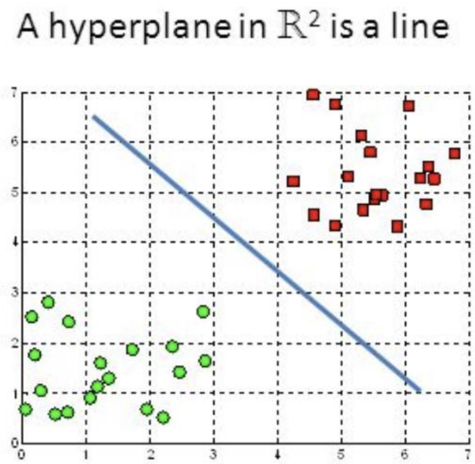
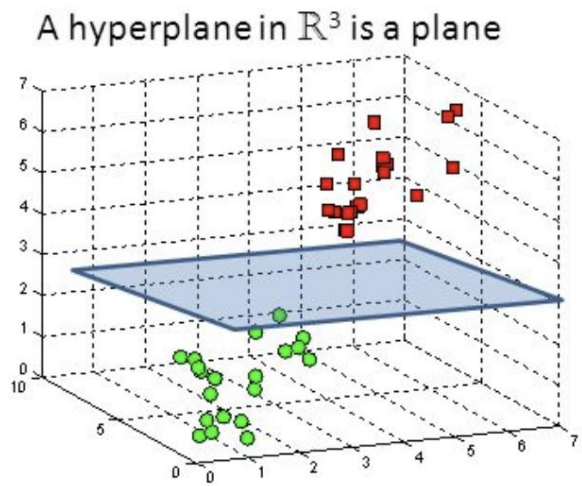




Support Vector Machine

구름

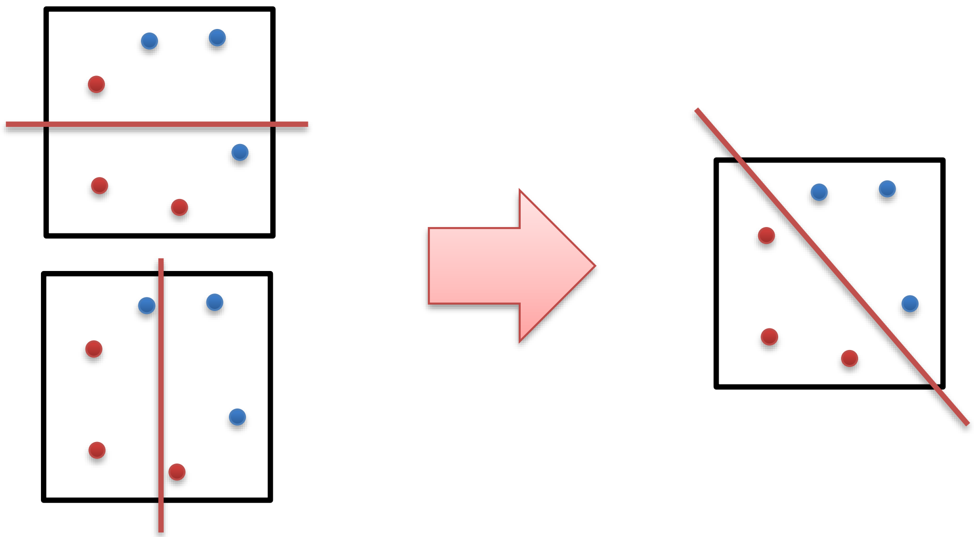
도시공학과 일반대학원

한양대학교

1. **SVM Hard margin**
2. SVM Soft margin
3. SVM kernel
4. Data

Suport Vector Machine (SVM)

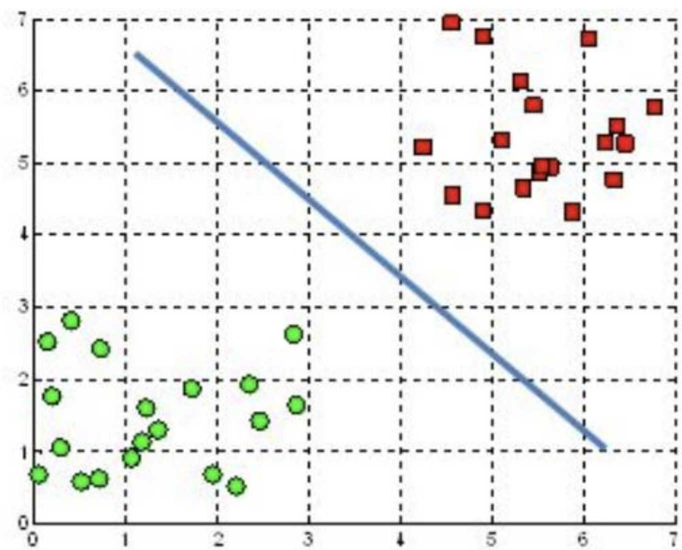
Cortes, C.; Vapnik, V. (1995). “Support-vector networks”.《Machine Learning》20 (3): 273. doi:10.1007/BF00994018.



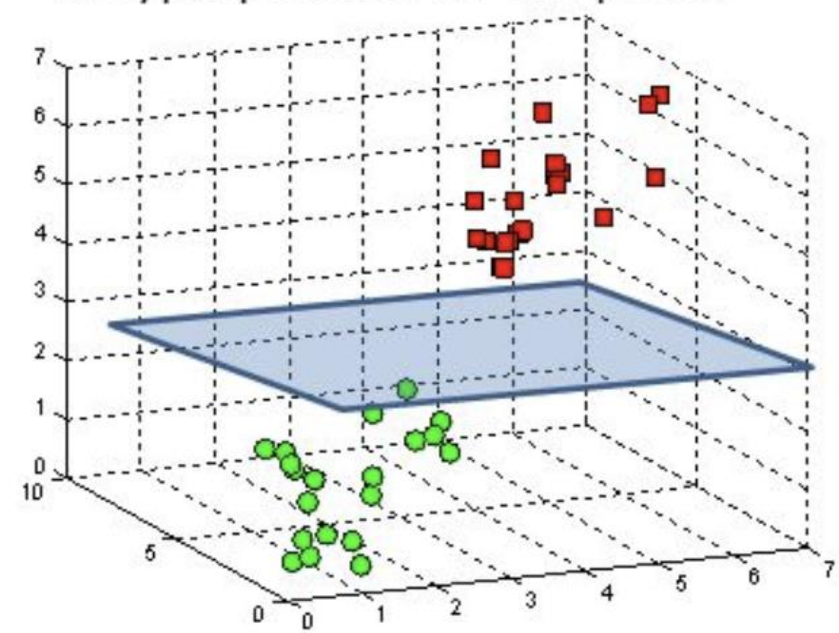
2차원 공간을 1차원 직선 함수로 분리

n차원 공간을 n-1차원 hyperplane으로 분리

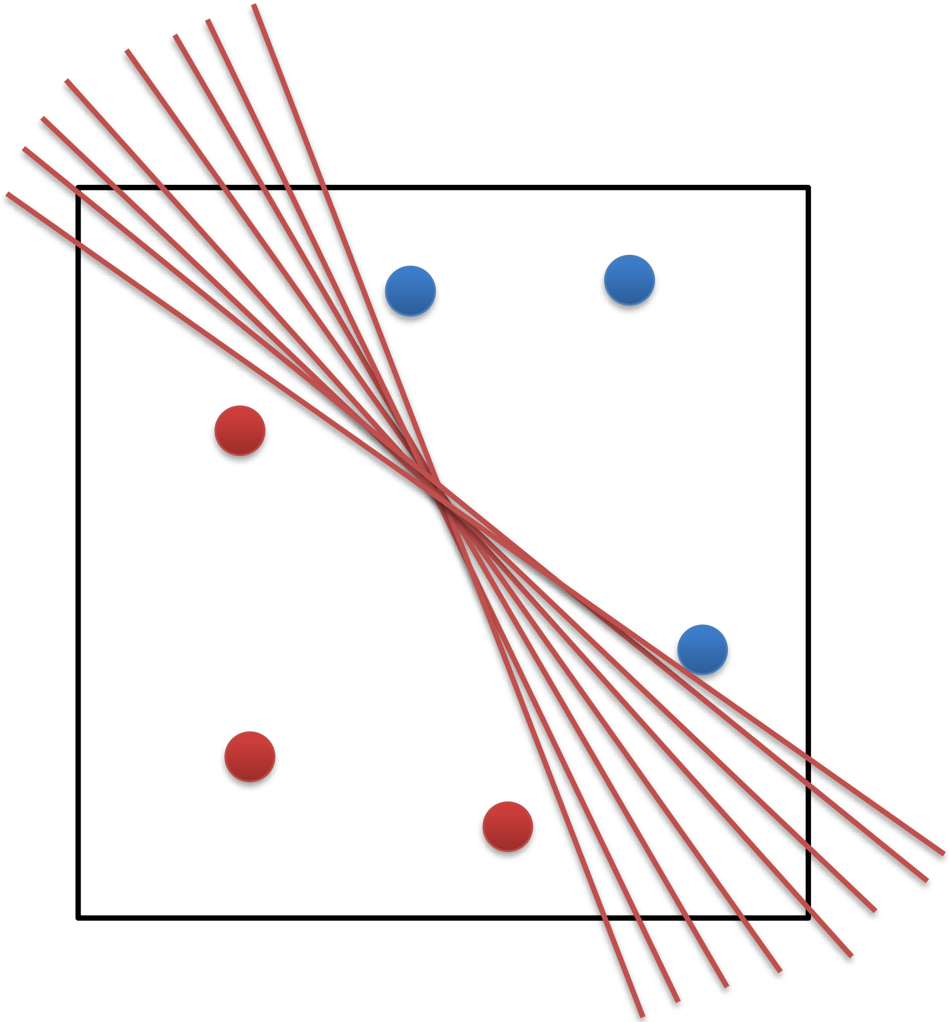
A hyperplane in $\mathbb{R}^2$ is a line



A hyperplane in $\mathbb{R}^3$ is a plane



평면을 분할할 수 있는 직선 분류기는 무한히 많음
기존 엔트로피 방식으로는 최적 분할 기준을 결정하기 어려움



ERM vs SRM

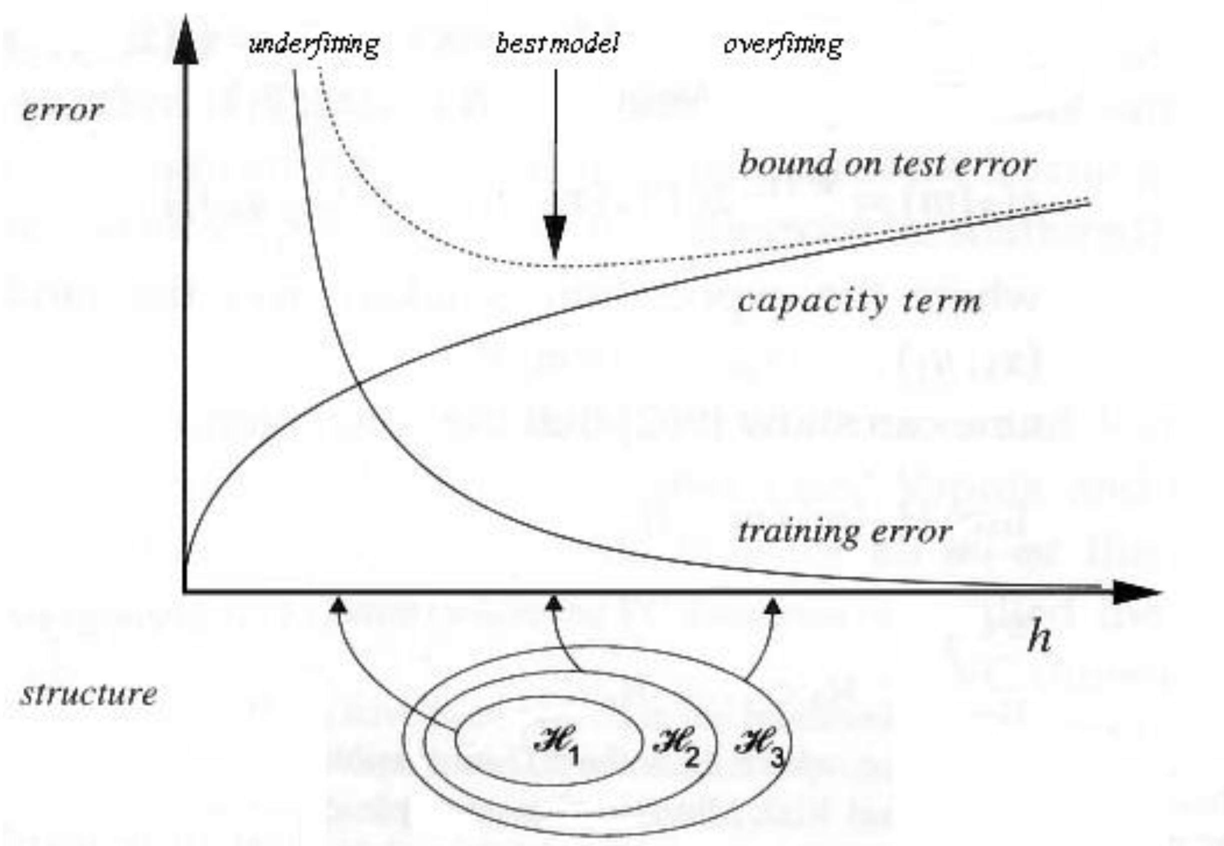
Empirical Risk Minimization vs Structural Risk Minimization

ERM : 학습데이터의 Error를 최소화 하는 모델을 선택
Decision tree의 Gini Index, Entropy 등이 추구하는 방향

SRM : ERM과 함께 모델의 구조적 위험을 최소화 하는 모델을 선택
SVM이 추구하는 방향

SRM 구조적 위험

구조적 복잡도(h)가 높아질수록 학습데이터의 에러율이 낮아지고
모델의 복잡도(Capacity term)이 높아지므로
학습데이터 에러율과 모델 복잡도를 동시에 고려하여 가장 낮은 지점의 모델을 선택 해야 한다



V.N. Vapnik and A.Ya. Chervonenkis. *Theory of Pattern Recognition*. Nauka, Moscow, 1974. (in Russian)
http://www.svms.org/srm/

SRM 구조적 위험

$$R_{srm}(f) \le R_{\mathrm{erm}}(f) + \sqrt{\frac{\mathrm{h}\left(\ln\frac{2\mathrm{n}}{\mathrm{h}} + 1\right) - \ln\left(\frac{\delta}{4}\right)}{\mathrm{n}}}$$

ERM　　　　Capacity Term

모델 복잡도는 학습에 사용된 데이터의 수 (n)이 많을 수록 낮고 함수 f의 VC-dimension(h)이 낮을 수록 낮아진다.

Hyperplane classifier VC dimension (γ)

$$\gamma \leq 1 + \ \mathrm{Min}\left(1\frac{R^2}{d_{min}^2}, n\right)$$

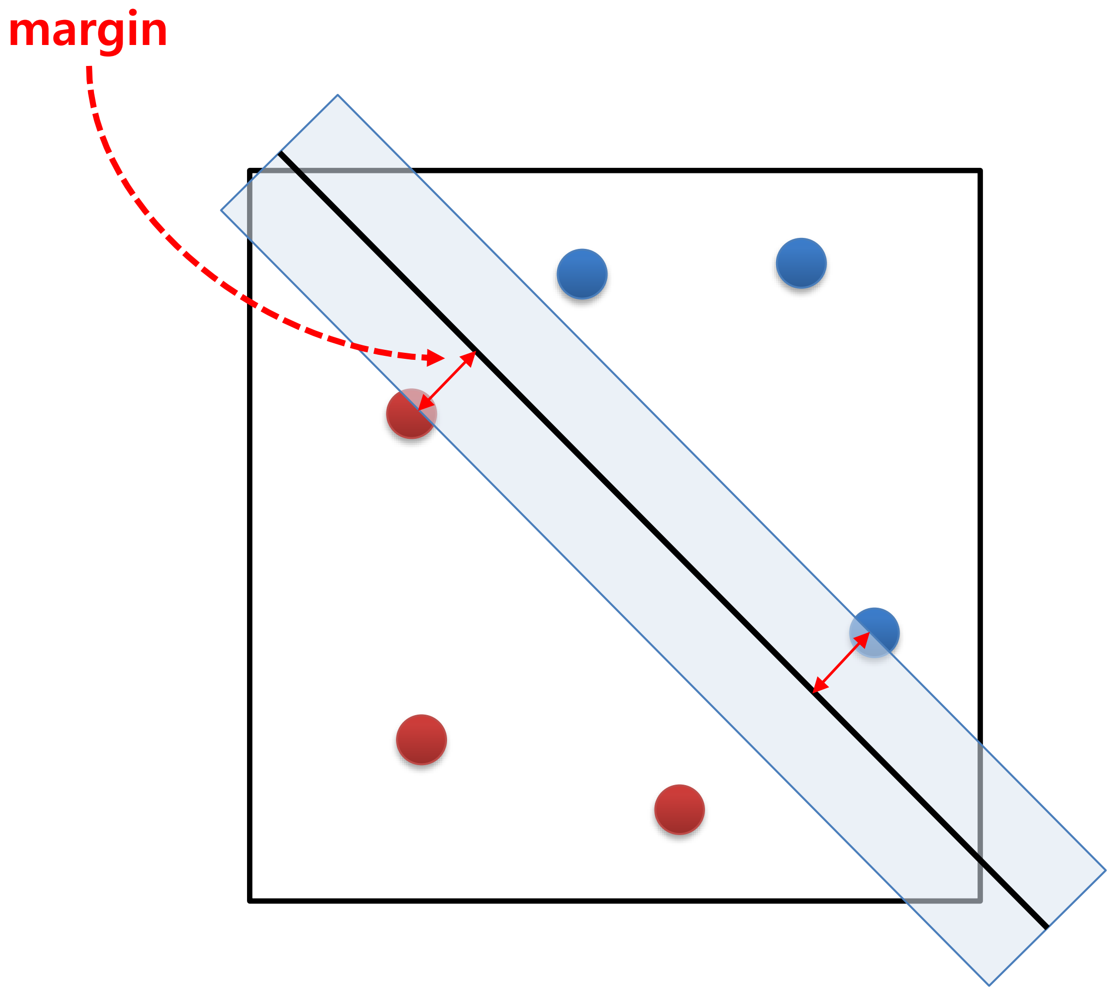
V. Vapnik, Statistical learning theory, Wiley, 1998.

R : 전체 학습 데이터가 포함되는 원의 반지름

n : 전체 학습데이터 수

d : Hyperplane 마진

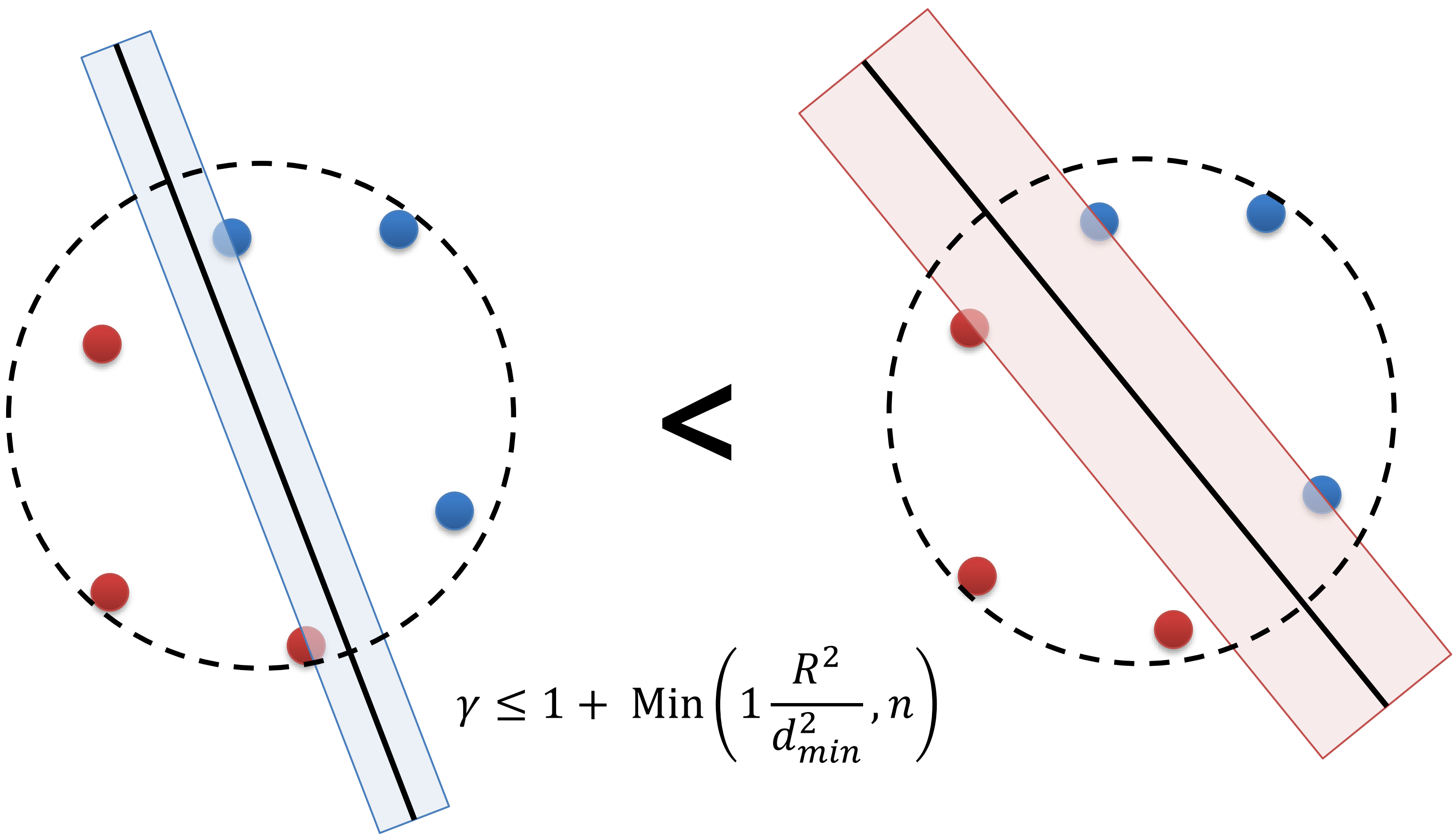
마진 : hyperplane으로부터 가장 가까운 데이터까지의 직선 거리



분류 함수에 따라서 margin의 길이가 달라짐

마진이 가장 큰 직선 함수가 VC dimension이 가장 작고

구조적 위험인 SRM이 가장 작은 함수가 됨



$$\gamma \leq 1 + \mathrm{Min}\left(1\frac{R^2}{d_{min}^2}, n\right)$$

직선의 방정식 이해

$$w^T \cdot d = c$$

$$w = \begin{bmatrix} a \\ b \end{bmatrix} \qquad \|w\| = \sqrt{a^2 + b^2}$$

$$w^T \cdot d = c$$

$$w = \begin{bmatrix} a \\ b \end{bmatrix} \qquad d = \begin{bmatrix} x \\ y \end{bmatrix} \qquad c$$

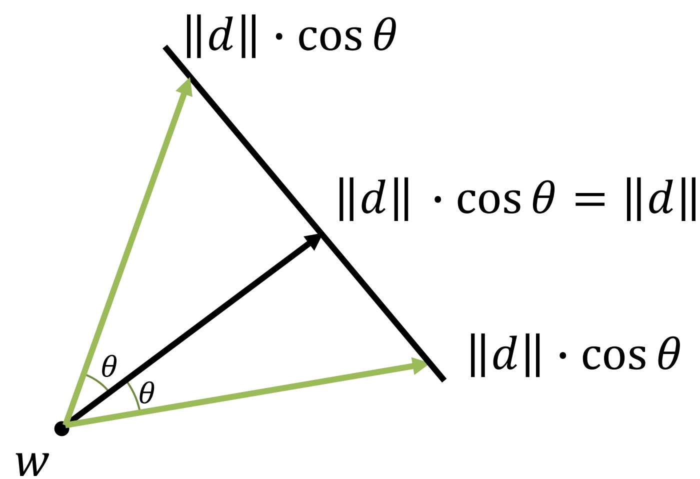
$$w^T \cdot d = \|w\| \cdot \|d\| \cdot \cos\theta = c$$

$$w^T \cdot d = c$$

$$\begin{bmatrix} a & b \end{bmatrix} \cdot \begin{bmatrix} x \\ y \end{bmatrix} = c$$

$$a \cdot x + b \cdot y = c$$

$$y = -\frac{a}{b} a \cdot x + \frac{c}{b}$$

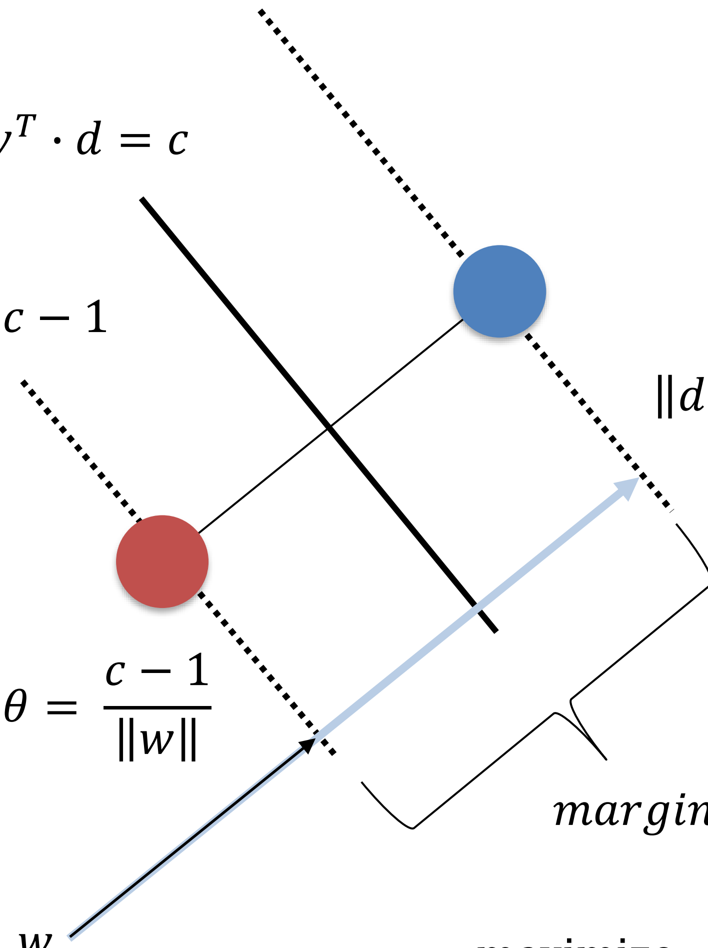


마진의 길이와 최적화 문제

$$w^T \cdot d = c + 1$$

$$w^T \cdot d = c$$

$$w^T \cdot d = c - 1$$



$$\|d\| \cdot \cos\theta = \frac{c+1}{\|w\|}$$

$$\|d\| \cdot \cos\theta = \frac{c-1}{\|w\|}$$

$$margin = \frac{c+1}{\|w\|} - \frac{c-1}{\|w\|} = \frac{2}{\|w\|}$$

$$\text{maximize} \frac{2}{\|w\|} = minimize\ \frac{1}{2}\|w\|^2$$

목표: $\min \frac{1}{2}\|w\|^2$

조건: $w^T \cdot \{d_1, d_3, d_6\} - c \geq +1$

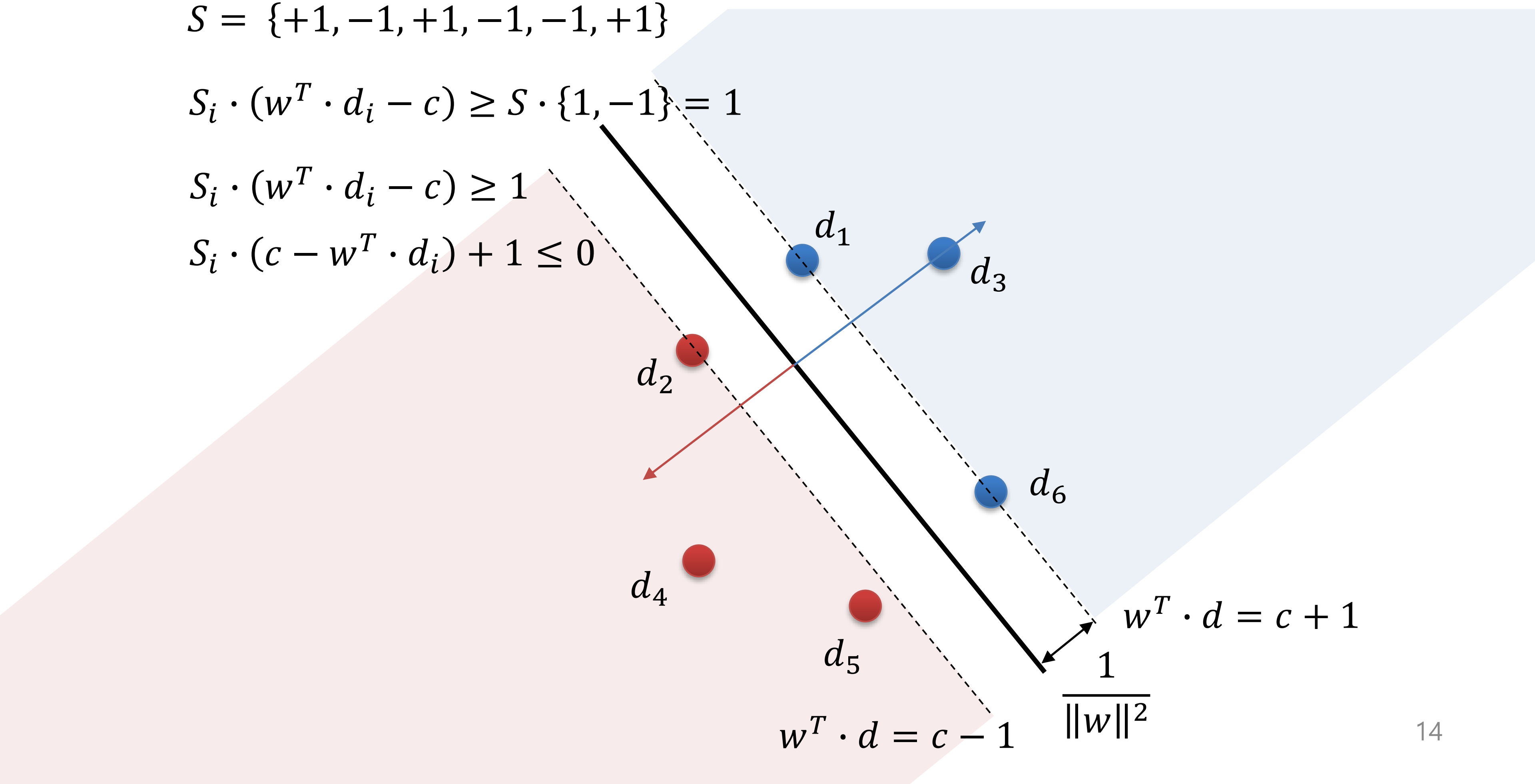
$w^T \cdot \{d_2, d_4, d_5\} - c \leq -1$

$S = \{+1, -1, +1, -1, -1, +1\}$

$S_i \cdot (w^T \cdot d_i - c) \geq S \cdot \{1, -1\} = 1$

$S_i \cdot (w^T \cdot d_i - c) \geq 1$

$S_i \cdot (c - w^T \cdot d_i) + 1 \leq 0$



마진최대 w 찾기 : $\min \frac{1}{2}\|w\|^2 \qquad s.t.\ S_i \cdot (c - w^T \cdot d_i) + 1 \leq 0$

Lagrangian Problem (primal)

$$\min L_p(w,c) = \frac{1}{2}\|w\|^2 + \sum_{i=1}^{n} \gamma_i (S_i(c - w^T \cdot d_i) + 1)$$

$$s.t. \gamma_i \geq 0$$

KKT condition

$$\frac{\partial L_p}{\partial w} = 0 \rightarrow w = \sum_{i=1}^{n} \gamma_i S_i d_i \qquad \frac{\partial L_p}{\partial c} = 0 \rightarrow \sum_{i=1}^{n} \gamma_i S_i = 0$$

Lagrangian Dual

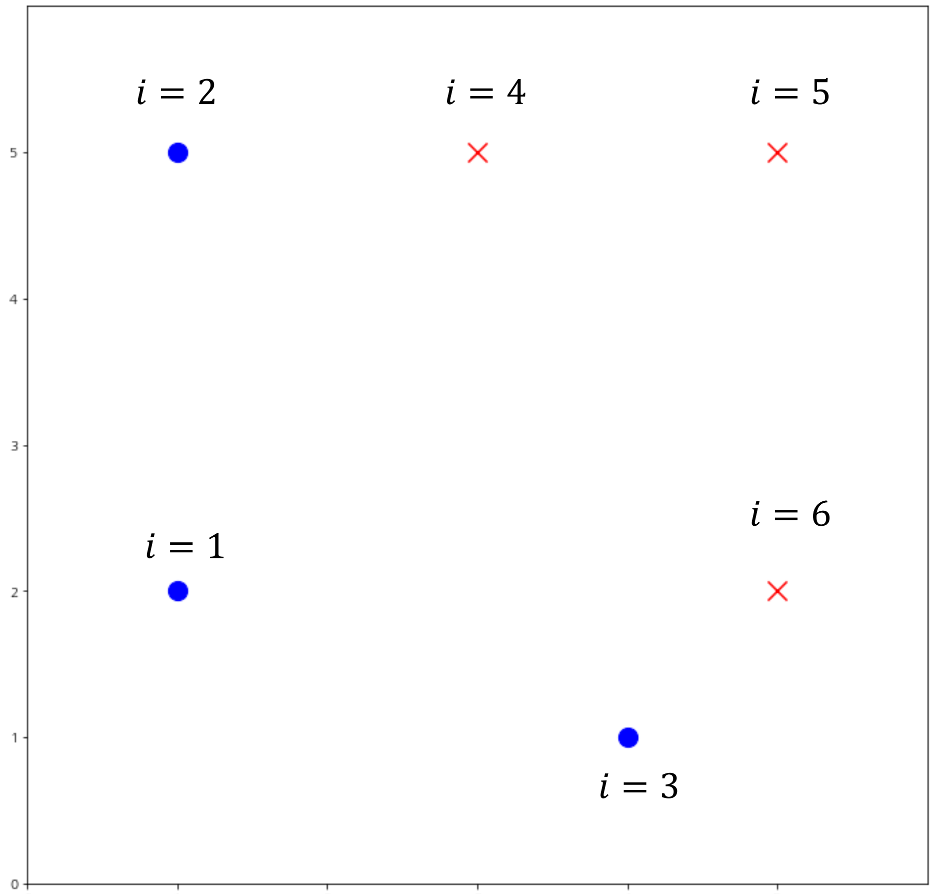
$$\max L_D(\gamma_i) = \sum_{i=1}^{n} \gamma_i - \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} S_i \gamma_i (d_i^T d_j) S_j \gamma_j$$

$$s.t. \sum_{i=1}^{n} a_i y_i = 0 \qquad s.t.\ a_i \geq 0$$

데이터를 통해 SVM 구하기

x = np.array([[1, 2], [1, 5], [4, 1], [3, 5], [5, 5], [5, 2]])

y = np.array([1, 1, 1, -1, -1, -1])



Cvxopt 라이브러리 사용

https://cvxopt.org/
참조 : https://xavierbourretsicotte.github.io/SVM_implementation.html

$$\begin{array}{ll} \text{minimize} & (1/2)x^T P x + q^T x \\ \text{subject to} & Gx \preceq h \\ & Ax = b \end{array}$$

$$\max L_D(\gamma_i) = -\frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} S_i\gamma_i(d_i^T d_j)S_j\gamma_j + \sum_{i=1}^{n}\gamma_i$$

$$s.t.\, \gamma_i \geq 0$$

$$s.t. \sum_{i=1}^{n}\gamma_i y_i = 0$$

alphas =

9.53968e-13	0.0408163	0.979592	6.73121e-12	5.0051e-13	1.02041

w 구하기

alphas =

9.53968e-13	0.0408163	0.979592	6.73121e-12	5.0051e-13	1.02041

$$w = \sum_{i=1}^{n} \gamma_i S_i d_i$$

KKT 조건에서 도출,
γ_i, S_i, d_i 모두 알고 있으므로 계산 가능

```
w = np.zeros(x.shape[1])
for ai, xi, yi in zip(maximum[0,:], x, y):
    w += ai * yi * xi
```

	0
0	-1.14286
1	-0.857143

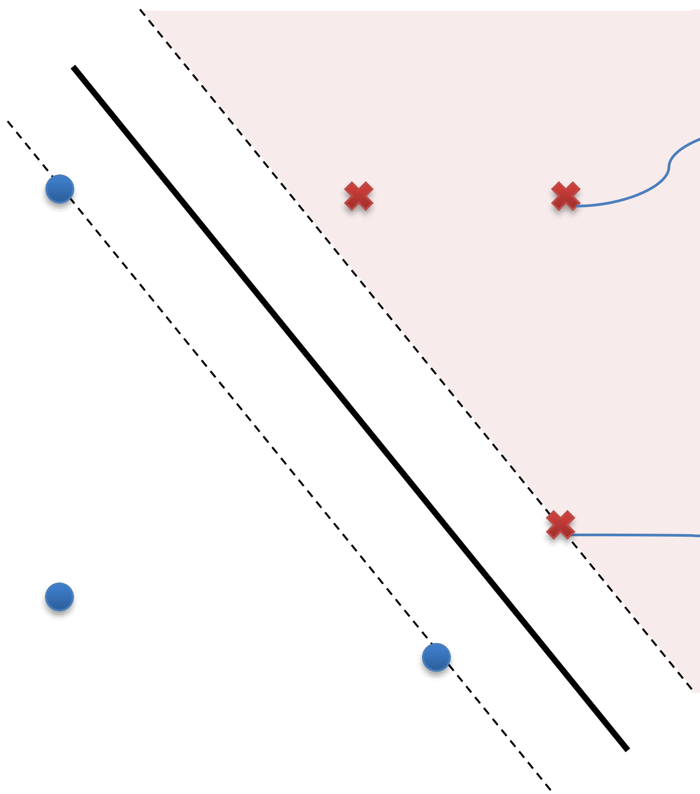
Support Vector 구하기

$$\gamma_i(S_i(c \; - \; w^T \cdot d_i) + 1)$$

라그랑지안에 따라 목적함수와 미분 방향이 같은 조건식만 γ_i가 양수

따라서 γ_i가 양수인 d_i 점이 Support Vector에 해당함

9.53968e-13	0.0408163	0.979592	6.73121e-12	5.0051e-13	1.02041



$a_5 = 0$

$y_5(w^T \cdot x_5 + b) \; - 1$

-1 *(-3.571428) - 1 = 0

$a_6 = 1.02041$

$y_6(w^T \cdot x_6 + b) \; - 1$

-1 * (-1) - 1 = 0

c 구하기

$w^T d_i + c = S_i$ 마진 위에 존재하는 Support Verctor들은 등식이 성립

$c = S_i - w^T d_i$ SV 들의 c값을 계산하면 미세하게 차이가 존재

평균을 사용

```
bs = np.empty((0), float)
support_vectors_ = np.empty((0,2), float)
for ai, xi, yi in zip(maximum[0,:], x, y):
    if ai > 0.000000000001 : #threshold
        bs = np.append(bs, yi - np.dot(w.T, xi))
        support_vectors_= np.append(support_vectors_, np.array([xi]), axis=0)

b = bs.sum() / len(bs)
```

	0
0	6.4285714283576
1	6.42857142874564
2	6.42857142874855

$b = 6.4285714286172615$

SVM 학습 완료

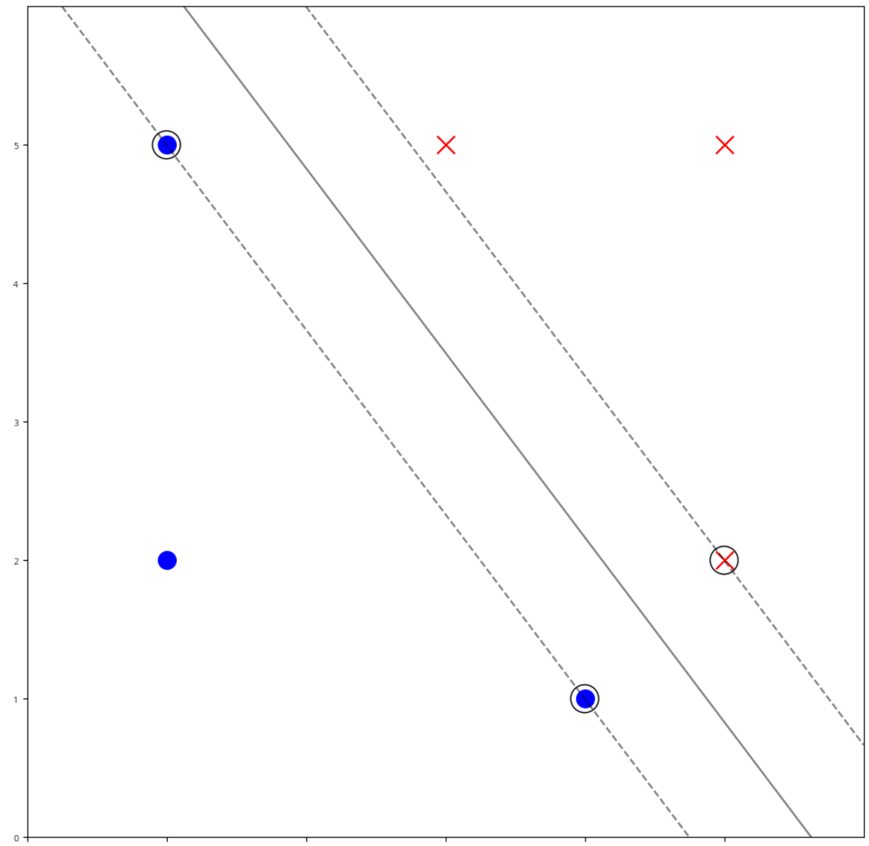
$\gamma =$

9.53968e-13	0.0408163	0.979592	6.73121e-12	5.0051e-13	1.02041

$w =$

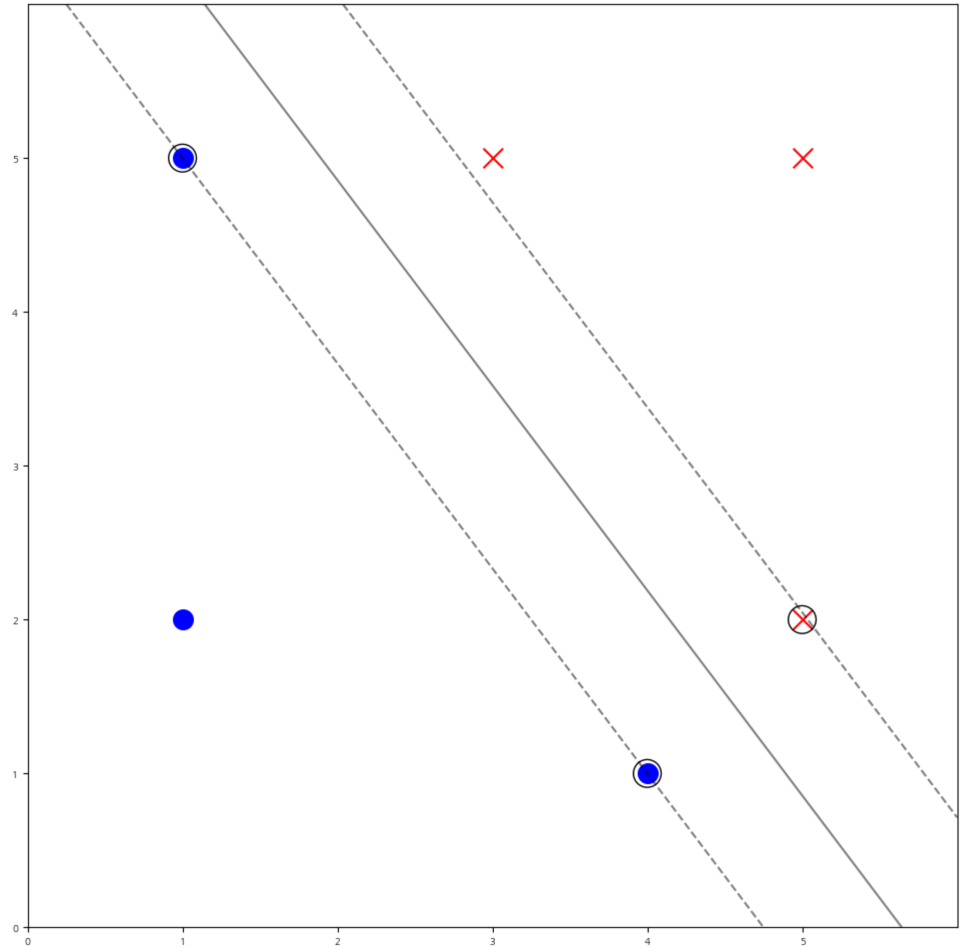
-1.14286	-0.857143

$c =$ 6.428571428617261 5



라이브러리 사용 SVM

```
from sklearn.svm import SVC
classifier = SVC(kernel = 'linear', c=1e10)
classifier.fit(x, y)
```



라이브러리 학습 내용 추출

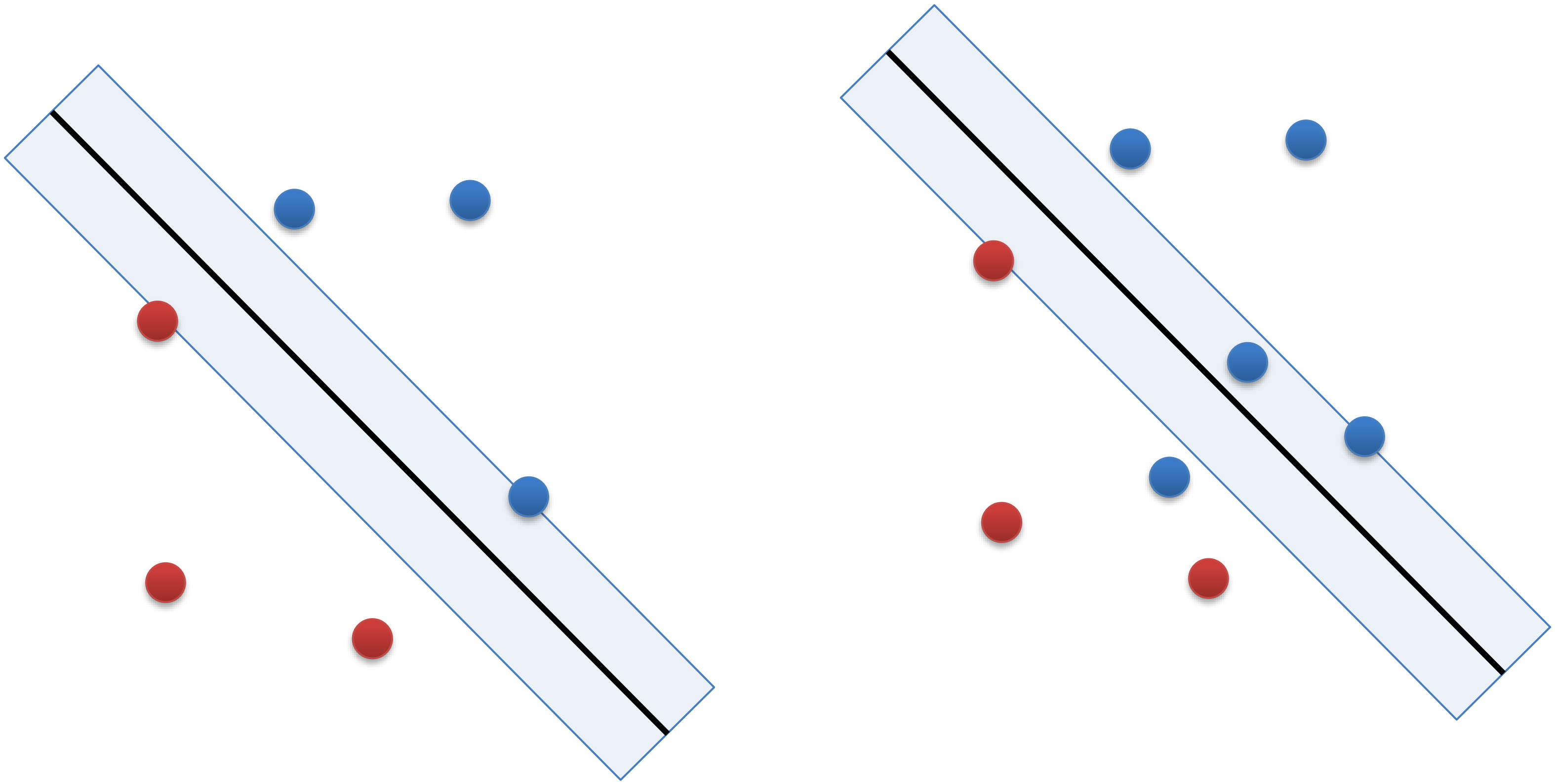
```
classifier.coef_
w
classifier.intercept_
b


alphas2 = np.zeros(y.shape, float)
alphas2[classifier.support_] = classifier.dual_coef_.reshape(alphas2[classifier.support_].shape)
alphas2 = alphas2 * y

alphas2
alphas
```

1. SVM Hard margin
2. **SVM Soft margin**
3. SVM kernel
4. Data

Hard Margin vs Soft Margin



Soft margin

마진최대 w, ξ 찾기 : $\min \ \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{n}\xi_i$ $\quad s.t.\ S_i \cdot (c - w^T \cdot d_i) + 1 + \xi_i \le 0$

$s.t.\ \ -\xi_i \le 0$

Lagrangian Problem (primal)

$$\min L_p(w, c, \xi) = \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{n}\xi_i + \sum_{i=1}^{n}\gamma_i(S_i(c - w^T \cdot d_i) + 1 + \xi_i) - \sum_{i=1}^{n}\mu_i\xi_i$$

$$s.t. \gamma_i \ge 0 \qquad s.t. \mu_i \ge 0$$

KKT condition

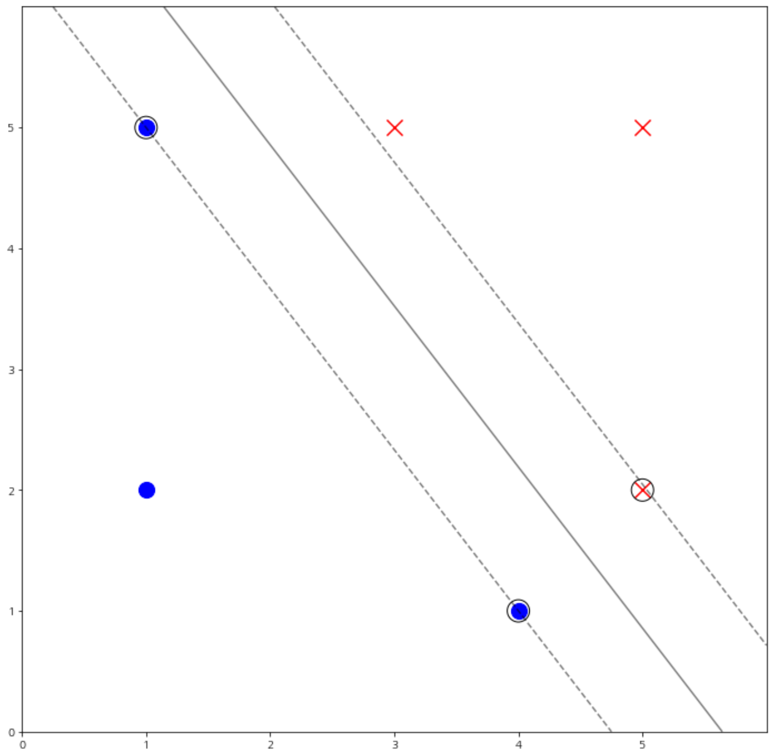
$$\frac{\partial L_p}{\partial w} = 0 \rightarrow \ w = \sum_{i=1}^{n}\gamma_i S_i d_i \qquad \frac{\partial L_p}{\partial c} = 0 \rightarrow \sum_{i=1}^{n}\gamma_i S_i = 0$$

$$\frac{\partial L_p}{\partial \xi_i} = 0 \rightarrow \ \mathrm{C} - \gamma_i - \mu_i = 0$$

Lagrangian Dual $\max L_D(\gamma_i) = \sum_{i=1}^{n}\gamma_i - \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n}S_i\gamma_i(d_i^T d_j)S_j\gamma_j$

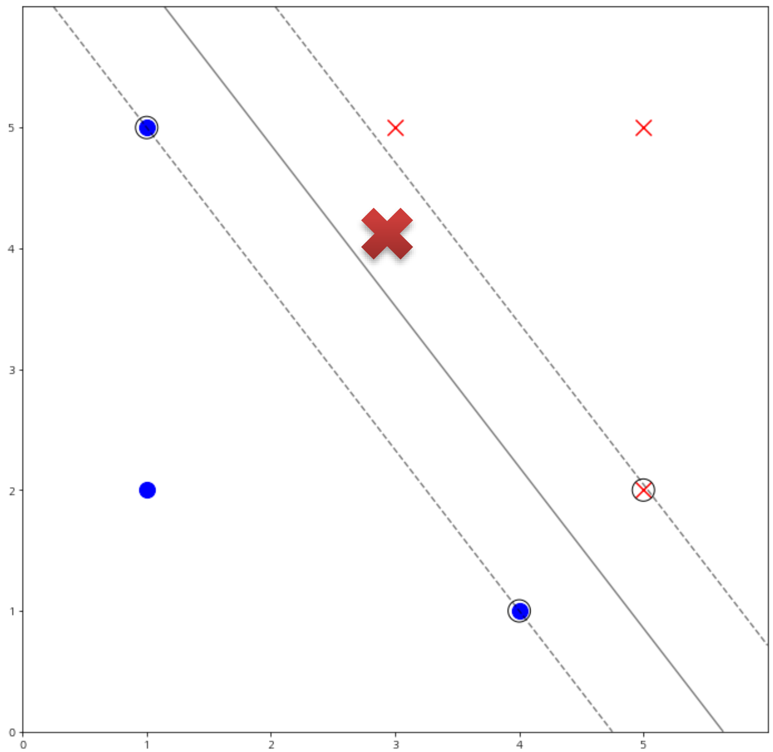
$$s.t. \sum_{i=1}^{n}\gamma_i S_i = 0 \qquad s.t.\ 0 \le a_i \le \mathrm{C}$$

데이터를 추가하여 Softmargin Test

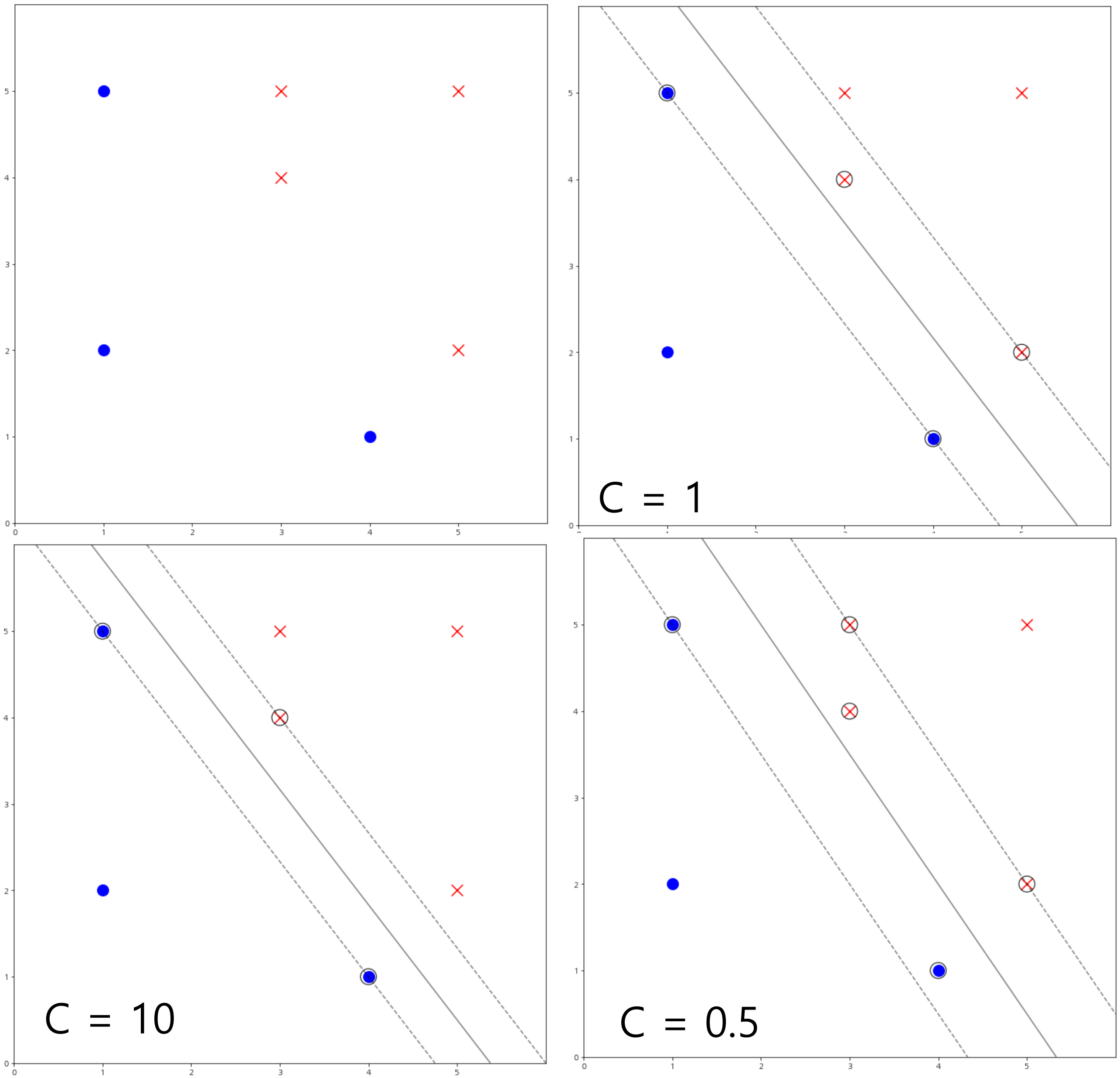


데이터 추가

$\{x_7, y_7\}$ = { (3, 4) , -1}



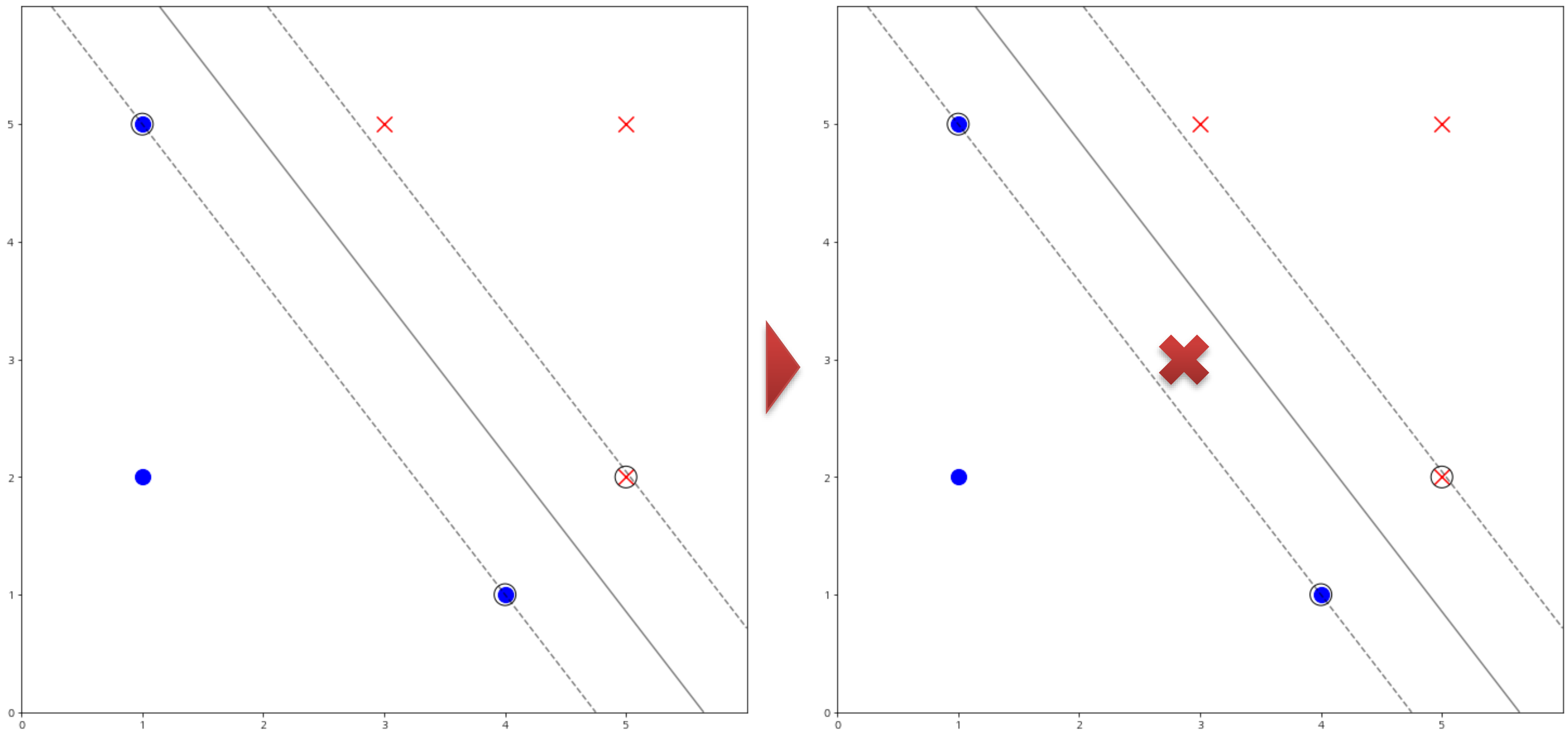
Hyper Parameter C



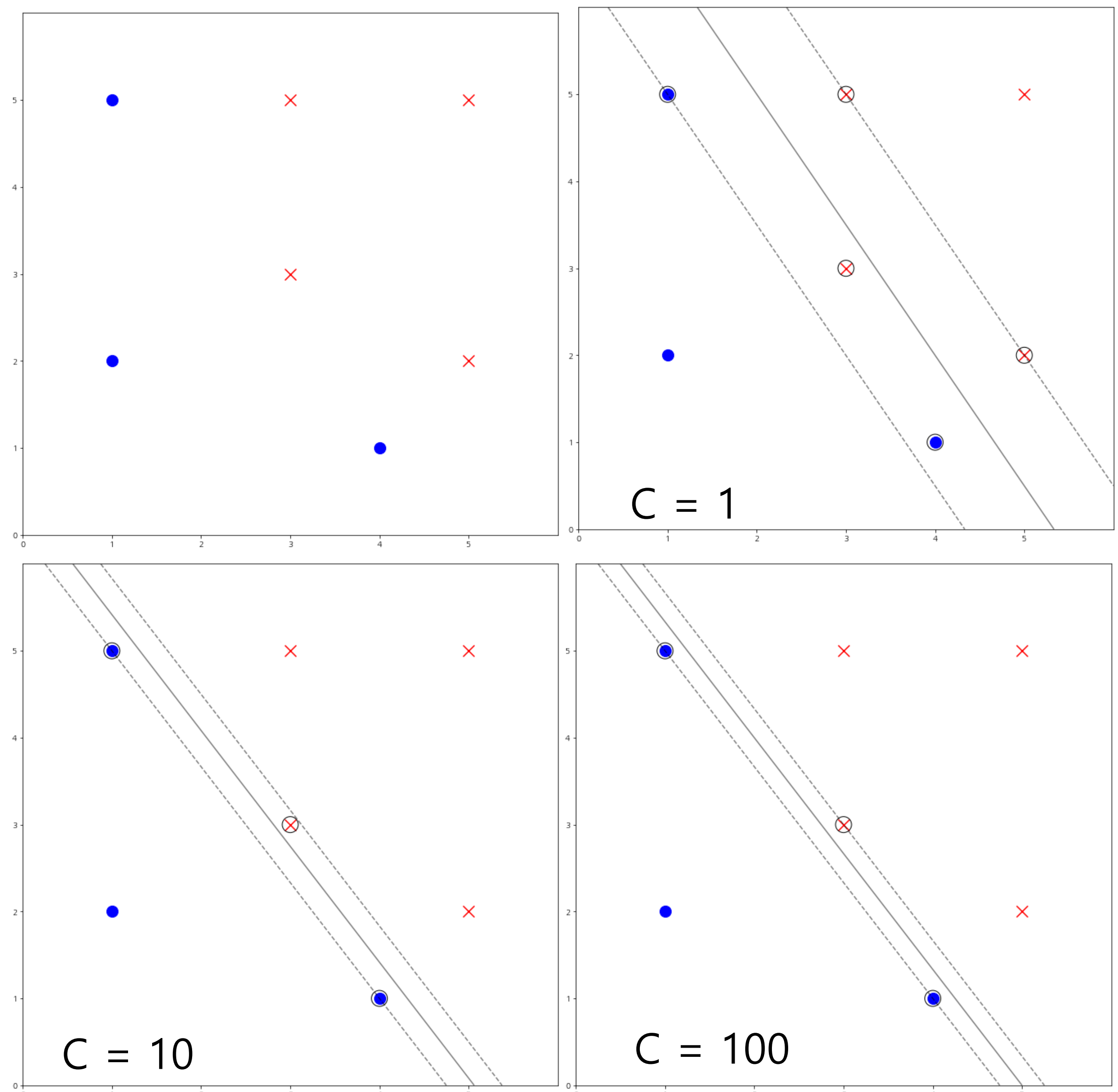
데이터를 추가하여 Soft margin Test

데이터 추가

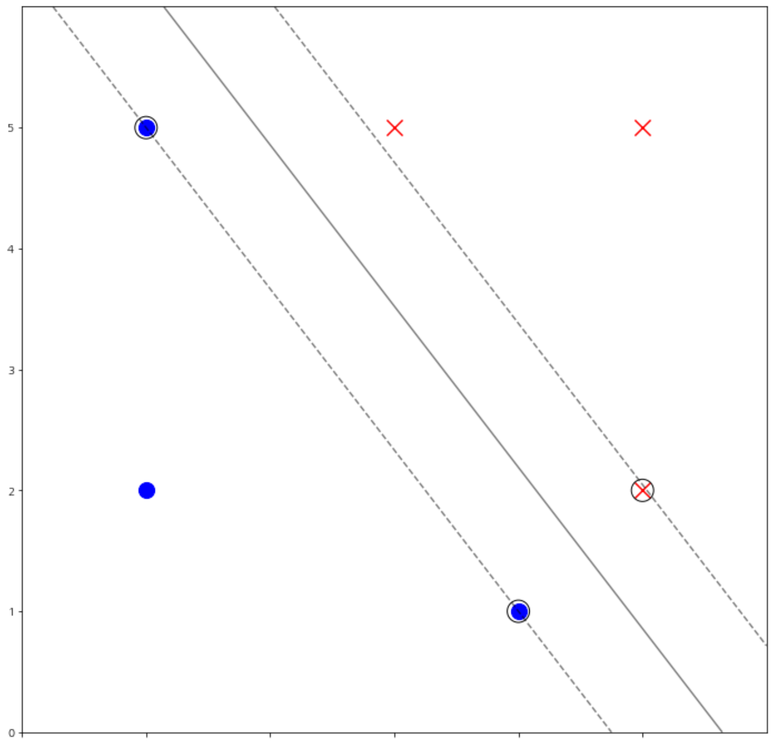
$\{x_7, y_7\}$ = { (3, 3) , -1}



Hyper Parameter C

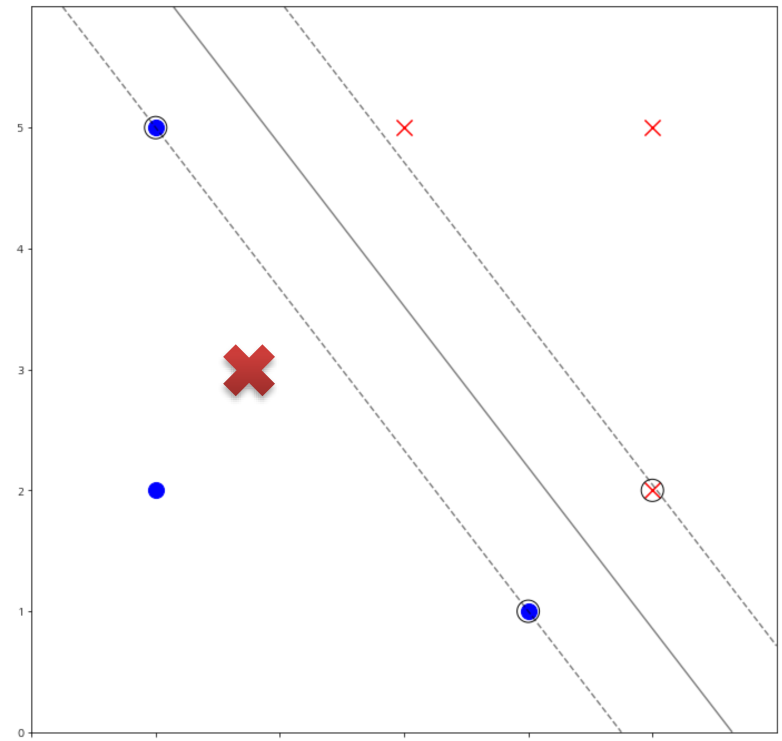


데이터를 추가하여 Softmargin Test

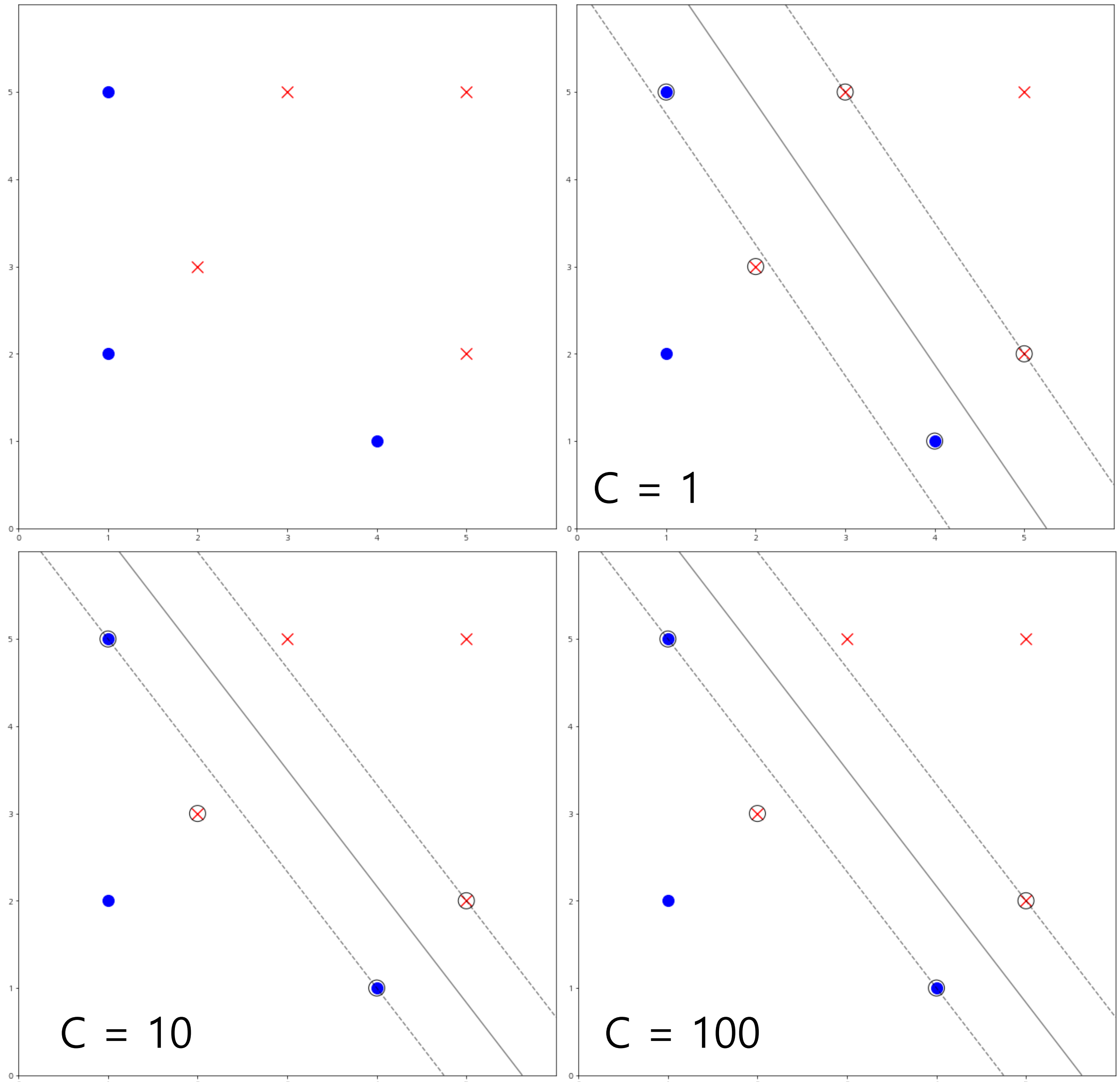


데이터 추가

$\{x_7, y_7\}$ = { (2, 3) , -1}

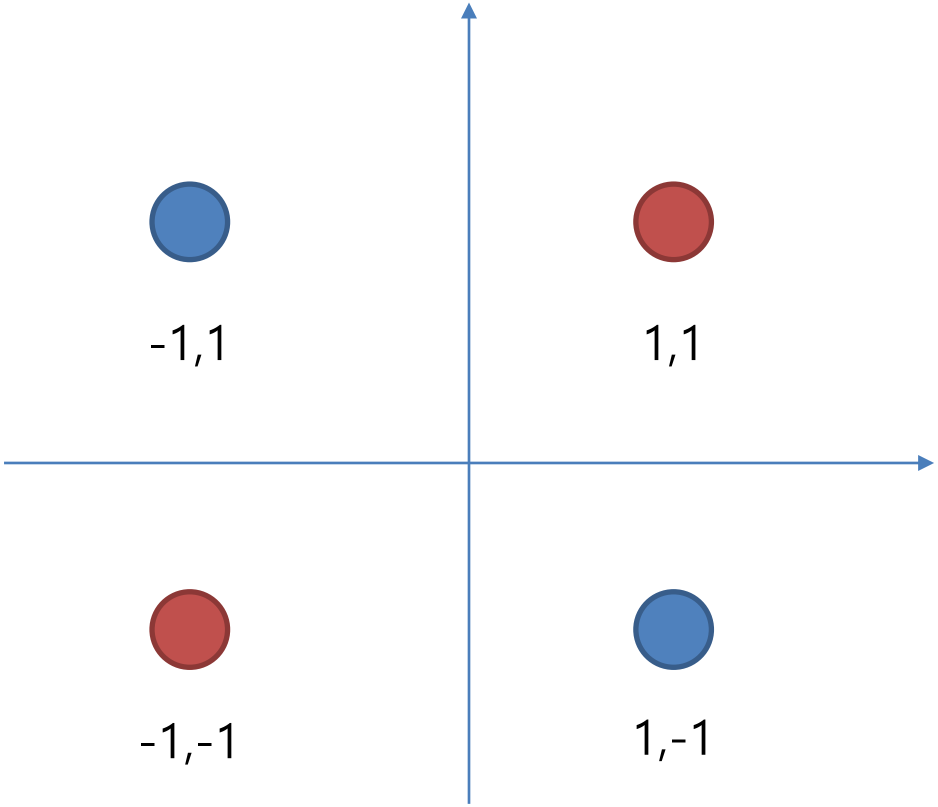


Hyper Parameter C



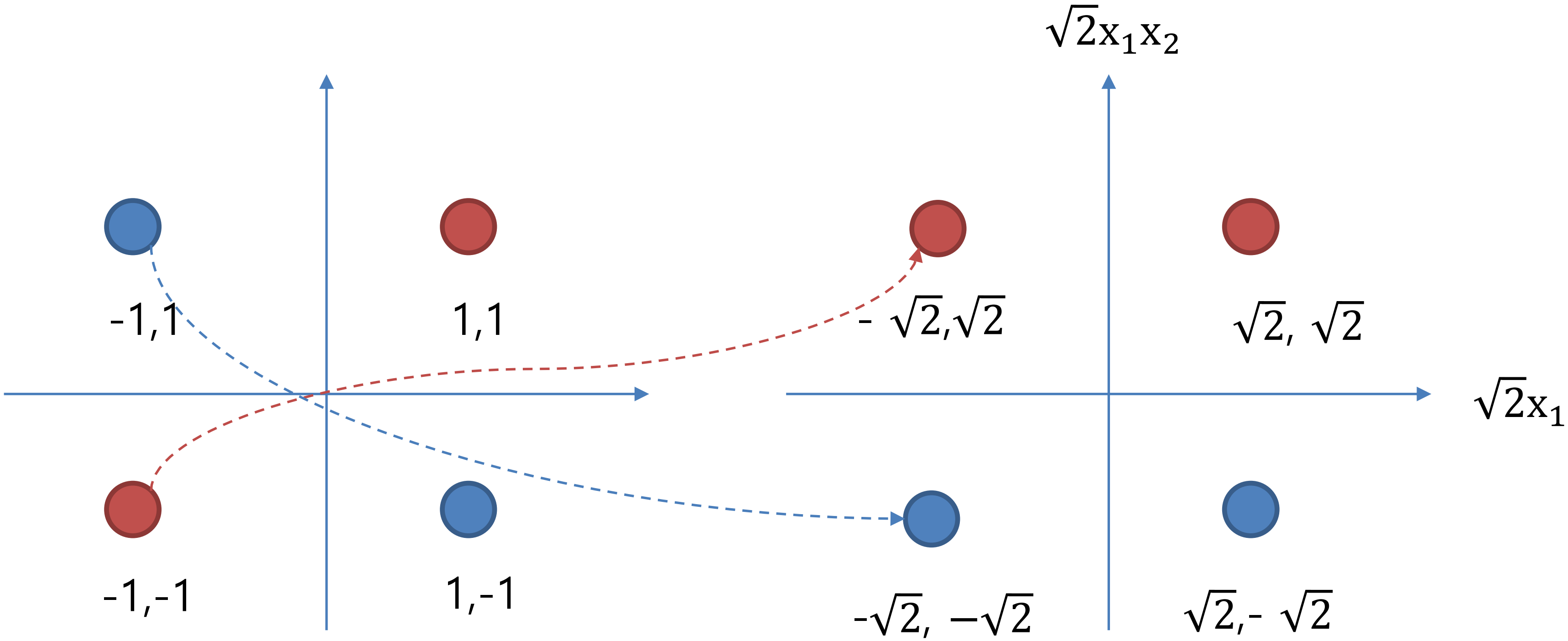
1. SVM Hard margin
2. SVM Soft margin
3. **SVM kernel**
4. Data

XOR 데이터 : 선형 분류가 불가능

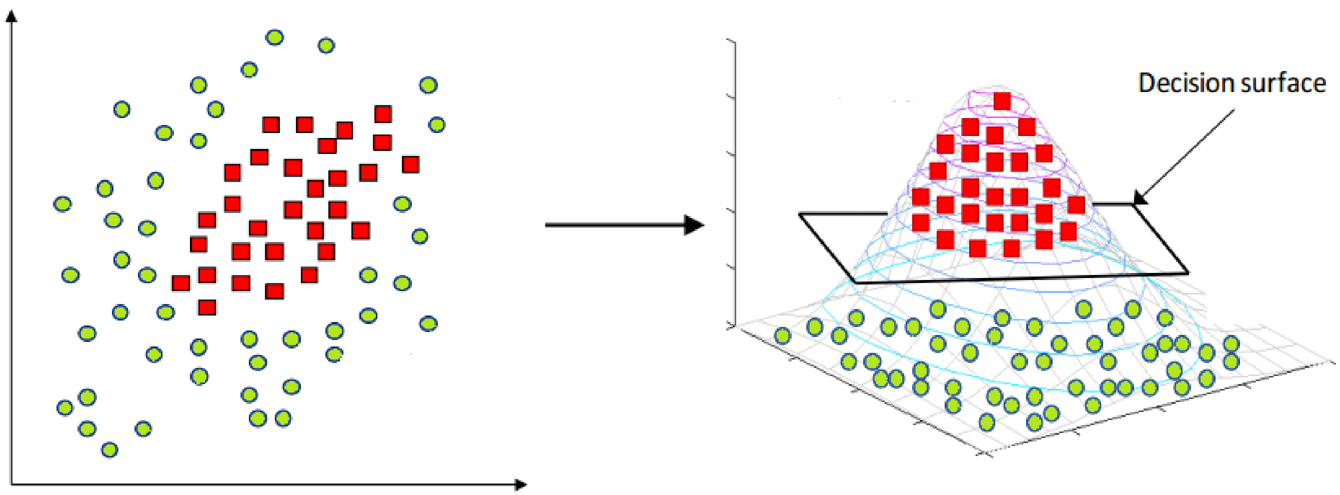


Mapping Function을 이용하면 차원을 변경시켜서 구분이 가능

$$\Phi(x_1, x_2) \rightarrow \left(x_1^2, x_2^2, \sqrt{2}x_1, \sqrt{2}x_2, \sqrt{2}x_1x_2, 1\right)$$



Mapping Function을 통해 차원을 d → D 높이면 SVM 분석이 가능



Mapping Function으로 변환 후 SVM을 계산하는 것에 비해

Kernel 함수를 이용하면 변환 없이 한번에 계산이 가능

Kernel함수 = Mapping이후의 내적이 동일

$$\max L_D(\gamma_i) = \sum_{i=1}^{n} \gamma_i - \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} S_i \gamma \Phi(d_\mathrm{i})^\mathrm{T} \Phi(d_\mathrm{j}) S_j \gamma_j$$

$$s.t. \sum_{i=1}^{n} \gamma_i S_i = 0 \qquad s.t.\ 0 \le \gamma_i \le \mathrm{C}$$

Kernel Trick

$$\max L_D(\gamma_i) = \sum_{i=1}^{n} \gamma_i - \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} S_i \gamma_i \mathrm{K}(d_\mathrm{i}, d_\mathrm{j}) S_j \gamma_j$$

$$s.t. \sum_{i=1}^{n} \gamma_i S_i = 0 \qquad s.t.\ 0 \le \gamma_i \le \mathrm{C}$$

대표적인 Kernel 함수

Polynomial

$$K(x_i, x_j) = \left(\gamma(x_i \cdot x_j) + c\right)^{d},\ c > 0$$

Gaussian (RBF)

$$K(x_i, x_j) = \exp\left(-\gamma \left\| x_i - x_j \right\|^2\right)^{d},\ \sigma \neq 0$$

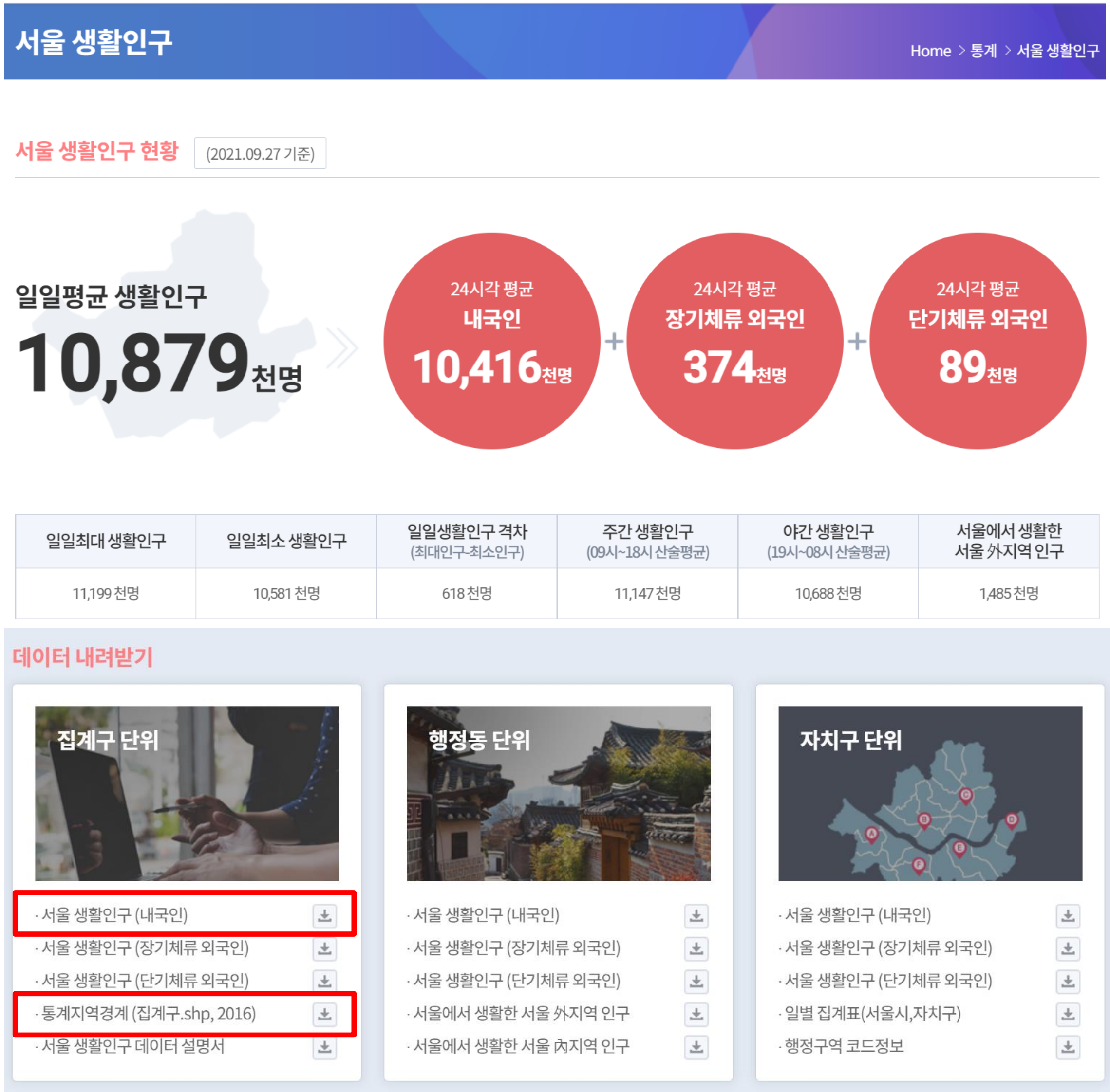
Sigmoid

$$K(x_i, x_j) = \tanh\left(\gamma(x_i \cdot x_j) + b\right),\ a, b > 0$$

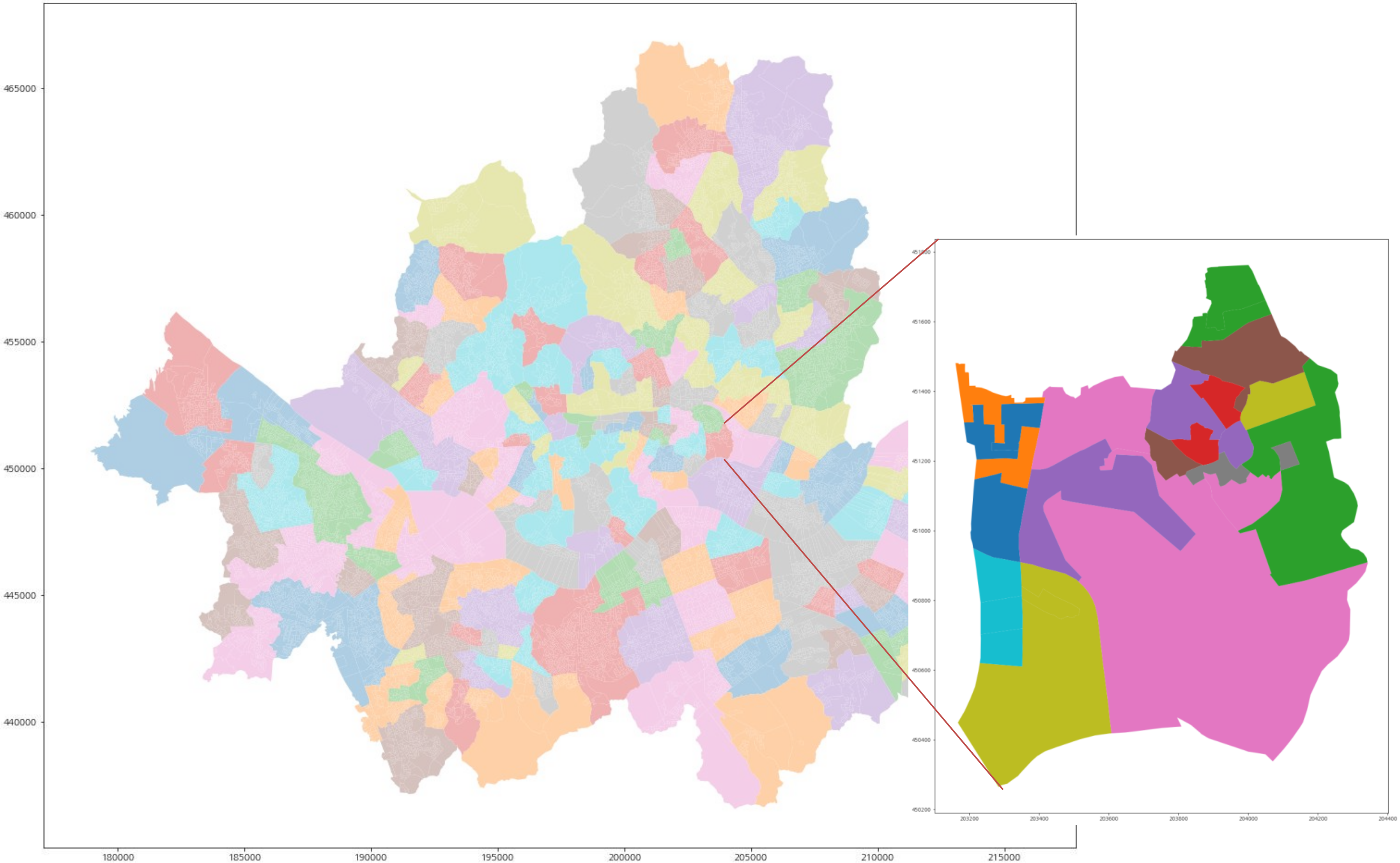
1. SVM Hard margin
2. SVM Soft margin
3. SVM kernel
4. **Data**

서울시 생활인구 데이터

https://data.seoul.go.kr/dataVisual/seoul/seoulLivingPopulation.do

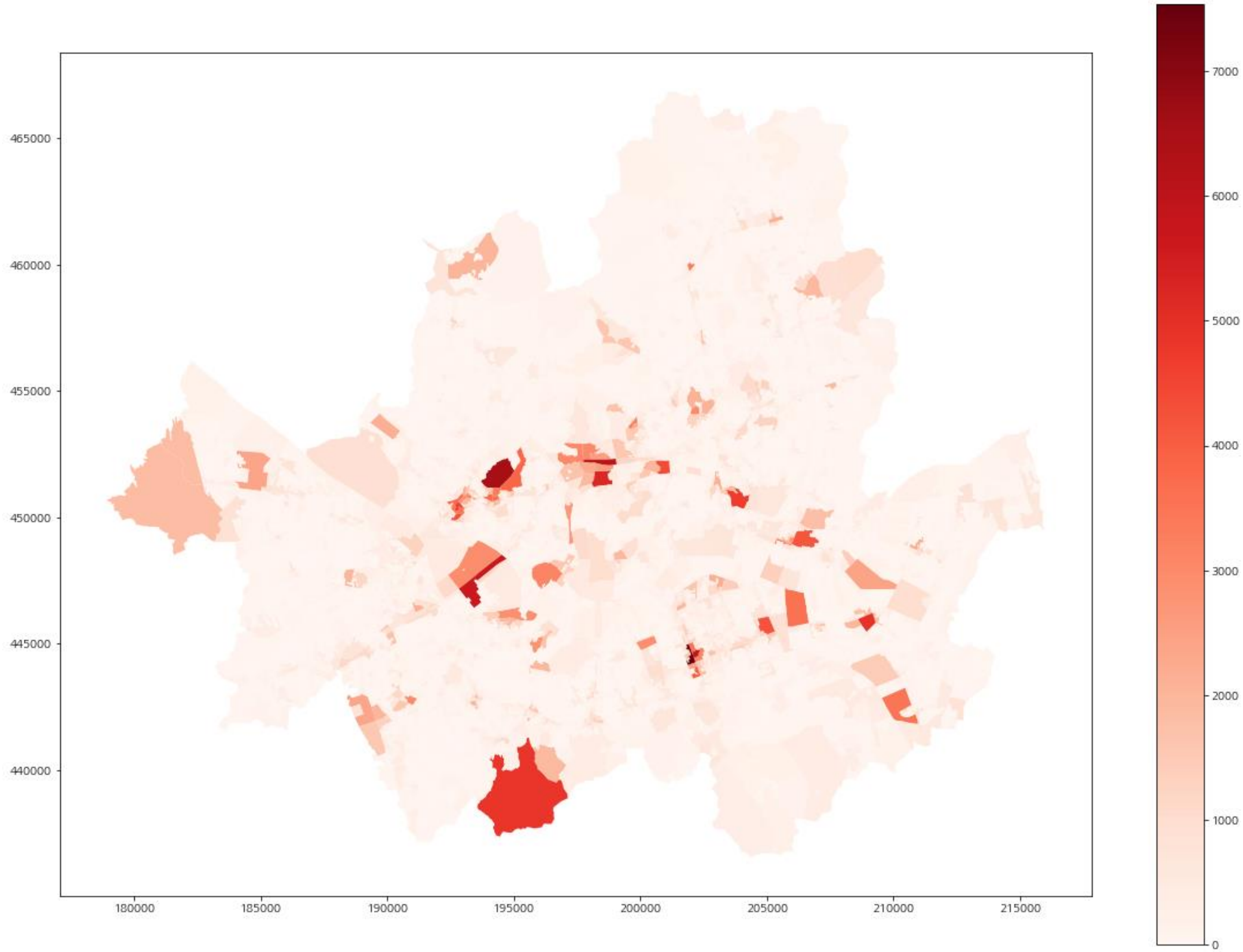


집계구 shp 파일 불러오기, 집계구는 기관마다 다르게 구성 (EPSG:5179)



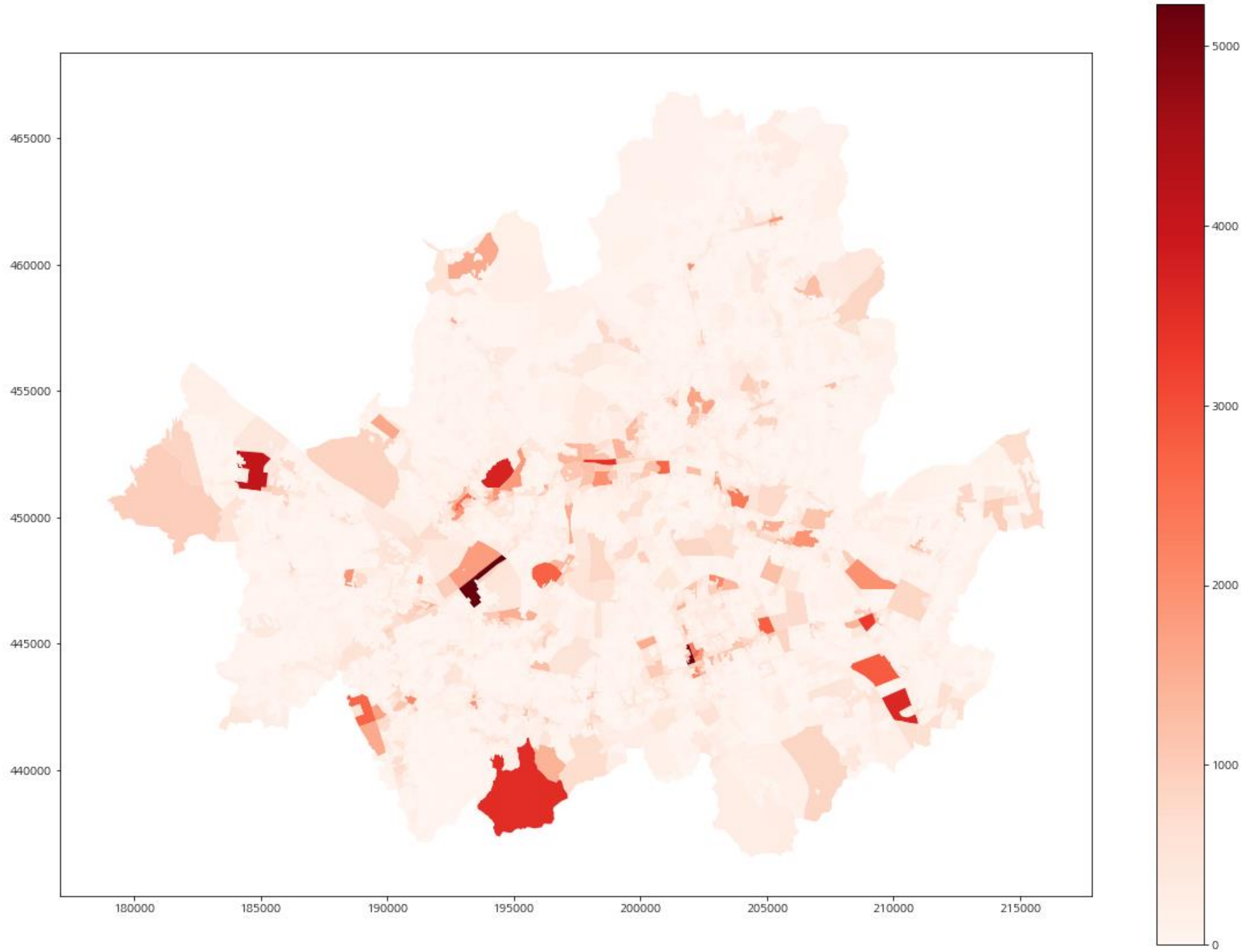
2018년 6월 각 집계구별 / 20대 / 오후 7시~10시 / 생활인구 평균 추출

월~금요일만 합산

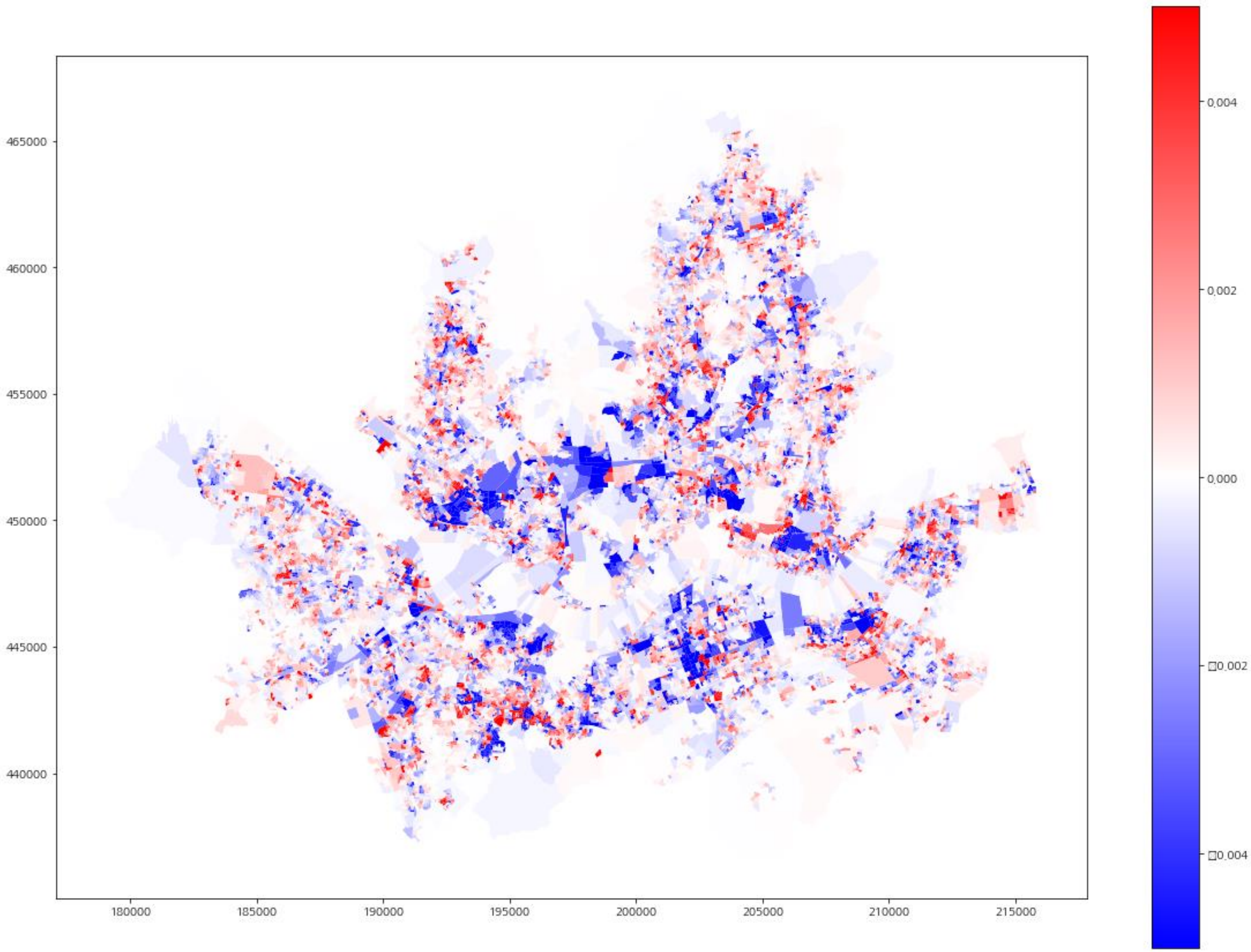


2021년 6월 각 집계구별 / 20대 / 오후 7시~10시 / 생활인구 평균 추출

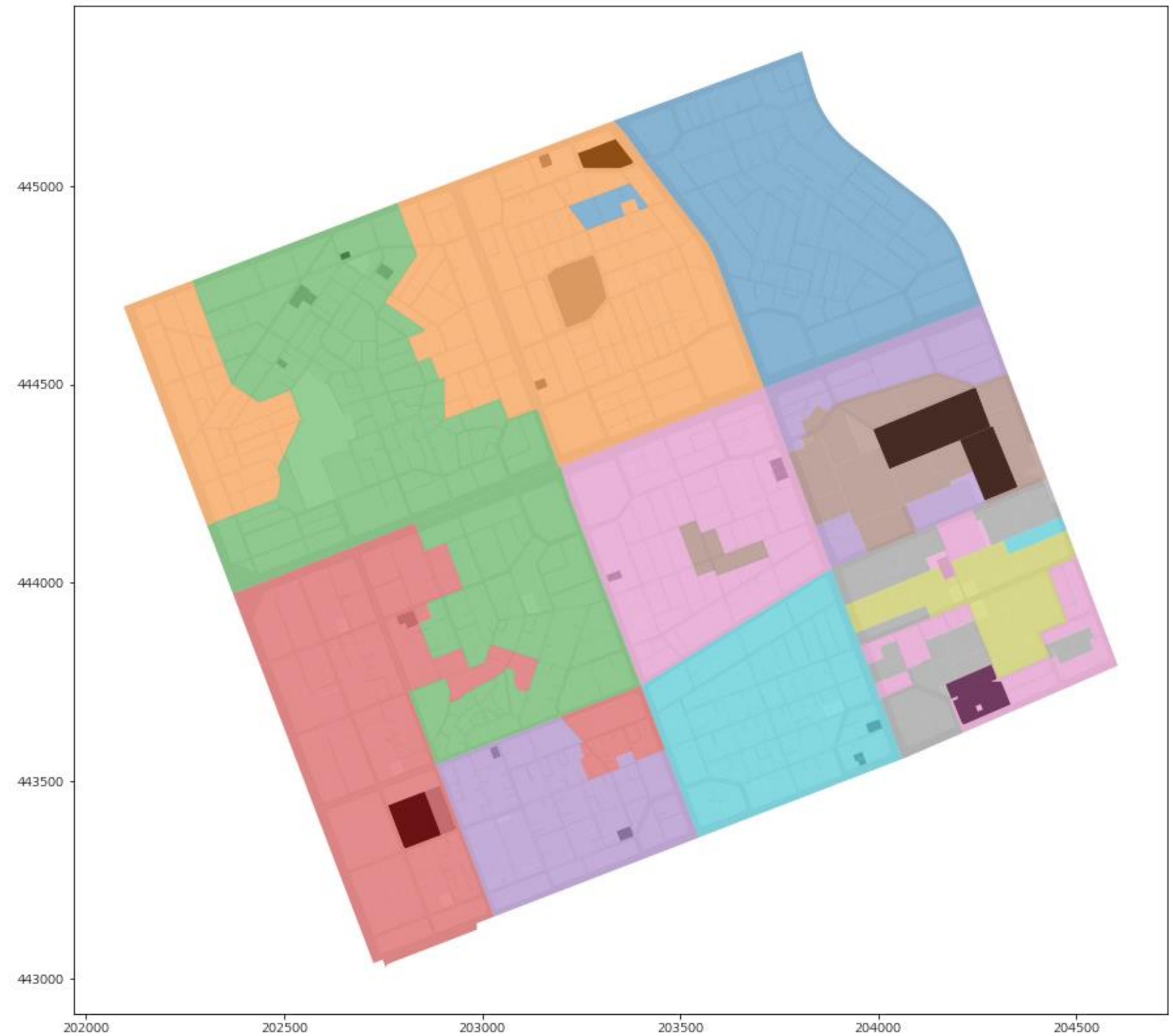
월~금요일만 합산



2021년 – 2018년 20대 ㎡당 생활인구 변화



집계구 단위 지적도


445000
444500
444000
443500
443000
202000
202500
203000
203500
204000
204500

건축물대장 층별 개요 정보 활용

순번	업무구분	서비스명	파일크기(Mb)	데이터제공년월	비고	다운로드
826	건축물대장	지역지구구역 (2021년 08월)	202.4	2021-09-23	설명	다운로드
825	건축물대장	총괄표제부 (2021년 08월)	41.52	2021-09-23	설명	다운로드
824	건축물대장	소유자구분정보 (2021년 08월)	1,047.39	2021-09-23	설명	다운로드
823	건축물대장	전유부 (2021년 08월)	762.87	2021-09-23	설명	다운로드
822	건축물대장	오수정화시설 (2021년 08월)	291.6	2021-09-23	설명	다운로드
821	건축물대장	주택가격 (2021년 08월)	1,898.49	2021-09-23	설명	다운로드
820	건축물대장	전유공용면적 (2021년 08월)	2,061.01	2021-09-23	설명	다운로드
819	건축물대장	부속지번 (2021년 08월)	85.19	2021-09-23	설명	다운로드
818	건축물대장	표제부 (2021년 08월)	726.84	2021-09-23	설명	다운로드
817	건축물대장	층별개요 (2021년 08월)	660.55	2021-09-23	설명	다운로드

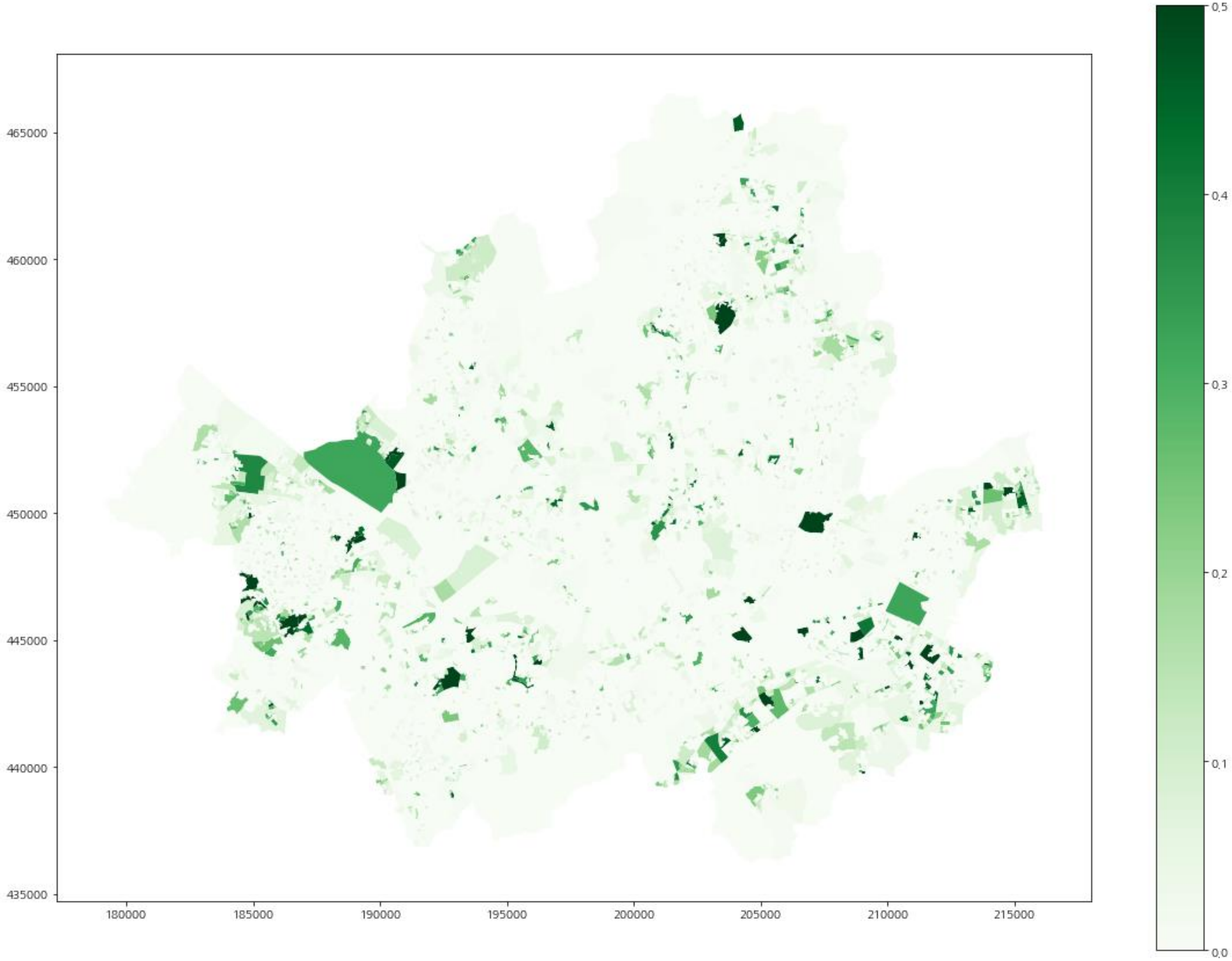
생활인구 독립 변수 (집계구 단위)

1. 지목별 대지면적 비율
2. 건축물 용도별 층별 면적 합계 비율 (집계구 대지면적 합계 대비)

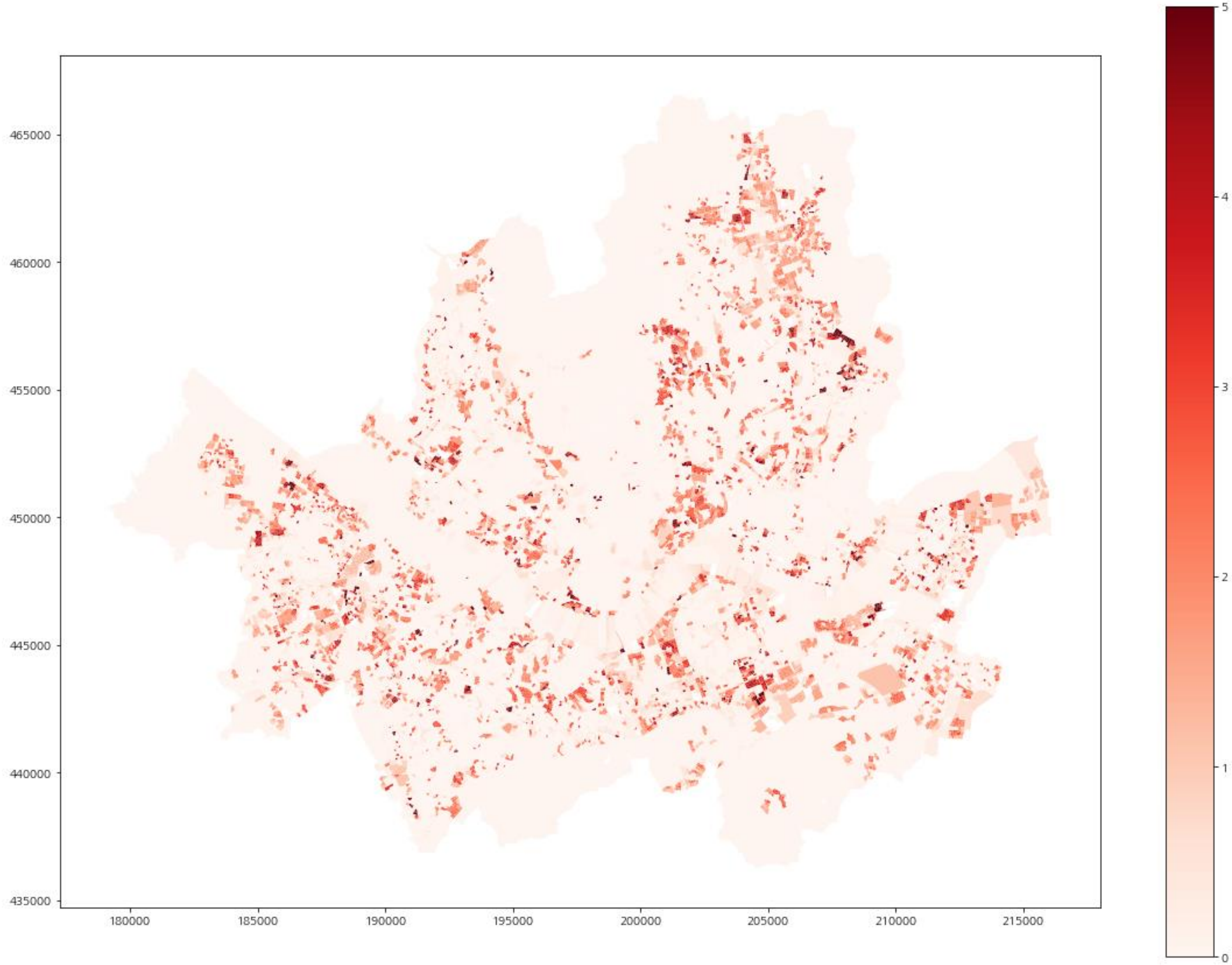
생활인구 종속변수

1. 2018년 6월 평일 20대 19~22시 생활인구 평균
2. 2021년 6월 평일 20대 19~22시 생활인구 평균
3. 21년 – 18년 인구차이 (㎡당)

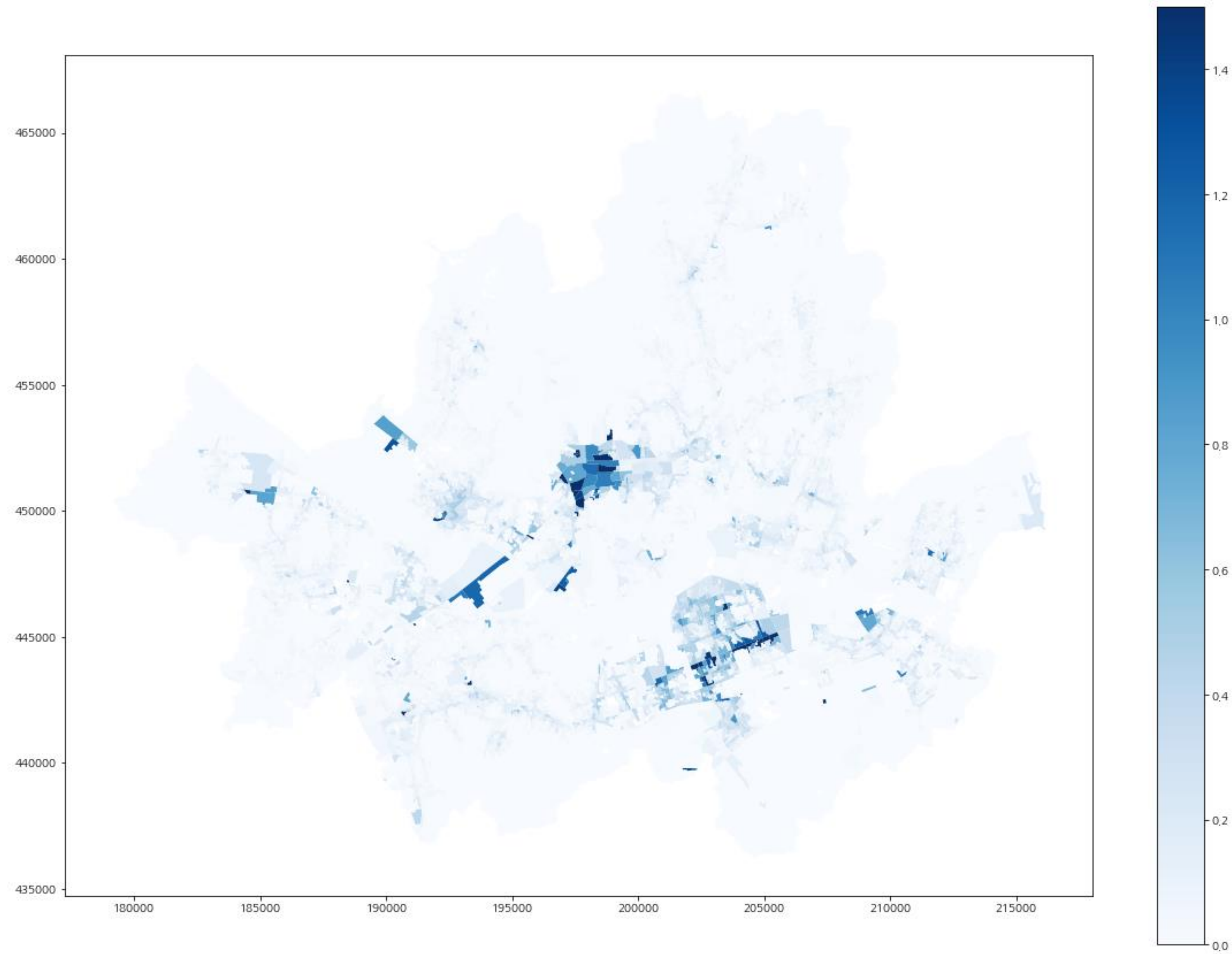
집계구 지목 공(공원) 비중


465000
460000
455000
450000
445000
440000
435000
180000
185000
190000
195000
200000
205000
210000
215000
0.5
0.4
0.3
0.2
0.1
0.0

집계구 아파트 용도 층면적 비율



집계구 사무실 용도 층면적 비율



집계구 고시원 용도 층면적 비율