geq 1$$
$$y_i = -1 \; when \; \vec{w}\vec{x_-} + b \leq -1$$

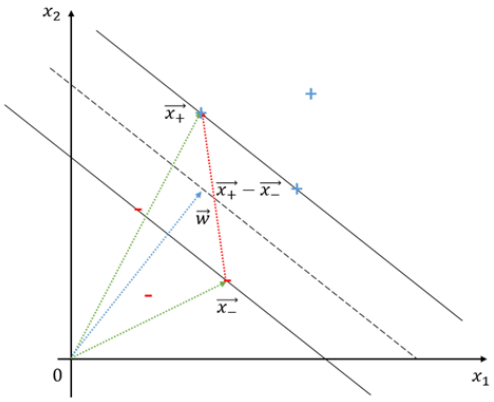
로 표현할 수 있고, 위 두 식을 하나로 합치면

$$y_i(\vec{w} \cdot \vec{x}_i + b) \geq 1 \qquad (7)$$

로 나타낼 수 있다.

(2). Margin

이번에는 마진을 최대화하는 수식을 살펴보자. 마진은 두 점선 사이의 거리라고 했는데, 즉 가장 가까워 보이는 각 클래스 점들 사이의 거리이다. 그렇다면 두 점 사이의 거리 공식을 이용해서 구할 수 있는 것인데, 두 점선 사이의 거리인 마진은



$$\frac{\vec{w}}{|\vec{w}|} \cdot (\vec{x}_+ - \vec{x}_-) \qquad (8)$$

이다. 이 때 x_i 가 positive sample 이라면 $y_i = 1$ 이고 x_i 가 negative sample 이라면 $y_i = -1$ 이기 때문에

$$\vec{w} \cdot \vec{x} + b - 1 = 0$$
$$\Rightarrow \vec{w} \cdot \vec{x}_i = 1 - b$$

이고,

$$\vec{w} \cdot \vec{x} + b + 1 = 0$$
$$\Rightarrow \vec{w} \cdot \vec{x}_i = -1 - b$$

이다. 위 관계식을 식 (7)에 대입해보면,

$$\begin{aligned} \frac{\vec{w}}{|\vec{w}|} \cdot (\vec{x}_+ - \vec{x}_-) \Rightarrow \frac{1}{|\vec{w}|}(\vec{w} \cdot \vec{x}_+ - \vec{w} \cdot \vec{x}_-) \\ = \frac{1}{|\vec{w}|}(1 - b + 1 + b) \\ = \frac{2}{|w|} \end{aligned}$$

이다.

식 (7)과 식 (8)은 모두 마진을 표현한 수식인데, 둘의 차이가 존재한다.

식 (7)의 경우 수식 계산을 통해 특정 데이터 포인트 $\vec{x_i}$ 의 클래스를 분류해준다. 그러나 특정 포인트가 어떤 클래스에 속하는지에 대한 정보만 전달할 뿐, 결정 경계와 얼마나 가까운지 또는 멀리 있는지에 대한 정보는 포함하고 있지 않다. 이렇게 특정 데이터 포인트의 클래스만 분류하는 마진을 **Functional Margin**이라고 한다.

반면 식 (8)의 경우 해당 포인트가 어떤 클래스에 속하는지 뿐만 아니라 결정 경계와 얼마나 멀리 떨어져 있는지에 대한 지리적 정보를 전달한다. 이를 **Geometric Margin** 이라고 한다.

즉, 두 실선 사이의 거리는 $\frac{2}{|w|}$ 이고 우리의 목표는 마진을 최대화하는 것이므로 $\frac{2}{|w|}$ 을 **최대화**하는 것이라고 할 수 있다. 이는 다시 말하면 $|w|$ 을 **최소화**하는 것이며, 수학적인 편의를 위해 이것을

$$min\frac{1}{2}|\vec{w}|^2$$

로 나타낼 수 있을 것이다.

(3). Margin의 Optimization

마진을 최대화하는 문제는 결국 최적화 문제로 해석될 수 있다. (이 때 미분 계산의 편리성을 위해 $\frac{2}{|w|}$ 을 $min\frac{1}{2}|\vec{w}|^2$ 으로 나타낸 것이다.)

조건은

$$y_i(\vec{w}\cdot\vec{x}_i+b)-1=0$$

가 되고 목적함수는

$$\frac{1}{2}|\vec{w}|^2$$

가 되는 것.

이를 라그랑주 승수법에 대입해보면 다음과 같은 과정을 거치게 된다. 우선 보조방정식 L 을 세우고,

$$L=\frac{1}{2}|\vec{w}|^2-\sum_i\alpha_i\left[y_i(\vec{w}\cdot\vec{x}_i+b)-1\right]$$

위 보조방정식을 벡터 $\vec{w}$ 에 대해 편미분해보자.

$$\frac{\partial L}{\partial\vec{w}}=\vec{w}-\sum_i\alpha_iy_ix_i=0$$
$$\vec{w}=\sum_i\alpha_iy_i\vec{x}_i \qquad (9)$$

즉, 벡터 $\vec{w}$ 는 몇 개의 $\vec{x_i}$ 들의 **선형 결합**으로 나타낼 수 있는 것이다.

여기서 또 다른 변수들에 대해 편미분을 하고 연립하는 과정을 거치면(자세한 내용은 reference 참고) 우리가 최대화하고자 하는 L 은

$$\sum_i\alpha_i-\frac{1}{2}\sum_i\sum_j\alpha_i\alpha_jy_iy_j\vec{x}_i\cdot\vec{x}_j \qquad (10)$$

로 나타낼 수 있다.

▼ 좀 더 자세한 Margin의 Optimization

앞에서 Functional Margin과 Geometric Margin을 통해 우리는 다음과 같은 최소화할 목적함수를 얻어냈다.

$$min_{w,b}\frac{1}{2}\|w\|_2^2\,,subject\ to\ \ y_i(w^Tx_i-b)\geq 1$$

이를 이번 스터디에 공부한 라그랑주 승수를 이용해서 제약식을 포함하는 최소화할 목적함수를 구할 수 있다.

- Lagrange Primal

제약식을 포함하여 우리가 최소화 할 목적함수는 다음과 같다.

$$L_P=\frac{1}{2}\|w\|^2-\sum_i\alpha_i(y_i(w^Tx_i-b)-1),$$
$$min_{w,b}\ L_P\quad subject\ \ to\ \ \alpha_i\geq 0$$

그러나 이번 스터디에서도 나왔던 것처럼, 이와 같이 라그랑주 승수를 이용해서 제약식을 풀기 위해서는 KKT Condition을 만족시켜야 한다.

이를 위해서 만족시켜야 할 2가지 성질은 다음과 같았고,

$$(1)\ \mu g(x^*)=0$$
$$(2)\ \mu\geq 0$$

이를 좀더 변형하여 SVM에서 만족해야 할 KKT Condition은 다음과 같다.

$$(1)\quad \frac{\partial L_p}{\partial w}=0,\quad \frac{\partial L_p}{\partial b}=0$$
$$(2)\quad \alpha_i\sum_i(y_i(w^Tx_i-b)-1)=0$$
$$(3)\quad \sum_i(y_i(w^Tx_i-b)-1)\geq 0$$
$$(4)\quad \alpha_i\geq 0$$

이를 천천히 살펴보자.
(1)은 최적화하려는 미지수로 편미분해주면 그 값이 0이 된다는 조건이다. 이 조건을 만족시키기 위해서 L_p를 w와 b에 대해 편미분해주면, α에 대한 값을 구해줄 수 있다. 편미분을 진행해주면, 다음과 같은 결과를 얻을 수 있다.

$$1.\quad \frac{\partial L_p}{\partial w}=w-\sum_i\alpha_iy_ix_i=0\Longleftrightarrow w=\sum_i\alpha_iy_ix_i$$
$$2.\quad \frac{\partial L_p}{\partial b}=-\sum_i\alpha_iy_i=0\Longleftrightarrow \sum_i\alpha_iy_i=0$$

1.을 통해 우리는 w가 x_i들의 선형결합으로 이루어져있다고 생각할 수 있고, 2.를통해서는 x_i의 계수들(α_iy_i)들의 합이 0이라는 걸 보여준다. 이때, SVM의 Label $y_i\in\{-1,1\}$이고, $\alpha_i\geq 0$이라는 걸 생각해보면, 우리가 ANOVA에서 Contrast를 만들때와 비슷하다는 생각을 해볼 수 있을것이다.

(2)는 $\mu g(x^*)=0$가 되어야 하기 때문에 도출되고, (4) 또한 $\mu\geq 0$이어야 하기 때문에 도출된다. (이 부분도 수학 1팀 설명 참조)
(3)은 라그랑주 승수 형태로 바꾸기 전에 원래 목적함수의 제약식이었다.

결과적으로 L_P도 이차계획법(Quadratic Programming)의 형태를 띄기 때문에 이 조건 아래에서 L_p를 최소화 할 수 있지만, 이 경우에는 Primal Problem을 Dual Problem으로 변환하여 푸는 것이 더 쉽다.

그렇다면 Dual Problem으로 바꾼다는 것은 무엇을 의미할까? 쉽게 말해서 기존의 Primal problem이 목적함수를 최소화하는 문제였다면, 이를 최대화 하는 문제로 바꿔서 풀 수 있다는 것이다.

우리가 최소화하려고 했던 식을 다시 살펴보자. L의 Parameter는 α,w,b 총 3개였고, 우리의 Primal Problem은 Lagrange Primal L_p를 w,b에 대해서 최소화하는 것이었다. 이를 Dual Problem으로 바꿔준다는 것은, L_p를 α에 대한 함수 L_D로 바꿔주어 이를 α에 대해서 최대화하는 문제로 바꿔준다는 것이다. 이를 수식으로 표현해주면 다음과 같다.

$$min_{w,b}\ L_P\Rightarrow max_{\alpha}\ L_D$$

그러나, 이 또한 문제가 있다. L_P를 최소화하는 것은 L_P가 가질 수 있는 값들의 하한선을 찾는다는 이야기인데, 이를 Duality Problem으로 바꾼다는 말은 L_D를 최대화하는 것, 즉 L_D의 상한선을 찾는다는 이야기로 바꿔서 보겠다는 것이기 때문이다. 따라서, Primal Problem의 해와 Dual Problem의 해가 일치하지 않는다면, Dual Problem을 해결하는 것으로 Primal Problem을 해결할 수 없다는 것이다. 즉 Dual Problem의 최적해가 Primal Problem의 최적해라는 보장을 할 수 없다는 것이다.

따라서, 이를 보장하기 위한, 즉 Strong Duality를 지니기 위한 또다른 조건이 필요하다. 이 조건을 Slater's Condition이라고 하고, 그 내용은 다음과 같다.

> 📖 Slater's Condition
> 1. Primal Problem의 목적함수가 convex하고, 부등식의 제약조건도 convex하며, 등식제약조건이 affine이다.
> 2. 최소한 하나의 Strictly feasible solution, 즉 제약 조건을 강력하게 만족하는 해가 존재한다.

이 경우에는 Primal Problem이 Slater's Condition을 만족하기 때문에, Dual Problem으로 접근하여 해를 구할 수 있게 된다.

그렇다면 다시 돌아와서 KKT Condition을 만족하는 L_P를 다시 살펴보면 다음과 같다.

$$L_P=\frac{1}{2}\|w\|^2-\sum_i\alpha_i(y_i(w^Tx_i-b)-1)$$
$$subject\ \ to$$
$$w=\sum_i\alpha_iy_ix_i,\ \ \sum_i\alpha_iy_i=0$$
$$\alpha_i\geq 0$$

이를 L_D로 바꿔주기 위하여, 제약식에 있는 $w=\sum_i\alpha_iy_ix_i$를 L_P에 대입하여 α에 대한 식으로 바꿔준다.

- Process of Lagrange Primal → Lagrange Dual

$$
\begin{aligned}
(1)\ \frac{1}{2}\|w\|^2 &= \frac{1}{2}w^T w \\
&= \frac{1}{2}w^T \sum_{j=1}^{n} \alpha_j y_j x_j \\
&= \frac{1}{2}\sum_{j=1}^{n} \alpha_j y_j (w^T x_j) \\
&= \frac{1}{2}\sum_{j=1}^{n} \alpha_j y_j (\sum_{i=1}^{n} \alpha_i y_i x_i^T x_j) \\
&= \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j x_i^T x_j
\end{aligned}
$$

$$
\begin{aligned}
(2)\ -\sum_{i=1}^{n} \alpha_i (y_i (w^T x_i - b) - 1) \\
= -\sum_{i=1}^{n} \alpha_i y_i (w^T x_i - b) + \sum_{i=1}^{n} \alpha_i \\
= -\sum_{i=1}^{n} \alpha_i y_i w^T x_i + b\sum_{i=1}^{n} \alpha_i y_i + \sum_{i=1}^{n} \alpha_i \\
= -\sum_{i=1}^{n}\sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j x_i^T x_j + \sum_{i=1}^{n} \alpha_i \ \ (\because \sum_{i=1}^{n} \alpha_i y_i = 0)
\end{aligned}
$$

$$
\begin{aligned}
\therefore \frac{1}{2}\|w\|^2 - \sum_i \alpha_i (y_i (w^T x_i - b) - 1) \\
= \sum_{i=1}^{n} \alpha_i - \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j x_i^T x_j
\end{aligned}
$$

- Lagrange Dual

$$
\begin{aligned}
L_D = \sum_{i=1}^{n} \alpha_i - \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j x_i^T x_j \\
max_\alpha\ L_D \quad subject\ \ to\ \ \sum_i \alpha_i y_i = 0\ \ \&\ \ \alpha_i \geq 0
\end{aligned}
$$

L_D도 이차계획법(Quadratic Programming)의 형태를 띄기 때문에 이 함수를 최대화하는 것으로 최적해에 도달할 수 있으며, 이를 통해 제약조건을 만족하는 α를 찾는다면 다음 식을 통해, 우리가 찾고싶어하던 Decision Boundary인 hyperplane을 찾을 수 있다.

$$
\begin{aligned}
f(x) = sgn(\sum_i \alpha_i y_i x^T x_i + b) \\
where\ \ sgn(x) \begin{cases} \frac{x}{|x|} & when\ \ x \neq 0 \\ 0 & otherwise \end{cases}
\end{aligned}
$$

∗위의 유도는 Hard Margin에 대한 경우이고, Soft Margin(혹은 C-SVM)의 경우에도 같은 방식으로 최적화할 수 있다. 그때의 목적함수와 Lagrange Primal L_P는 다음과 같고, 이를 다음과 같이 Lagrange Dual L_D로 변환하여 최적화한다.

$$
\begin{aligned}
min_{w,b} \frac{1}{2}\|w\|_2^2\ + C\sum_i \xi_i \\
subject\ to\ \ y_i (w^T x_i - b) \geq 1 - \xi_i,\ \ \xi_i \geq 0
\end{aligned}
$$

$$L_P = \frac{1}{2}\|w\|^2 + C\sum_i \xi_i$$
$$-\sum_i \alpha_i(y_i(w^T x_i - b) - 1 + \xi_i) - \sum_i \beta_i \xi_i$$
$$subject\ \ to$$
$$w = \sum_i \alpha_i y_i x_i,\ \ \sum_i \alpha_i y_i = 0,\ \ C - \alpha_i - \beta_i = 0$$
$$\alpha_i \geq 0,\ \ \beta_i \geq 0$$

$$L_D = \sum_{i=1}^{n} \alpha_i - \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j x_i^T x_j$$
$$max_\alpha\ L_D$$
$$subject\ \ to\ \ \sum_i \alpha_i y_i = 0\ \ \&\ \ 0 \leq \alpha_i \leq C$$
$$(\because \beta_i = C - \alpha_i \geq 0 \Rightarrow \alpha_i \leq 0)$$

Reference:

Duality

https://ratsgo.github.io/convex%20optimization/2018/01/25/duality/

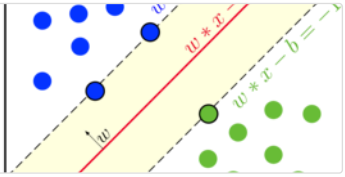
데이터 사이언스 스쿨

https://datascienceschool.net/02%20mathematics/05.03%20%EC%84%A0%ED%98%95%EA%B3%84%ED%9A%8D%EB%B2%95%20%EB%AC%B8%EC%A0%9C%EC%99%80%20%EC%9D%B4%EC%B0%A8%EA%B3%84%ED%9A%8D%EB%B2%95%20%EB%AC%B8%EC%A0%9C.html

[Data Mining] 10-1. SVM - Separable case

1979년 Vapnik가 박사과정때 개발한 SVM(Support Vector Machine), 한동안 주목받지 못하다가 1990년을 넘어서 대중적으로 알려지기 시작한 알고리즘이다. 본 챕터에서는 2개 범주를 SVM으로 분류하는 문제를 다루면서 아래 원리를 알아보도록 한다. 두 범주가 하나의 선형함수로 분리 가능한 경우(Separable case) 어떻게 선형 함

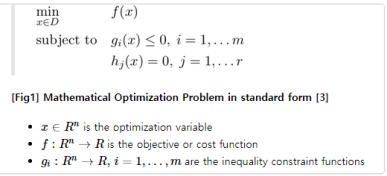
https://sonsnotation.blogspot.com/2020/11/data-mining-10-svm.html#



Convex optimization & Quadratic programming

Optimization problem

https://wonjun.oopy.io/17d73beb-3b70-468a-b1ef-1bff0d44e065



(3). Kernel Tricks

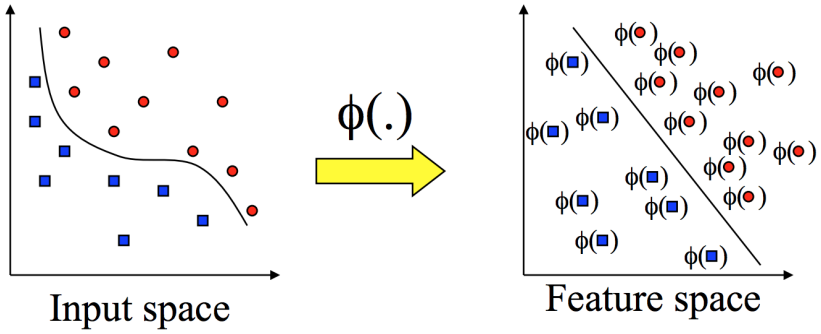
식 (10)을 보면, 결국 L 의 최대값은 $\vec{x_i} \cdot \vec{x_j}$ 에 의해 결정된다. 따라서 $\vec{x_i} \cdot \vec{x_j}$ 를 적절하게 잘 변형시켜주어야 하는데, 이것을 하는 방법은 1) $\vec{x_i}$ 와 $\vec{x_j}$ 각각에 대해 적절한 변환을 취해주거나 2) $\vec{x_i} \cdot \vec{x_j}$ 를 통째로 적절하게 변형시켜 주는 것이다.

즉,-

1) $\vec{x_1}$ 에 대한 적절한 변환 $\phi(\cdot)$ 을 찾아주어 $\vec{x_i} \cdot \vec{x_j}$ 대신에 $\phi(\vec{x_i}) \cdot \phi(\vec{x_j})$ 를 대입하거나

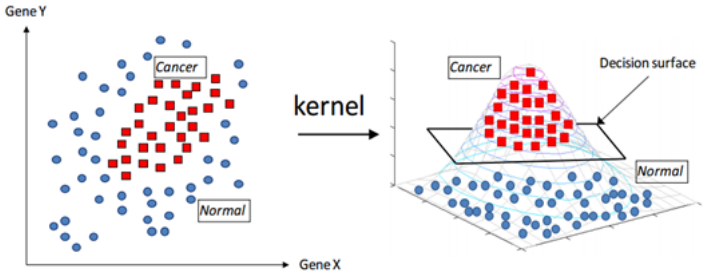
2) $\vec{x_i} \cdot \vec{x_j}$ 에 자체를 적절히 변환할 수 있는 $K(\vec{x_i}, \vec{x_j})$ 를 찾아주어 대입하는 것이다.

두 번째 방법이 sample들이 있는 공간을 kernel 함수를 이용하여 고차원 공간으로 변형시켜주는 커널 기법을 뜻한다.



위 그림은 방법1에 대한 그림으로, 이렇게 2차원의 데이터를 3차원에 생성된 데이터들 $\Phi(x)$ 를 feature map이라고 하며 $f(x) = w^T\Phi(x) + b$ 를 학습시키는 mapping을 나타낸다.

이와 달리, 커널 트릭은 SVM에서 mapping $\phi(x_i)^T,\ \phi(x_j)$ 를 인자로 받고 내적하여 실수를 출력하는 것이다. 즉 직접 mapping을 하지 않아도 되는것. 고차원에서 각 점들을 구하는 것보다 고차원에서 내적을 바로 구해버리는 지름길(trick)이라고 생각하면 된다.



$$\phi(x_i)^T \cdot \phi(x_j) = (x_i^T \cdot x_j)^2$$

정리하자면 커널은 새로운 고차원의 데이터를 기존 데이터의 선형 결합으로 나타낸 것이며 원본 데이터들 사이의 내적으로 볼 수 있다.

Reference

1. 행렬곱의 기하학적 의미

[기하학 (Geometry)] 내적 (Inner Product) 란?

내적에서는 두 개의 벡터를 입력으로 취하고, 단일 숫자 즉, 스칼라를 결과로 출력한다. 유클리디안 기하학에서, 두 벡터에 대한 데카르트 좌표계의 내적은 매우 널리 사용되고, 유클리디안 공간의 내적이라고 불린다. 유클리디안 공간에서 정의될 수 있는 내적은 유일한 것은 아니지만 말이다. 이것에 대해서는 내적 공간을 살펴보면 좀 더 자세히 알 수 있겠

https://m.blog.naver.com/sw4r/221939046286

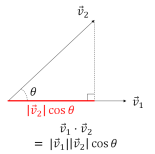
$$\mathbf{a} \cdot \mathbf{b} = 0.$$

행벡터의 의미와 벡터의 내적

지금까지 우리는 벡터란 무엇인지에 대해 알아보고, 행렬과 벡터의 곱 에 대해 알아보았다. 짧게 요약하자면 벡터란 상수배(곱셈 규칙)와 덧셈 규칙이 정의되는 원소들이라고 하였으며, 이들의 집합에 이 연산들이 정의된 집합을 벡터 공간(vector space)라고 한다고 하였다. 여기서 이러한 상수배와 덧셈 규칙이 정의되는 원소들을 '선형성을 갖는다'라

https://angeloyeo.github.io/2020/09/09/row_vector_and_inner_product.html

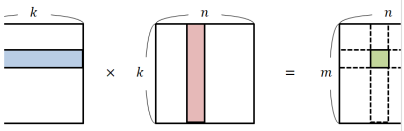
$$\vec{v}_1 \cdot \vec{v}_2 = \begin{bmatrix} a & b \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = ax + by$$



행렬 곱에 대한 또 다른 시각

필자는 고등학교 시절 7차 교육과정을 거치면서 고등학교 시절 행렬에 대해 배운 적이 있다. 그때 처음 배운 행렬의 곱셈 방식은 너무나도 독특했는데, 지금 생각해도 익숙하지 않다면 행렬의 곱은 "왜 이렇게 계산하지?"라고 생각이 들 정도로 이상하다. 행렬의 곱은 일반적으로 다음과 같이 생각한다. 그림 2. 행렬의 일반적인 곱셈 방법 수식을 이용해

https://angeloyeo.github.io/2020/09/08/matrix_multiplication.html

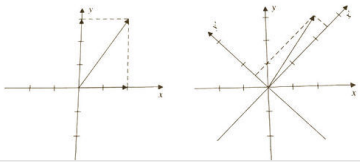


file:///C:/Users/user/Downloads/%EC%84%A0%ED%98%95%EB%8C%80%EC%88%98%ED%95%99%20%ED%81%B4%EB%9
2 선대 클린업 교안 1주차)

[LINEAR ALGEBRA - 1] [선형대수학 이야기 - 1] 기저와 기저변환

안녕하세요 오늘은 로봇공학을 포함한 모든 공학에서 공통적으로 필요로 되는 선형대수학에 대해서 포스팅하려합니다. 그 첫번째 포스팅으로 기저와 기저변환, 그리고 선형변환까지 소개해드리려합니다. 포스팅 시작하겠습니다!! 1) 기저 기저란, 영어로 Basis 라고 불립니다. 느낌적으로 기본이라는 뜻의 Base와 비슷하다고 생각되지 않나요? 기저

https://homo-robotics.tistory.com/16

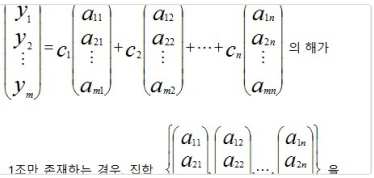


2. 기저

[선형대수] 기저 (Basis)

이번 포스팅에서는 기저 (basis)와 차원 (dimension)에 대해서 알아보겠습니다. 선형독립의 정의를 알지못하면 기저를 이해할 수 없으며, 선형독립과 기저의 개념이 헷갈릴 수도 있습니다. 따라서 혹시 선형독립(linearly independent)에 대해서 잘 모르고 있거나 가물가물하다면 이전 선형독립 포스팅을 다시 한번 살펴보시길 바랍니다. (☞ 선형독립

https://rfriend.tistory.com/164



[수학의 기초] 기저와 기저변환 행렬

수능, 교육청모의고사, 삼사, 경찰대 등의기출문제 풀이 동영상, 서울대 등 명문대 심층면접문제, 수리논술문제 풀이 동영상을 제공하여 자기주도적 인 수학을 공부할 수 있게 한다. 수학문제를 어떻게 풀 수 있는지 수학문제를 통해 제시한다. 또, 과학고 학생들이 공부하는 심화수학1,2, 고급수학1,2 선형대수학, AP Calculus 를 공부하는 참고자료와 학교

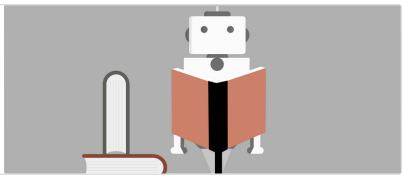
https://plusthemath.tistory.com/251

3. SVM

[ML] 5-1. 서포트 벡터 머신(SVM)이란 무엇일까?

ML] 4-2. 머신러닝의 테스트와 검증 및 데이터 불일치 여부 확인 ittrue.tistory.com/43 서포트 벡터 머신(SVM : Support Vector Machine)이란 두 클래스로부터 최대한 멀리 떨어져 있는 결정 경계 를 찾는 분류기로 특정 조건을 만족하는 동시에 클래스를 분류하는 것을 목표로 합니다. 결정 경계를 통해 어느 쪽에 속하는지 판단하는 것으로, 선

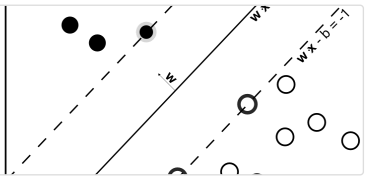
https://ittrue.tistory.com/44



서포트 벡터 머신 - 위키백과, 우리 모두의 백과사전

서포트 벡터 머신(support vector machine, SVM)은 기계 학습의 분야 중 하나로 패턴 인식, 자료 분석을 위한 지도 학습 모델이며, 주로 분류와 회귀 분석을 위해 사용한다. 두 카테고리 중 어느 하나에 속한 데이터의 집합이 주어졌을 때, SVM 알고리즘은 주어진 데이터 집합을 바탕으로 하여 새로운 데이터가 어느 카테고리에 속할지 판단하는 비 확

W https://ko.wikipedia.org/wiki/%EC%84%9C%ED%8F%AC%ED%8A%B8_%EB%B2%A1%ED%84%B0_%EB%A8%B8%EC%8B%A0



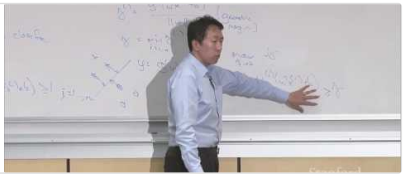
Lecture 6 - Support Vector Machines | Stanford CS229: Machine Learning Andrew Ng (Autumn 2018)

https://www.youtube.com/watch?v=lDwow4aOrtg&list=PLoROMvodv4rMiGQp3WXShtMGgzqpfVfbU&index=6

Lecture 7 - Kernels | Stanford CS229: Machine Learning Andrew Ng (Autumn 2018)

For more information about Stanford's Artificial Intelligence professional and graduate programs, visit: https://stanford.io/3GftN16Andrew Ng Adjunct Profess...

https://www.youtube.com/watch?v=8NYoQiRANpg&list=PLoROMvodv4rMiGQp3WXShtMGgzqpfVfbU&index=7



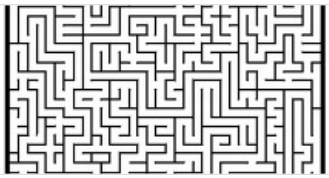
https://github.com/maxim5/cs229-2018-autumn/blob/main/notes/cs229-notes3.pdf

4. 커널

SVM과 Kernel 보기

SVM은 기본적으로 지도 학습의 한 알고리즘으로 Classification과 Regression 모두 가능한 알고리즘입니다. 1963년에 Vladimir N. Vapnik, Alexey Ya. Chervonenkis가 개발한 이후로 꾸준히 발전된 모델이고 한때는 기계학습을 대표하는 알고리즘이었습니다. 특히 커널의 발명으로 말이죠. Backpropagation의 등장으로 인공신경망 모델이 다시 뜨면

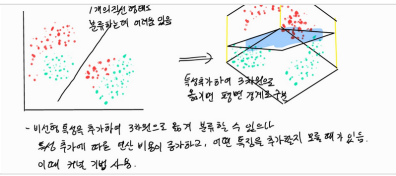
https://www.sallys.space/blog/2018/05/30/svm/



[Machine Learning] Kernel SVM(Support Vector Machine)

1. 커널 SVM이란 - 앞서 SVM 포스트에서는 선형 SVM을 중심으로 다뤘다. 그러나 실제로 선형 SVM으로 분류하기 어려운 데이터 형태들도 있다. 커널 기법의 기본적인 아이디어는 데이터를 높은 차원으로 이동시켜..

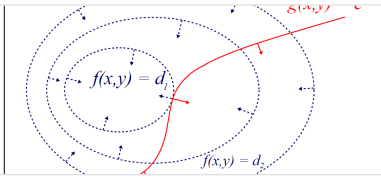
https://icefree.tistory.com/entry/Machine-Learning-Kernel-SVMSupport-Vector-Machine



서포트 벡터 머신(SVM)

본 포스팅은 MIT의 Patrick H. Winston 교수님의 강의를 정리한 것임을 밝힙니다.1. n-차원 공간에서 벡터를 이용한 hyperplane의 표현hyperplane이란 'a subspace of one dimension less than its ambient space...

https://angeloyeo.github.io/2020/09/30/SVM.html

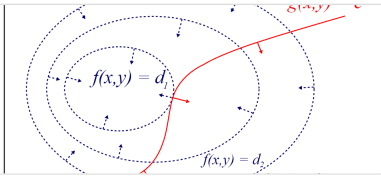


5. SVM에 대한 수식적 이해

서포트 벡터 머신(SVM)

본 포스팅은 MIT의 Patrick H. Winston 교수님의 강의를 정리한 것임을 밝힙니다.1. n-차원 공간에서 벡터를 이용한 hyperplane의 표현hyperplane이란 'a subspace of one dimension less than its ambient space...

https://angeloyeo.github.io/2020/09/30/SVM.html



SVM과 Kernel 보기

SVM은 기본적으로 지도 학습의 한 알고리즘으로 Classification과 Regression 모두 가능한 알고리즘입니다. 1963년에 Vladimir N. Vapnik, Alexey Ya. Chervonenkis가 개발한 이후로 꾸준히 발전된 모델이고 한때는 기계학습을 대표하는 알고리즘이었습니다. 특히 커널의 발명으로 말이죠. Backpropagation의 등장으로 인공신경망 모델이 다시 뜨면

https://www.sallys.space/blog/2018/05/30/svm/