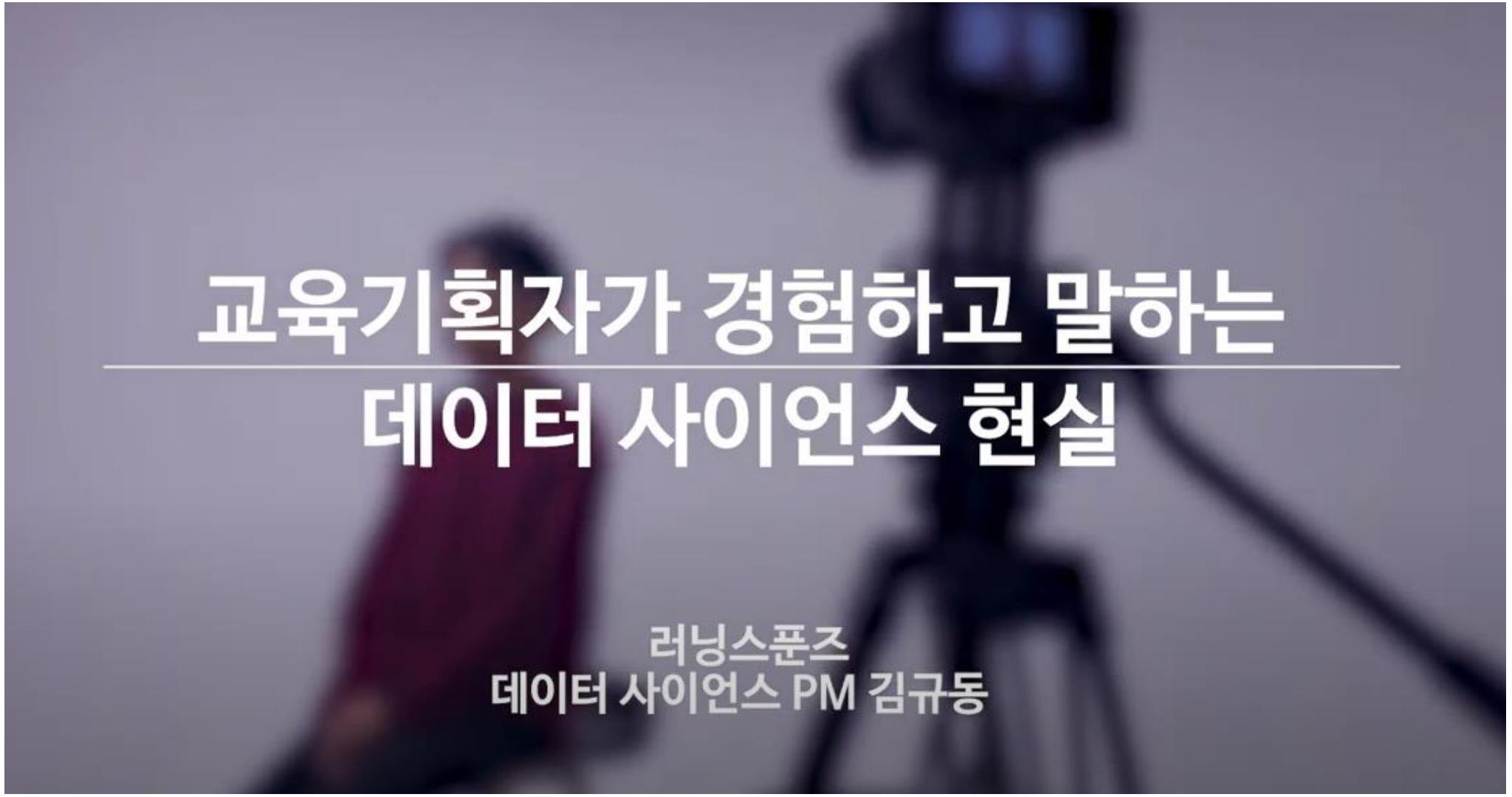
프로젝트 기반 데이터 과학자 양성과정(Data Science)
Machine Learning 및 분석실습

4주차
지도 학습
Naïve Bayes 알고리즘
서포트벡터머신 알고리즘

강사 : 최영진

ICE BREAKING



https://www.youtube.com/watch?v=UnTxzvrw6co&list=PLOkuDBjK90-YQnPQvMqD0y9dwg8pxSi1_&index=1

KNN은 최근접 이웃 알고리즘

❖ KNN 알고리즘

```
1 from sklearn.datasets import load_breast_cancer
2 breast_cancer_data = load_breast_cancer()
```

```
1 import pandas as pd
2 X_Data = pd.DataFrame(breast_cancer_data.data)
3 y = pd.DataFrame(breast_cancer_data.target)
```

```
1 y.head()
```

	0
0	0
1	0
2	0
3	0
4	0

```
1 print(breast_cancer_data.target_names)
2
```

```
['malignant' 'benign']
```

KNN은 최근접 이웃 알고리즘

❖ KNN 알고리즘

```
1  from sklearn.preprocessing import StandardScaler
2
3  scaler = StandardScaler()
4  scaler.fit(X_Data)
5  X_scaled = scaler.transform(X_Data)
6
7  X=pd.DataFrame(X_scaled)
8
9  X.columns = breast_cancer_data.feature_names
10 X.head()
```

	mean radius	mean texture	mean perimeter	mean area	mean smoothness	mean compactness	mean concavity	mean concave points	mean symmetry	mean fractal dimension	...	worst radius	worst texture	worst perimeter	v
0	1.097064	-2.073335	1.269934	0.984375	1.568466	3.283515	2.652874	2.532475	2.217515	2.255747	...	1.886690	-1.359293	2.303601	2.00
1	1.829821	-0.353632	1.685955	1.908708	-0.826962	-0.487072	-0.023846	0.548144	0.001392	-0.868652	...	1.805927	-0.369203	1.535126	1.89
2	1.579888	0.456187	1.566503	1.558884	0.942210	1.052926	1.363478	2.037231	0.939685	-0.398008	...	1.511870	-0.023974	1.347475	1.45
3	-0.768909	0.253732	-0.592687	-0.764464	3.283553	3.402909	1.915897	1.451707	2.867383	4.910919	...	-0.281464	0.133984	-0.249939	-0.55
4	1.750297	-1.151816	1.776573	1.826229	0.280372	0.539340	1.371011	1.428493	-0.009560	-0.562450	...	1.298575	-1.466770	1.338539	1.22

5 rows × 30 columns

KNN은 최근접 이웃 알고리즘

❖ KNN 알고리즘

```
from sklearn.model_selection import train_test_split
X_train, X_test , y_train, y_test = train_test_split(X, y, test_size = 0.3, random_state = 42)
```

```
print(len(X_train))
print(len(X_test))
print(len(y_train))
print(len(y_test))
```

```
398
171
398
171
```

KNN은 최근접 이웃 알고리즘

❖ KNN 알고리즘

```
1 from sklearn.neighbors import KNeighborsClassifier
2 classifier = KNeighborsClassifier(n_neighbors = 3)
```

```
1 classifier.fit(X_train, y_train)
```

```
C:₩Users₩choi₩Anaconda3₩envs₩tensor-gpu₩lib₩site-packages₩ipykernel_launcher.py:1: DataConversionWarning: A column-vector y was passed when a 1d array was expected. Please change the shape of y to (n_samples, ), for example using ravel().
  """Entry point for launching an IPython kernel.
```

```
KNeighborsClassifier(algorithm='auto', leaf_size=30, metric='minkowski',
                     metric_params=None, n_jobs=None, n_neighbors=3, p=2,
                     weights='uniform')
```

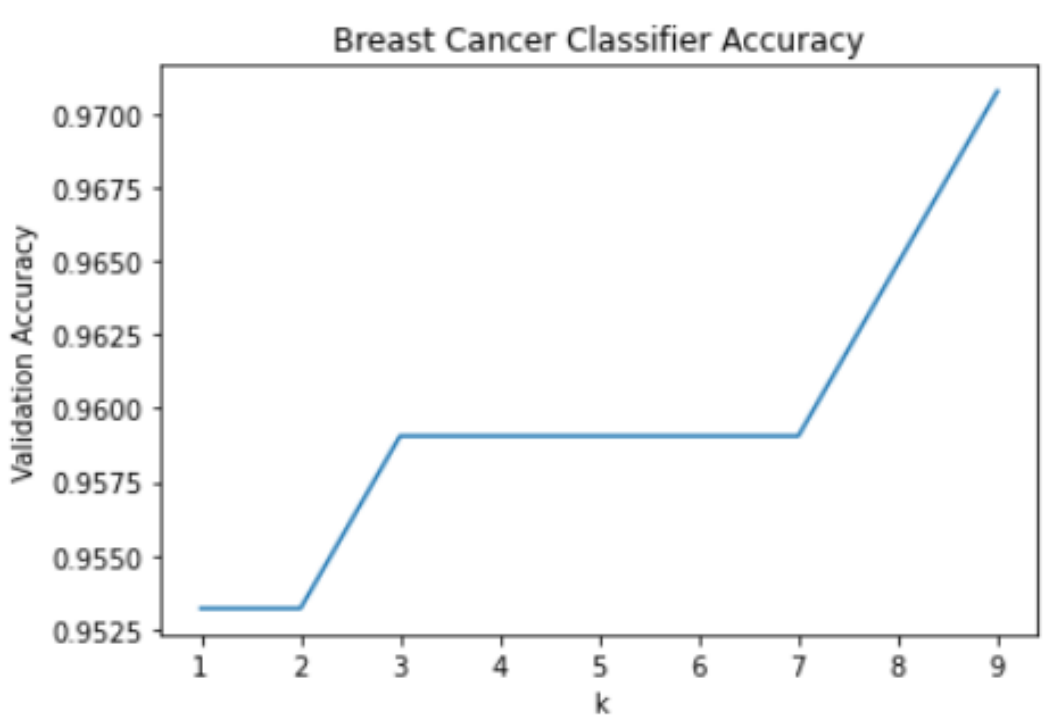
```
1 print(classifier.score(X_test, y_test))
```

```
0.9590643274853801
```

KNN은 최근접 이웃 알고리즘

❖ KNN 알고리즘

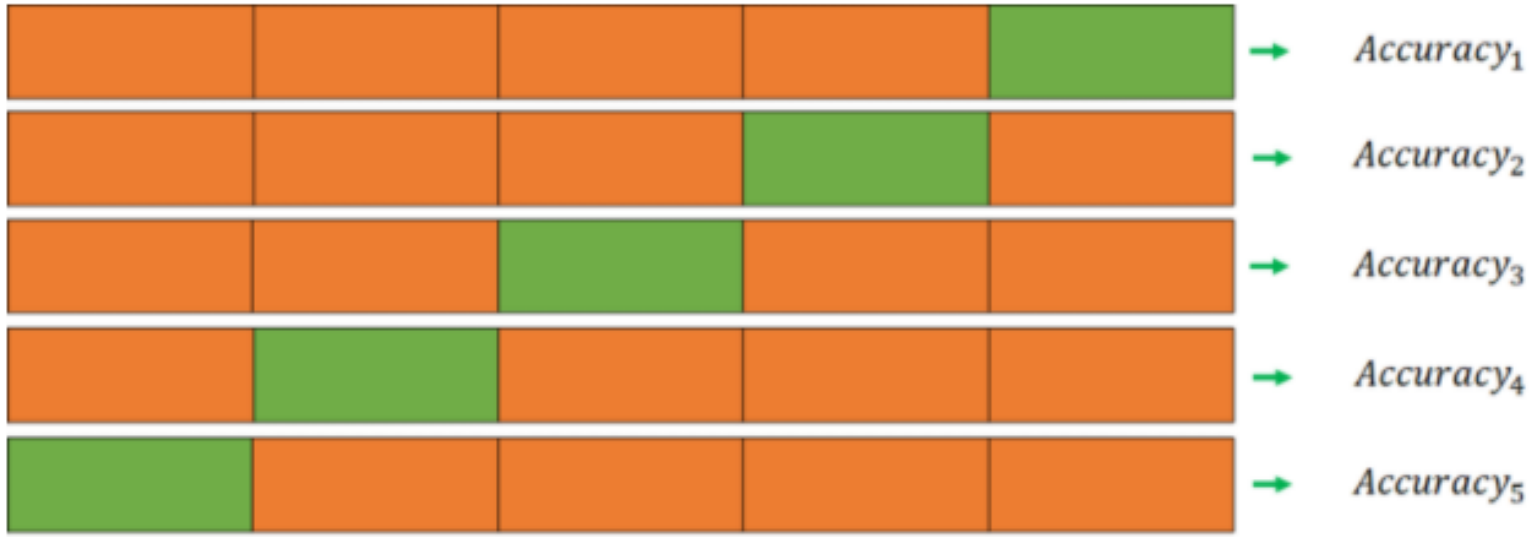
```
import matplotlib.pyplot as plt
k_list = range(1,10)
accuracies = []
for k in k_list:
    classifier = KNeighborsClassifier(n_neighbors = k)
    classifier.fit(X_train, y_train)
    accuracies.append(classifier.score(X_test, y_test))
plt.plot(k_list, accuracies)
plt.xlabel("k")
plt.ylabel("Validation Accuracy")
plt.title("Breast Cancer Classifier Accuracy")
plt.show()
```



KNN은 최근접 이웃 알고리즘

❖ KNN 알고리즘

- k-겹 교차 검증(k-fold cross validation)



$$Accuracy = Average(Accuracy_1, \cdots, Accuracy_k)$$

KNN은 최근접 이웃 알고리즘

❖ KNN 알고리즘

```
1  from sklearn.model_selection import cross_val_score # K-fold cross-validation module
2
3  scores = cross_val_score(classifier,X, y, cv=5, scoring='accuracy')
4  print(scores)
5  print(scores.mean())
6
```

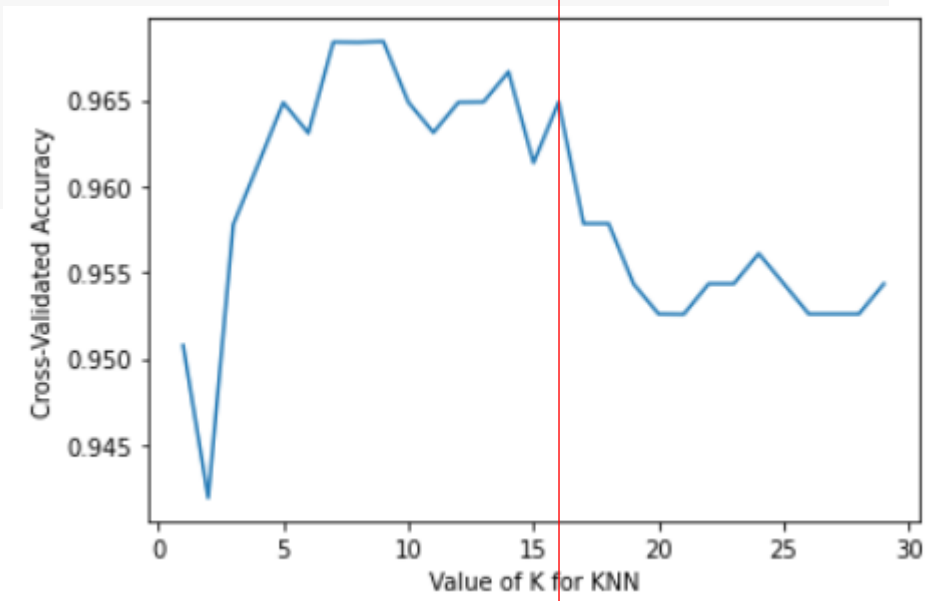
```
[0.97368421 0.95614035 0.98245614 0.94736842 0.92920354]
0.9577705325260053
```

KNN은 최근접 이웃 알고리즘

❖ KNN 알고리즘

```python
from sklearn import model_selection
import matplotlib.pyplot as plt

k_range = range(1, 30)

k_scores = []

for k in k_range:
    knn = KNeighborsClassifier(n_neighbors=k)
    scores = model_selection.cross_val_score(knn, X, y, cv=5, scoring='accuracy')
    k_scores.append(scores.mean())

#Visualizing data
plt.plot(k_range, k_scores)
plt.xlabel('Value of K for KNN')
plt.ylabel('Cross-Validated Accuracy')
plt.show()
```

underfitting



목차

1. Naïve Bayes

❖ Naïve Bayes란?

- **나이브베이즈(Naïve Bayes)는**
- **기계학습(Machine Learning)에서**
  지도학습(Supervised Learning) 알고리즘으로써 주로분류(Classification)의 목적으로 사용
- **속성들 사이의 독립을 가정하는 베이즈정리(Bayes theorem)를 적용한 확률적 분류기법**

- 스팸 이메일 필터링과 같은 텍스트 분류
- 컴퓨터 네트워크에서 침입이나 비정상행위 탐지
- 일련의 관찰된 증상에 대한 의학적 질병 진단

1. Naïve Bayes

❖ Naïve Bayes란?

Whether	Play
Sunny	No
Sunny	No
Overcast	Yes
Rainy	Yes
Rainy	Yes
Rainy	No
Overcast	Yes
Sunny	No
Sunny	Yes
Rainy	Yes
Sunny	Yes
Overcast	Yes
Overcast	Yes
Rainy	No

Frequency Table

Whether	No	Yes
Overcast		4
Sunny	2	3
Rainy	3	2
Total	5	9

Likelihood Table 1

Whether	No	Yes		
Overcast		4	=4/14	0.29
Sunny	2	3	=5/14	0.36
Rainy	3	2	=5/14	0.36
Total	5	9		
	=5/14	=9/14		
	0.36	0.64		

Likelihood Table 2

Whether	No	Yes	Posterior Probability for No	Posterior Probability for Yes
Overcast		4	0/5=0	4/9=0.44
Sunny	2	3	2/5=0.4	3/9=0.33
Rainy	3	2	3/5=0.6	2/9=0.22
Total	5	9		

1. Naïve Bayes

❖ 베이즈정리(Bayes theorem)

$$P(B|A) = \frac{P(A \cap B)}{P(A)}$$ 조건부 확률

$$P(A \cap B) = P(A)P(B|A) = P(B)P(A|B)$$ 확률의 곱셈정리

$$P(B|A) = \frac{P(A)P(B|A)}{P(A)} = \frac{P(B)P(A|B)}{P(A)}$$ 조건부 확률식의 확률 곱셈정리로 치환

- A: 원인, B: 결과
- P(A): 원인이 발생할 사전 확률
- P(B): 결과가 발생할 사전확률
- P(B|A): 원인이 발생했을 때, 결과가 발생할 확률
- P(A|B): 결과가 발생했을 때, 원인이 발생할 확률

1. Naïve Bayes

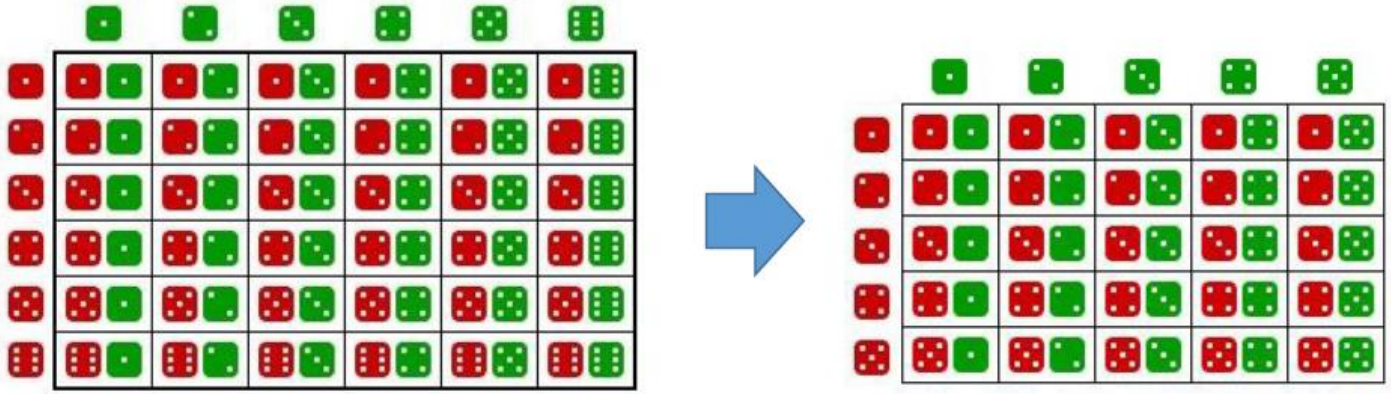
❖ 베이즈정리(Bayes theorem)

- 베이즈정리(Bayes theorem)
  - 1740년대 토마스베이즈(Thomas Bayes)가 정립한 조건부 확률에 대한 수학적정리
  - 베이즈정리는 베이즈룰(Bayes Rule), 베이즈법칙(Bayes Law)으로도 불림
- 두 확률변수의 사전확률과 사후확률 사이의 관계를 나타내는 정리
  - 사전 확률의 정보를 이용해 사후 확률을 추정
- 사전확률(Prior probability)
  - 가지고 있는 정보를 기초로 정한 초기확률
- 사후확률(Posterior probability)
  - 결과가 발생했다는 조건에서 어떤 원인이 발생했을 확률
- 우도(Likelihood)
  - 원인이 발생했다는 조건에서 결과가 발생했을 확률

1. Naïve Bayes

❖ 조건부확률(Conditional Probability)

- 조건부 확률 P(B|A): 사건 A가 일어난 상태에서 사건B가 일어날 확률
- 예제
  - 주사위2개를 던졌는데 6은 인정하지않는 상태에서 두 주사위의 차이가 2일 확률은? P(B|A)
  - 사건A: 주사위 중 하나라도 6이 나오는 것은 인정하지않는 사건
  - 사건B: 주사위 2개를 던졌을 때 두 주사위의 차이가 2인 사건



표본공간이 주사위 2개를 던졌을 때 나올 수 있는 모든 결과의 집합

주사위를 2개 던졌는데 그 중 6을 인정하지 않는 모든 결과의 집합

1. Naïve Bayes

❖ 조건부확률(Conditional Probability)

- **주사위 2개를 던졌는데 6은 인정하지않는 상태에서 두 주사위의 차이가 2일 확률은? P(B|A)**
  - 각 사건의 확률값 보정: 크기가 줄어든 표본공간의 전체확률 값이 1이 되어야함

$$P(B|A) = \frac{P(A \cap B)}{P(A)}$$

  - 원래 표본공간은 36개 이고 주사위6은 인정하지않는 조건에 해당하는 집합은 원소가 25개

조건부 확률 P(B|A)

$$P(A \cap B) = P(\{(1,3),(3,1),(2,4),(4,2),(3,5),(5,3)\}) = \frac{6}{36} = \frac{1}{6} \qquad P(A) = \frac{25}{36}$$

$$P(B|A) = \frac{P(A \cap B)}{P(A)} = \frac{\frac{1}{6}}{\frac{25}{36}} = \frac{6}{25}$$

1. Naïve Bayes

❖ Naïve Bayes

▪ 장점

- 우도 테이블 하나만 있으면 분류가 가능
- 계산 복잡성이 낮음 / 간단하고, 빠르며, 정확
- 자연언어처리 기법으로 각광받아 연속형보다 이산형 데이터에서 성능이 좋음

▪ 단점

- 모든 특징이 동등하게 중요하고 독립이라는 가정을 했지만, 이 가정이 잘못된 경우(feature 간의 독립성)
- 이메일 메시지를 감시해서 스팸을 식별하려고 할 때, A라는 특징이 B 특징보다 더 중요할 때가 있으나, 이를 무시하고 계산
- 수치 특징이 많은 데이터셋에는 이상적이지 않음
- 추정된 확률이 예측된 클래스보다 덜 신뢰

1. Naïve Bayes

❖ 예제를 이용한Naïve Bayes 실습

Outlook	Temperature	Humidity	PlayTennis
Sunny	Hot	High	No
Sunny	Hot	High	No
Overcast	Hot	High	Yes
Rain	Mild	High	Yes
Rain	Cool	Normal	Yes
Rain	Cool	Normal	No
Overcast	Cool	Normal	Yes
Sunny	Mild	High	No
Sunny	Cool	Normal	Yes
Rain	Mild	Normal	Yes
Sunny	Mild	Normal	Yes
Overcast	Mild	High	Yes
Overcast	Hot	Normal	Yes
Rain	Mild	High	No

진행과정

1. 데이터셋을 이용해 빈도테이블 생성
2. 데이터셋을 이용해 우도(Likelihood) 테이블생성
3. 나이브베이즈방정식을 사용해 각 클래스의 사후확률계산

1. Naïve Bayes

❖ 예제를 이용한Naïve Bayes 실습

1. Outlook 데이터를 이용

Outlook	PlayTennis
Sunny	No
Sunny	No
Overcast	Yes
Rain	Yes
Rain	Yes
Rain	No
Overcast	Yes
Sunny	No
Sunny	Yes
Rain	Yes
Sunny	Yes
Overcast	Yes
Overcast	Yes
Rain	No

빈도 테이블

	No	Yes
Overcast	0	4
Rain	2	3
Sunny	3	2
Total	5	9

우도(Likelihood) 테이블

	No	Yes	
Overcast	0	4	0.29
Rain	2	3	0.36
Sunny	3	2	0.36
Total	5	9	
	0.36	0.64	

$$\frac{4}{14} = 0.28571 \ldots = 0.29 \quad P(Overcast)$$

$$\frac{5}{14} = 0.35714 \ldots = 0.36 \quad P(Rain)$$

$$\frac{5}{14} = 0.35714 \ldots = 0.36 \quad P(Sunny)$$

1. Naïve Bayes

❖ 예제를 이용한Naïve Bayes 실습

1. Outlook 데이터를 이용

Outlook	PlayTennis
Sunny	No
Sunny	No
Overcast	Yes
Rain	Yes
Rain	Yes
Rain	No
Overcast	Yes
Sunny	No
Sunny	Yes
Rain	Yes
Sunny	Yes
Overcast	Yes
Overcast	Yes
Rain	No

빈도 테이블

	No	Yes
Overcast	0	4
Rain	2	3
Sunny	3	2
Total	5	9

우도(Likelihood) 테이블

	No	Yes	
Overcast	0	4	0.29
Rain	2	3	0.36
Sunny	3	2	0.36
Total	5	9	
	0.36	0.64	

$P(No)$

$$\frac{5}{14} = 0.35714 \ldots = 0.36$$

$P(Yes)$

$$\frac{9}{14} = 0.64285 \ldots = 0.64$$

1. Naïve Bayes

❖ 예제를 이용한Naïve Bayes 실습

1. Outlook 데이터를 이용

Outlook	PlayTennis
Sunny	No
Sunny	No
Overcast	Yes
Rain	Yes
Rain	Yes
Rain	No
Overcast	Yes
Sunny	No
Sunny	Yes
Rain	Yes
Sunny	Yes
Overcast	Yes
Overcast	Yes
Rain	No

빈도 테이블

	No	Yes
Overcast	0	4
Rain	2	3
Sunny	3	2
Total	5	9

우도(Likelihood) 테이블

	No	Yes	
Overcast	0	4	0.29
Rain	2	3	0.36
Sunny	3	2	0.36
Total	5	9	
	0.36	0.64	

$P(Overcast|Yes)$

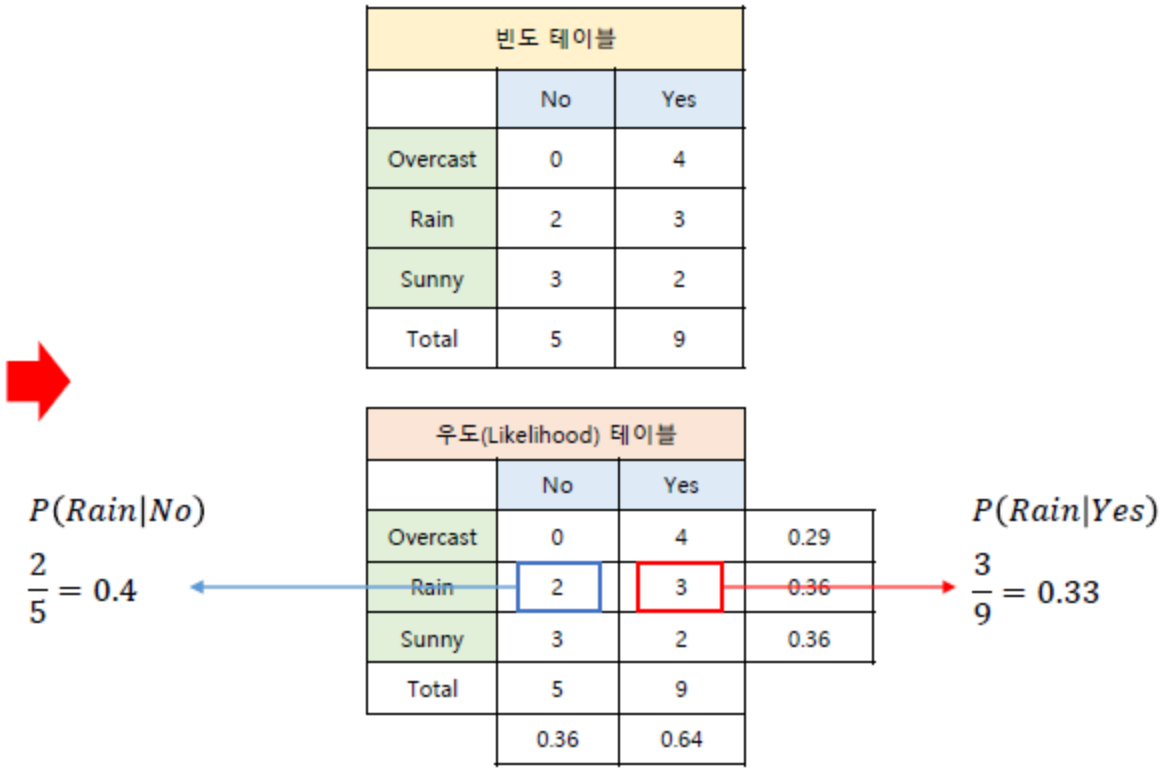
$\frac{4}{9} = 0.44$

1. Naïve Bayes

❖ 예제를 이용한Naïve Bayes 실습

1. Outlook 데이터를 이용

Outlook	PlayTennis
Sunny	No
Sunny	No
Overcast	Yes
Rain	Yes
Rain	Yes
Rain	No
Overcast	Yes
Sunny	No
Sunny	Yes
Rain	Yes
Sunny	Yes
Overcast	Yes
Overcast	Yes
Rain	No



1. Naïve Bayes

❖ 예제를 이용한Naïve Bayes 실습

1. Outlook 데이터를 이용

Outlook	PlayTennis
Sunny	No
Sunny	No
Overcast	Yes
Rain	Yes
Rain	Yes
Rain	No
Overcast	Yes
Sunny	No
Sunny	Yes
Rain	Yes
Sunny	Yes
Overcast	Yes
Overcast	Yes
Rain	No

빈도 테이블

	No	Yes
Overcast	0	4
Rain	2	3
Sunny	3	2
Total	5	9

우도(Likelihood) 테이블

	No	Yes	
Overcast	0	4	0.29
Rain	2	3	0.36
Sunny	3	2	0.36
Total	5	9	
	0.36	0.64	

$P(Sunny|No)$

$\frac{3}{5} = 0.6$

$P(Sunny|Yes)$

$\frac{2}{9} = 0.22$

1. Naïve Bayes

❖ 예제를 이용한Naïve Bayes 실습

1. Outlook 데이터를 이용

- $P(No|Sunny) = \frac{P(No)P(Sunny|No)}{P(Sunny)}$
  - $P(Sunny|No) = \frac{3}{5} = 0.6$
  - $P(No) = \frac{5}{14} = 0.36$
  - $P(Sunny) = \frac{5}{14} = 0.36$
  - $P(No|Sunny) = \frac{0.36*0.6}{0.36} = 0.6$

- $P(Yes|Sunny) = \frac{P(Yes)P(Sunny|Yes)}{P(Sunny)}$
  - $P(Sunny|Yes) = \frac{2}{9} = 0.22$
  - $P(Yes) = \frac{9}{14} = 0.64$
  - $P(Sunny) = \frac{5}{14} = 0.36$
  - $P(Yes|Sunny) = \frac{0.64*0.22}{0.36} = 0.39$

- $P(No|Rain) = \frac{P(No)P(Rain|No)}{P(Rain)}$
  - $P(Rain|No) = \frac{2}{5} = 0.4$
  - $P(No) = \frac{5}{14} = 0.36$
  - $P(Rain) = \frac{5}{14} = 0.36$
  - $P(No|Rain) = \frac{0.36*0.4}{0.36} = 0.4$

- $P(Yes|Rain) = \frac{P(Yes)P(Rain|Yes)}{P(Rain)}$
  - $P(Rain|Yes) = \frac{3}{9} = 0.33$
  - $P(Yes) = \frac{9}{14} = 0.64$
  - $P(Rain) = \frac{5}{14} = 0.36$
  - $P(Yes|Sunny) = \frac{0.64*0.33}{0.36} = 0.59$

- $P(Yes|Overcast) = \frac{P(Yes)P(Overcast|Yes)}{P(Overcast)}$
  - $P(Overcast|Yes) = \frac{4}{9} = 0.44$
  - $P(Yes) = \frac{9}{14} = 0.64$
  - $P(Overcast) = \frac{4}{14} = 0.29$
  - $P(Yes|Overcast) = \frac{0.64*0.44}{0.29} = 0.97$

1. Naïve Bayes

❖ 예제를 이용한Naïve Bayes 실습

2. Temperature 데이터를 이용

Temperature	PlayTennis
Hot	No
Hot	No
Hot	Yes
Mild	Yes
Cool	Yes
Cool	No
Cool	Yes
Mild	No
Cool	Yes
Mild	Yes
Mild	Yes
Mild	Yes
Hot	Yes
Mild	No

빈도 테이블

	No	Yes
Cool	1	3
Hot	2	2
Mild	2	4
Total	5	9

우도(Likelihood) 테이블

	No	Yes	
Cool	1	3	0.29
Hot	2	2	0.29
Mild	2	4	0.43
Total	5	9	
	0.36	0.64	

$\frac{4}{14} = 0.28571 \ldots = 0.29 \quad P(Cool)$

$\frac{4}{14} = 0.28571 \ldots = 0.29 \quad P(Hot)$

$\frac{6}{14} = 0.42857 \ldots = 0.43 \quad P(Mild)$

1. Naïve Bayes

❖ 예제를 이용한Naïve Bayes 실습

2. Temperature 데이터를 이용

Temperature	PlayTennis
Hot	No
Hot	No
Hot	Yes
Mild	Yes
Cool	Yes
Cool	No
Cool	Yes
Mild	No
Cool	Yes
Mild	Yes
Mild	Yes
Mild	Yes
Hot	Yes
Mild	No

빈도 테이블

	No	Yes
Cool	1	3
Hot	2	2
Mild	2	4
Total	5	9

우도(Likelihood) 테이블

	No	Yes	
Cool	1	3	0.29
Hot	2	2	0.29
Mild	2	4	0.43
Total	5	9	
	0.36	0.64	

$P(No)$

$\frac{5}{14} = 0.35714 \ldots = 0.36$

$P(Yes)$

$\frac{9}{14} = 0.64285 \ldots = 0.64$

1. Naïve Bayes

❖ 예제를 이용한Naïve Bayes 실습

2. Temperature 데이터를 이용

Temperature	PlayTennis
Hot	No
Hot	No
Hot	Yes
Mild	Yes
Cool	Yes
Cool	No
Cool	Yes
Mild	No
Cool	Yes
Mild	Yes
Mild	Yes
Mild	Yes
Hot	Yes
Mild	No

빈도 테이블

	No	Yes
Cool	1	3
Hot	2	2
Mild	2	4
Total	5	9

$P(Cool|No)$

$\frac{1}{5} = 0.2$

우도(Likelihood) 테이블

	No	Yes	
Cool	1	3	0.29
Hot	2	2	0.29
Mild	2	4	0.43
Total	5	9	
	0.36	0.64	

$P(Cool|Yes)$

$\frac{3}{9} = 0.33$

1. Naïve Bayes

❖ 예제를 이용한Naïve Bayes 실습

2. Temperature 데이터를 이용

Temperature	PlayTennis
Hot	No
Hot	No
Hot	Yes
Mild	Yes
Cool	Yes
Cool	No
Cool	Yes
Mild	No
Cool	Yes
Mild	Yes
Mild	Yes
Mild	Yes
Hot	Yes
Mild	No

빈도 테이블

	No	Yes
Cool	1	3
Hot	2	2
Mild	2	4
Total	5	9

우도(Likelihood) 테이블

	No	Yes	
Cool	1	3	0.29
Hot	2	2	0.29
Mild	2	4	0.43
Total	5	9	
	0.36	0.64	

$P(Hot|No)$

$\frac{2}{5} = 0.4$

$P(Hot|Yes)$

$\frac{2}{9} = 0.22$

1. Naïve Bayes

❖ 예제를 이용한Naïve Bayes 실습

2. Temperature 데이터를 이용

Temperature	PlayTennis
Hot	No
Hot	No
Hot	Yes
Mild	Yes
Cool	Yes
Cool	No
Cool	Yes
Mild	No
Cool	Yes
Mild	Yes
Mild	Yes
Mild	Yes
Hot	Yes
Mild	No

빈도 테이블

	No	Yes
Cool	1	3
Hot	2	2
Mild	2	4
Total	5	9

우도(Likelihood) 테이블

	No	Yes	
Cool	1	3	0.29
Hot	2	2	0.29
Mild	2	4	0.43
Total	5	9	
	0.36	0.64	

$P(Mild|No)$

$\frac{2}{5} = 0.4$

$P(Mild|Yes)$

$\frac{4}{9} = 0.44$

1. Naïve Bayes

❖ 예제를 이용한Naïve Bayes 실습

2. Temperature 데이터를 이용

- $P(No|Cool) = \frac{P(No)P(Cool|No)}{P(Cool)}$
  - $P(Cool|No) = \frac{1}{5} = 0.2$
  - $P(No) = \frac{5}{14} = 0.36$
  - $P(Cool) = \frac{4}{14} = 0.29$
  - $P(No|Cool) = \frac{0.36*0.2}{0.29} = 0.25$

- $P(No|Hot) = \frac{P(No)P(Hot|No)}{P(Hot)}$
  - $P(Hot|No) = \frac{2}{5} = 0.4$
  - $P(No) = \frac{5}{14} = 0.36$
  - $P(Hot) = \frac{4}{14} = 0.29$
  - $P(No|Hot) = \frac{0.36*0.4}{0.29} = 0.5$

- $P(No|Mild) = \frac{P(No)P(Mild|No)}{P(Mild)}$
  - $P(Mild|No) = \frac{2}{5} = 0.4$
  - $P(No) = \frac{5}{14} = 0.36$
  - $P(Mild) = \frac{6}{14} = 0.43$
  - $P(No|Mild) = \frac{0.36*0.4}{0.43} = 0.33$

- $P(Yes|Cool) = \frac{P(Yes)P(Cool|Yes)}{P(Cool)}$
  - $P(Cool|Yes) = \frac{3}{9} = 0.33$
  - $P(Yes) = \frac{9}{14} = 0.64$
  - $P(Cool) = \frac{4}{14} = 0.29$
  - $P(Yes|Cool) = \frac{0.64*0.33}{0.29} = 0.73$

- $P(Yes|Hot) = \frac{P(Yes)P(Hot|Yes)}{P(Hot)}$
  - $P(Hot|Yes) = \frac{2}{9} = 0.22$
  - $P(Yes) = \frac{9}{14} = 0.64$
  - $P(Hot) = \frac{4}{14} = 0.29$
  - $P(Yes|Hot) = \frac{0.64*0.22}{0.29} = 0.49$

- $P(Yes|Cool) = \frac{P(Yes)P(Mild|Yes)}{P(Cool)}$
  - $P(Mild|Yes) = \frac{4}{9} = 0.44$
  - $P(Yes) = \frac{9}{14} = 0.64$
  - $P(Mild) = \frac{6}{14} = 0.43$
  - $P(Yes|Mild) = \frac{0.64*0.44}{0.43} = 0.65$

1. Naïve Bayes

❖ 예제를 이용한Naïve Bayes 실습

3. Humidity 데이터를 이용

Humidity	PlayTennis
High	No
High	No
High	Yes
High	Yes
Normal	Yes
Normal	No
Normal	Yes
High	No
Normal	Yes
Normal	Yes
Normal	Yes
High	Yes
Normal	Yes
High	No

빈도 테이블

	No	Yes
High	4	3
Normal	1	6
Total	5	9

우도(Likelihood) 테이블

	No	Yes	
High	4	3	0.5
Normal	1	6	0.5
Total	5	9	
	0.36	0.64	

$\frac{7}{14} = 0.5$ $P(High)$

$\frac{7}{14} = 0.5$ $P(Normal)$

1. Naïve Bayes

❖ 예제를 이용한Naïve Bayes 실습

3. Humidity 데이터를 이용

Humidity	PlayTennis
High	No
High	No
High	Yes
High	Yes
Normal	Yes
Normal	No
Normal	Yes
High	No
Normal	Yes
Normal	Yes
Normal	Yes
High	Yes
Normal	Yes
High	No

빈도 테이블

	No	Yes
High	4	3
Normal	1	6
Total	5	9

우도(Likelihood) 테이블

	No	Yes	
High	4	3	0.5
Normal	1	6	0.5
Total	5	9	
	0.36	0.64	

$P(No)$

$\frac{5}{14} = 0.35714 \ldots = 0.36$

$P(Yes)$

$\frac{9}{14} = 0.64285 \ldots = 0.64$

1. Naïve Bayes

❖ 예제를 이용한Naïve Bayes 실습

3. Humidity 데이터를 이용

Humidity	PlayTennis
High	No
High	No
High	Yes
High	Yes
Normal	Yes
Normal	No
Normal	Yes
High	No
Normal	Yes
Normal	Yes
Normal	Yes
High	Yes
Normal	Yes
High	No

빈도 테이블

	No	Yes
High	4	3
Normal	1	6
Total	5	9

$P(High|No)$

$\frac{4}{5} = 0.8$

우도(Likelihood) 테이블

	No	Yes	
High	4	3	0.5
Normal	1	6	0.5
Total	5	9	
	0.36	0.64	

$P(High|Yes)$

$\frac{3}{9} = 0.33$

1. Naïve Bayes

❖ 예제를 이용한Naïve Bayes 실습

3. Humidity 데이터를 이용

Humidity	PlayTennis
High	No
High	No
High	Yes
High	Yes
Normal	Yes
Normal	No
Normal	Yes
High	No
Normal	Yes
Normal	Yes
Normal	Yes
High	Yes
Normal	Yes
High	No

빈도 테이블

	No	Yes
High	4	3
Normal	1	6
Total	5	9

우도(Likelihood) 테이블

	No	Yes	
High	4	3	0.5
Normal	1	6	0.5
Total	5	9	
	0.36	0.64	

$P(Normal|No)$

$\frac{1}{5} = 0.2$

$P(Normal|Yes)$

$\frac{6}{9} = 0.67$

1. Naïve Bayes

❖ 예제를 이용한Naïve Bayes 실습

3. Humidity 데이터를 이용

- $P(No|High) = \frac{P(No)P(High|No)}{P(High)}$
  - $P(High|No) = \frac{4}{5} = 0.8$
  - $P(No) = \frac{5}{14} = 0.36$
  - $P(High) = \frac{7}{14} = 0.5$
  - $P(No|High) = \frac{0.36*0.8}{0.5} = 0.58$
- $P(No|Normal) = \frac{P(No)P(Normal|No)}{P(Normal)}$
  - $P(Normal|No) = \frac{1}{5} = 0.2$
  - $P(No) = \frac{5}{14} = 0.36$
  - $P(Normal) = \frac{7}{14} = 0.5$
  - $P(No|Normal) = \frac{0.36*0.2}{0.5} = 0.14$
- $P(Yes|High) = \frac{P(Yes)P(High|Yes)}{P(High)}$
  - $P(High|Yes) = \frac{3}{9} = 0.33$
  - $P(Yes) = \frac{9}{14} = 0.64$
  - $P(High) = \frac{7}{14} = 0.5$
  - $P(Yes|High) = \frac{0.64*0.33}{0.5} = 0.42$
- $P(Yes|Normal) = \frac{P(Yes)P(Normal|Yes)}{P(Normal)}$
  - $P(Normal|Yes) = \frac{6}{9} = 0.67$
  - $P(Yes) = \frac{9}{14} = 0.64$
  - $P(Normal) = \frac{7}{14} = 0.5$
  - $P(Yes|Normal) = \frac{0.64*0.67}{0.29} = 0.86$

1. Naïve Bayes

❖ 예제를 이용한Naïve Bayes 실습

X = (Outlook = Rain, Temperature = Cool, Humidity = Normal)

$$P(X|No) = P(Outlook = Rain|Class = No) * P(Temperature = Cool|Class = No) * P(Humidity = Normal|Class = No)$$
$$= 0.4 * 0.2 * 0.2 = 0.016$$

$$P(X|Yes) = P(Outlook = Rain|Class = Yes) * P(Temperature = Cool|Class = Yes) * P(Humidity = Normal|Class = Yes)$$
$$= 0.33 * 0.33 * 0.67 = 0.073$$

$$P(No|X) = \frac{P(No)P(X|No)}{P(X)} = \frac{0.36 * 0.016}{0.36 * 0.29 * 0.5} = \frac{0.00576}{0.052} = 0.1107 \ldots = 0.1107$$

$$P(Yes|X) = \frac{P(Yes)P(X|Yes)}{P(X)} = \frac{0.64 * 0.073}{0.36 * 0.29 * 0.5} = \frac{0.04672}{0.052} = 0.8984 \ldots = 0.8984$$

$P(No|X) < P(Yes|X)$ => Class = Yes

1. Naïve Bayes

❖ 예제를 이용한Naïve Bayes 실습

- from sklearn.naive_bayes import GaussianNB
- from sklearn.metrics import accuracy_score
- clf = GaussianNB()
- clf.fit(X_train, y_train)
- pred = clf.predict(X_test)
- accuracy = accuracy_score(y_pred, y_test)
- print("score using accuracy: ", accuracy)

1. Naïve Bayes

❖ 예제를 이용한Naïve Bayes 실습

```
from sklearn.naive_bayes import GaussianNB
from sklearn.metrics import accuracy_score

clf = GaussianNB()

clf.fit(features_train, labels_train)

pred = clf.predict(features_test)
accuracy = accuracy_score(pred, test)

print( "score using accuracy: ", accuracy)

```

1. Naïve Bayes

❖ 예제를 이용한Naïve Bayes 실습

```
import numpy as np
import pandas as pd
from sklearn.naive_bayes import GaussianNB
from sklearn.preprocessing import LabelEncoder
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score
```

1. Naïve Bayes

❖ 예제를 이용한Naïve Bayes 실습

```
1 play_tennis = pd.read_csv("D:/big_data/PlayTennis.csv")
2
```

```
1 play_tennis.head()
2
```

	Outlook	Temperature	Humidity	Wind	Play Tennis
0	Sunny	Hot	High	Weak	No
1	Sunny	Hot	High	Strong	No
2	Overcast	Hot	High	Weak	Yes
3	Rain	Mild	High	Weak	Yes
4	Rain	Cool	Normal	Weak	Yes

1. Naïve Bayes

❖ 예제를 이용한Naïve Bayes 실습

```
number = LabelEncoder()
play_tennis['Outlook'] = number.fit_transform(play_tennis['Outlook'])
play_tennis['Temperature'] = number.fit_transform(play_tennis['Temperature'])
play_tennis['Humidity'] = number.fit_transform(play_tennis['Humidity'])
play_tennis['Wind'] = number.fit_transform(play_tennis['Wind'])
play_tennis['Play Tennis'] = number.fit_transform(play_tennis['Play Tennis'])

print(play_tennis)
```

	Outlook	Temperature	Humidity	Wind	Play Tennis
0	2	1	0	1	0
1	2	1	0	0	0
2	0	1	0	1	1
3	1	2	0	1	1
4	1	0	1	1	1
5	1	0	1	0	0
6	0	0	1	0	1
7	2	2	0	1	0
8	2	0	1	1	1
9	1	2	1	1	1
10	2	2	1	0	1
11	0	2	0	0	1
12	0	1	1	1	1
13	1	2	0	0	0

A LabelEncoder converts a categorical data into a number ranging from 0 to n-1

1. Naïve Bayes

❖ 예제를 이용한Naïve Bayes 실습

```
features = ["Outlook", "Temperature", "Humidity", "Wind"]
target = "Play Tennis"

X=play_tennis[features]
y=play_tennis[target]


print(X, y)
```

```
    Outlook  Temperature  Humidity  Wind
0         2            1         0     1
1         2            1         0     0
2         0            1         0     1
3         1            2         0     1
4         1            0         1     1
5         1            0         1     0
6         0            0         1     0
7         2            2         0     1
8         2            0         1     1
9         1            2         1     1
10        2            2         1     0
11        0            2         0     0
12        0            1         1     1
13        1            2         0     0 0     0
```

```
1     0
2     1
3     1
4     1
5     0
6     1
7     0
8     1
9     1
10    1
11    1
12    1
13    0
Name: Play Tennis, dtype: int64
```

1. Naïve Bayes

❖ 예제를 이용한Naïve Bayes 실습

```
1  X_train, x_test, Y_train, y_yest = train_test_split(X,y,test_size = 0.3,random_state = 42)
```

```
1  model = GaussianNB()
2  model.fit(X_train, Y_train)
```

```
GaussianNB(priors=None, var_smoothing=1e-09)
```

```
1  pred = model.predict(x_test)
2  accuracy = accuracy_score(y_yest, pred)
3
4  print( "acc : ", accuracy)
```

```
acc :  0.6
```

1. Naïve Bayes

❖ 예제를 이용한Naïve Bayes 실습

- Outlook = Overcast /
- Temperature = Hot/
- Humidity = Normal/
- Wind = strong

```
1  print(model.predict([[0,1,1,0]]))
2
```

```
[1]
```

1. Naïve Bayes

❖ 예제를 이용한Naïve Bayes 실습

```
from sklearn import datasets
from sklearn.metrics import confusion_matrix, classification_report
from sklearn.model_selection import train_test_split
from sklearn.naive_bayes import GaussianNB
from sklearn.preprocessing import StandardScaler

iris = datasets.load_iris()
print(type(iris))

print('data shape:', iris.data.shape)
print('iris target:', iris.target_names)
print('iris features:', iris.feature_names)

X = iris.data # 데이터(특성, 변수)
print('type X:', type(X))
print(X[:5])
y = iris.target # 분류 클래스(레이블)
print('type y:', type(y))
print(y[:5])


```

```
<class 'sklearn.utils.Bunch'>
data shape: (150, 4)
iris target: ['setosa' 'versicolor' 'virginica']
iris features: ['sepal length (cm)', 'sepal width (cm)', 'petal length (cm)', 'petal width (cm)']
type X: <class 'numpy.ndarray'>
[[5.1 3.5 1.4 0.2]
 [4.9 3.  1.4 0.2]
 [4.7 3.2 1.3 0.2]
 [4.6 3.1 1.5 0.2]
 [5.  3.6 1.4 0.2]]
type y: <class 'numpy.ndarray'>
[0 0 0 0 0]
```

1. Naïve Bayes

❖ 예제를 이용한Naïve Bayes 실습

```
# 데이터 세트를 학습(train)/검증(test) 세트로 나눔.
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2)

scaler = StandardScaler()
scaler.fit(X_train, y_train)
X_train_scale = scaler.transform(X_train)
X_test_scale = scaler.transform(X_test)

print(X_train_scale)
print(X_test_scale)
```

```
[[ 0.40175499 -1.91127029  0.37672815  0.34662953]
 [-0.19197159 -0.07092979  0.20195736 -0.05590799]
 [-0.78569817  0.84924046 -1.42923669 -1.39769974]
 [-0.54820754  1.53936815 -1.37097976 -1.39769974]
 [-0.90444349  1.76941071 -1.13795204 -1.12934139]
 [-0.42946223 -1.45118517 -0.08932729 -0.32426634]
 [-0.07322628 -0.76105748  0.72626973  0.88334623]
 [ 0.75799093  0.38915534  0.72626973  1.01752541]
 [-0.31071691 -1.2211426   0.02718657 -0.19008716]
 [-0.54820754 -0.07092979  0.37672815  0.34662953]
 [ 0.63924562 -0.53101492  1.01755438  1.15170458]
 [-0.07322628 -0.76105748  0.14370043 -0.32426634]
 [-0.07322628 -0.99110004  0.0854435  -0.05590799]
 [-0.54820754  1.99945327 -1.48749362 -1.12934139]
 [-1.26067944  0.15911277 -1.31272283 -1.39769974]
 [ 0.99548156 -1.2211426   1.13406824  0.74916706]
 [-0.42946223  2.68958096 -1.42923669 -1.39769974]
 [ 0.04551904  0.38915534  0.55149894  0.74916706]
 [-0.54820754  0.84924046 -1.37097976 -1.12934139]
```

1. Naïve Bayes

❖ 예제를 이용한Naïve Bayes 실습

```python
# 머신 러닝 모델 선택 - Naive Bayes
gnb = GaussianNB()
gnb.fit(X_train_scale, y_train) # 모델 학습
y_pred = gnb.predict(X_test_scale) # 예측

# 성능 측정
print(confusion_matrix(y_test, y_pred))
print(classification_report(y_test, y_pred))
```

```
[[14  0  0]
 [ 0  7  0]
 [ 0  1  8]]
              precision    recall  f1-score   support

           0       1.00      1.00      1.00        14
           1       0.88      1.00      0.93         7
           2       1.00      0.89      0.94         9

    accuracy                           0.97        30
   macro avg       0.96      0.96      0.96        30
weighted avg       0.97      0.97      0.97        30
```

1. Naïve Bayes

❖ 예제를 이용한Naïve Bayes 실습

```
1  from sklearn import datasets
2
3  #Load dataset
4  wine = datasets.load_wine()
```

```
1  print("Features: ", wine.feature_names)
2  print("Labels: ", wine.target_names)
3
```

```
Features:  ['alcohol', 'malic_acid', 'ash', 'alcalinity_of_ash', 'magnesium', 'total_phenols', 'flavanoids', 'nonflavanoid_phenols', 'proan
thocyanins', 'color_intensity', 'hue', 'od280/od315_of_diluted_wines', 'proline']
Labels:  ['class_0' 'class_1' 'class_2']
```

1. Naïve Bayes

❖ 예제를 이용한Naïve Bayes 실습

```
X= wine.data
y= wine.target
```

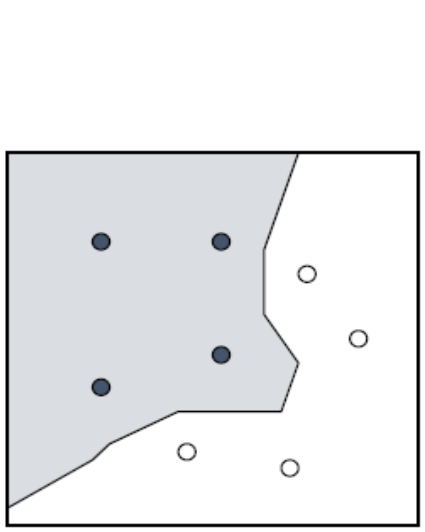
```
from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=1)

from sklearn.naive_bayes import GaussianNB

gnb = GaussianNB()

gnb.fit(X_train, y_train)

y_pred = gnb.predict(X_test)

from sklearn import metrics

print("Accuracy:", metrics.accuracy_score(y_test, y_pred))
```

```
Accuracy: 0.9814814814814815
```

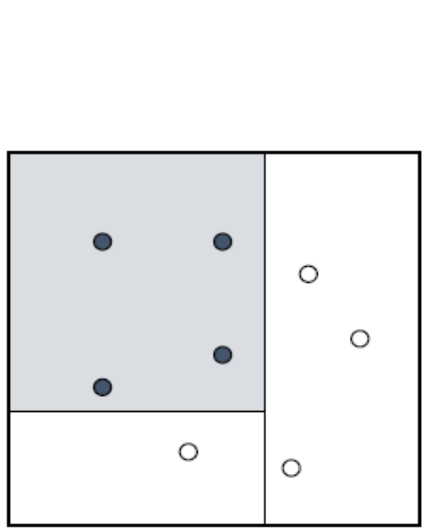
2. Support Vector Machine

❖ SVM 개념

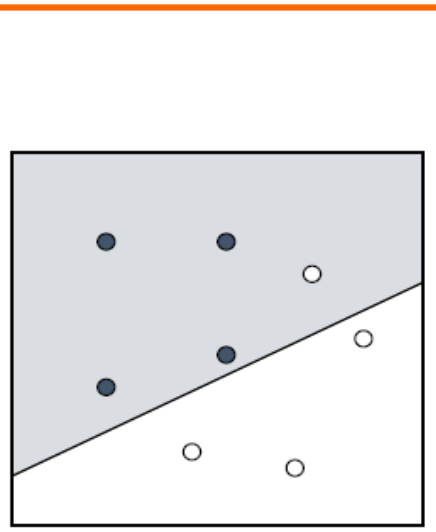
- "How do we divide the space with decision boundaries?"



Nearest Neighbor

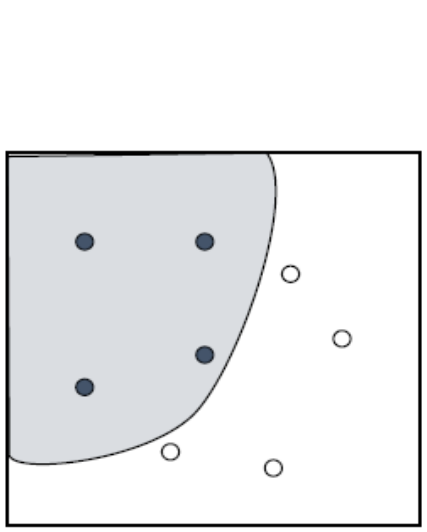


Decision Tree



Linear Functions

$$g(\mathbf{x}) = \mathbf{w}^T\mathbf{x} + b$$

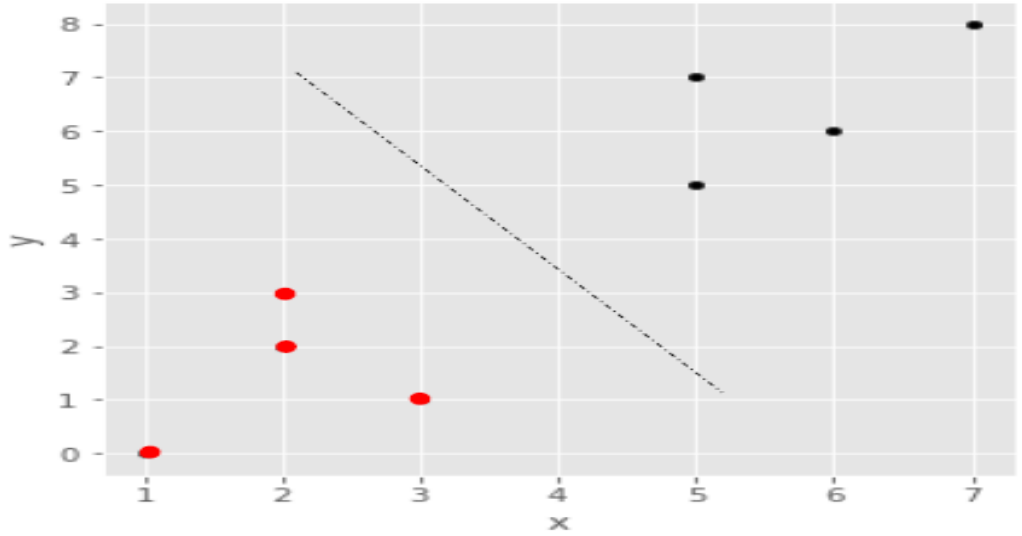


Nonlinear Functions

2. Support Vector Machine

❖ SVM 개념

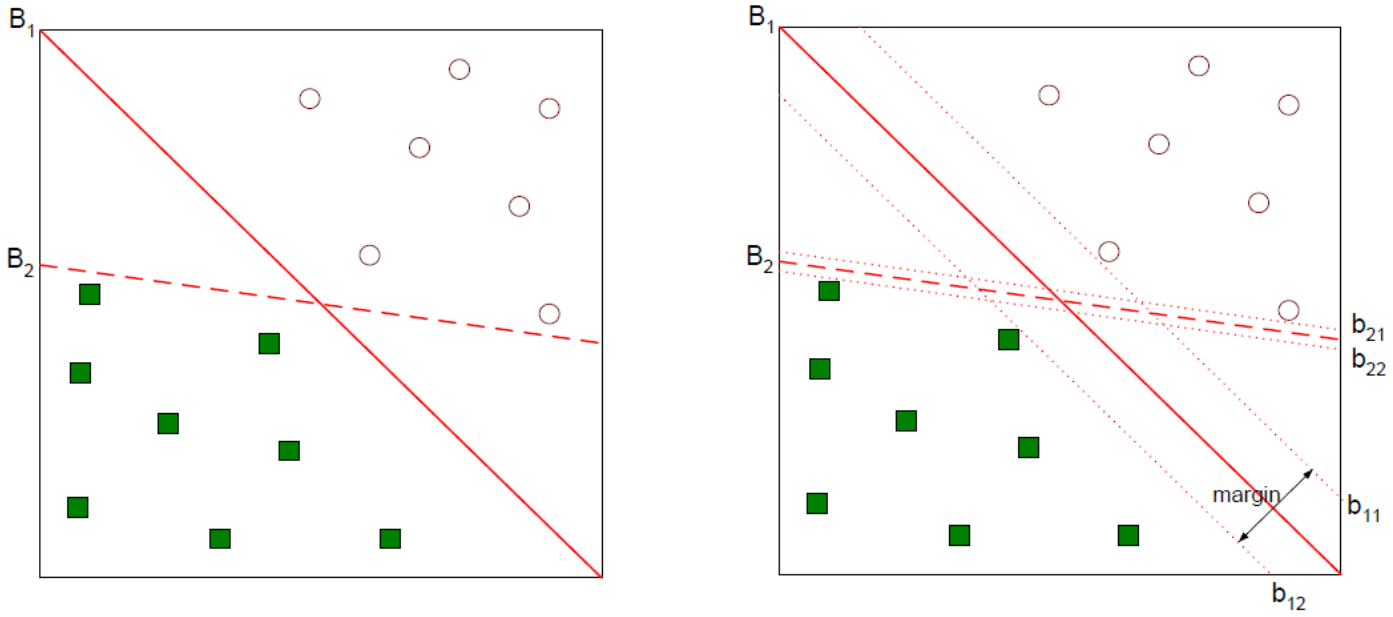
- "Support Vector Machine"은 분류 또는 회귀문제에 사용할 수 있는 기계학습 알고리즘
- 딥 러닝 못지 않은 성능을 내고, 무엇보다도 가벼움
- 분류(classification)나 회귀(regression)의 목적으로 활용
- SVM 알고리즘에서는 각 데이터항목을 n 차원 공간상 하나의 점으로 표시
- 이질적인 두개 또는 그 이상의 데이터 집단을 잘 구분하는 최적의 초평면(Optimal Hyper Plane)을 찾는 방법을 제공



2. Support Vector Machine

❖ SVM 개념

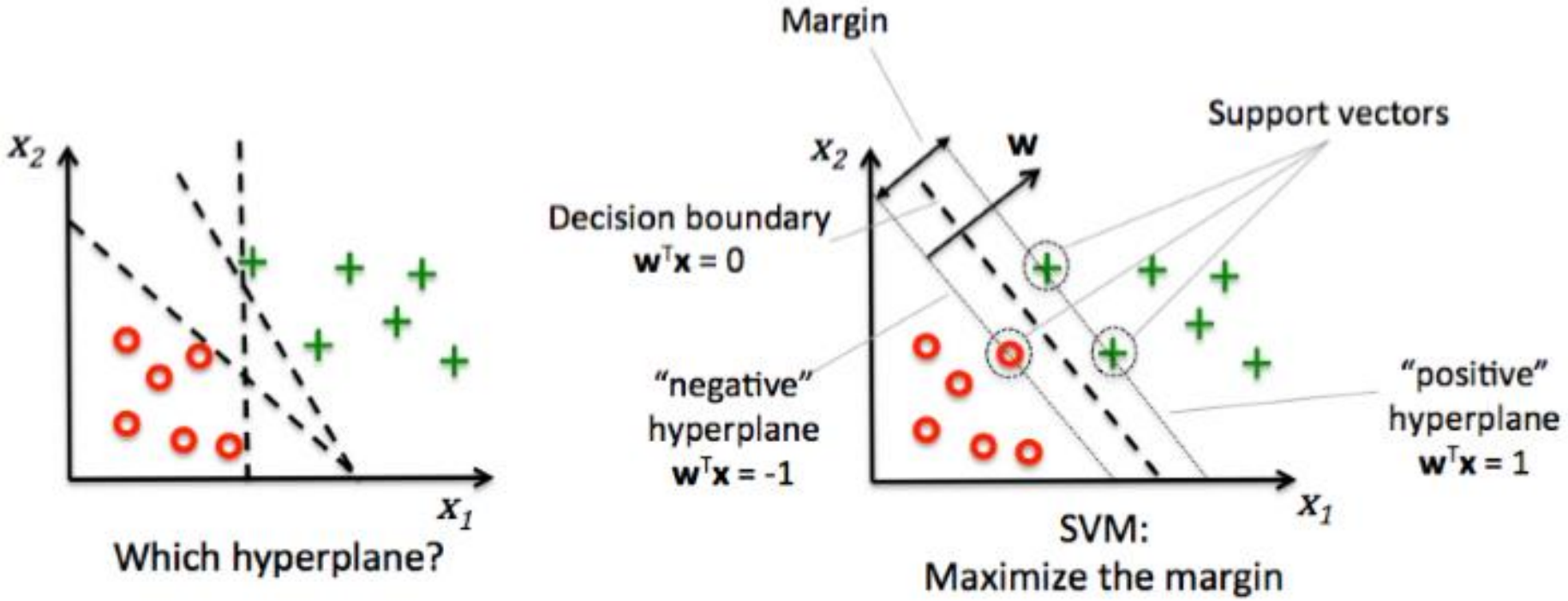
- SVM은 Classification을 한 다음에 Margin이 가장 큰 선
- 서포트 벡터들은 두 클래스 사이의 경계에 위치한 데이터 포인트
- 서포트 벡터들이 결정 경계를 만드는데 영향



2. Support Vector Machine

❖ SVM 개념

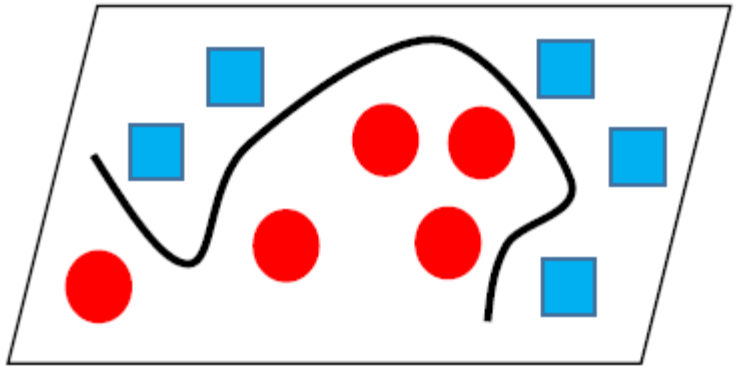
- 데이터들이 결정 경계를 지지(support)하고 있다고 말할 수 있기 때문에, 서포트벡터
- Margin 이란 초평면 가까이에 있는 Support Vector에서 초평면까지의 거리의 합을 의미함



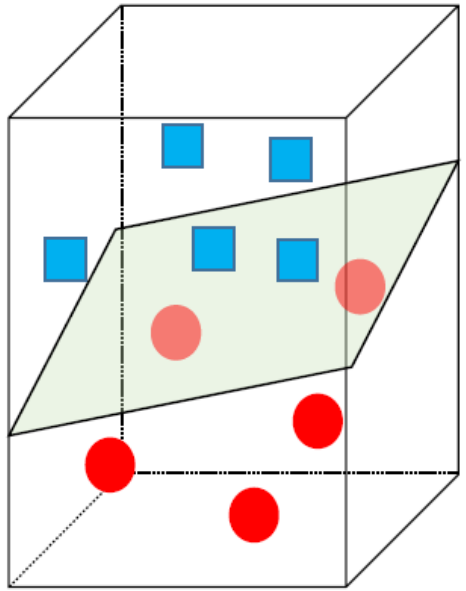
2. Support Vector Machine

❖ SVM 분류

- 각각의 커널에서는 최적화를 도와주는 파라미터들이 따로 존재함
- 일반적으로 각 문제에 대해서 어떠한 커널의 파라미터를 선택하는 것이 가장 좋은지를 자동적으로 알려주는 방법은 없음
- 실험을 통해 모든 조건을 바꾸면서 SVM의 학습과 예측을 반복해서 최적의 예측률을 보여주는 조건을 찾아야함



입력 공간

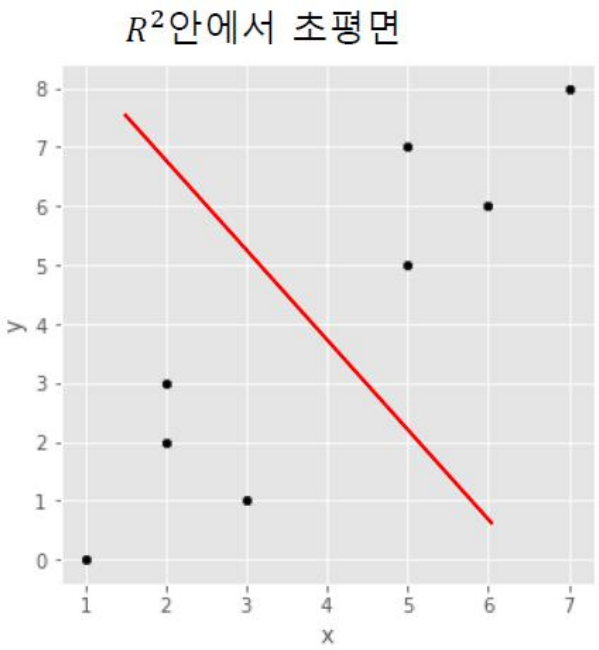


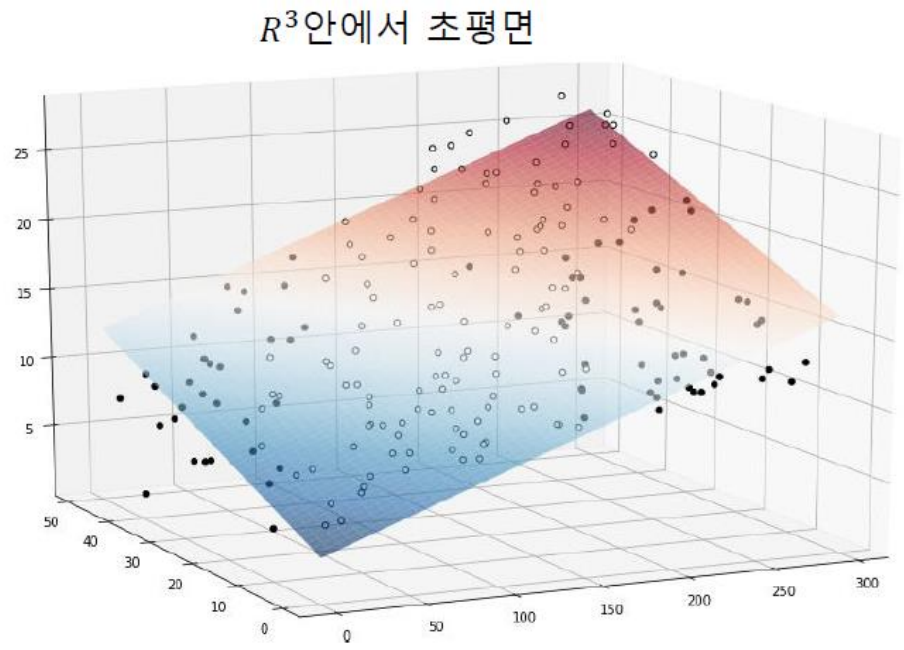
특징 공간

2. Support Vector Machine

❖ SVM 초평면의 개념

- 2차원 공간 ($R2$)에서의 초평면은 아래 왼쪽그림과 같이 나타낼 수 있고
- 3차원 공간($R3$)에서의 초평면은 아래 오른쪽 그림과 같이 나타낼 수 있음





2. Support Vector Machine

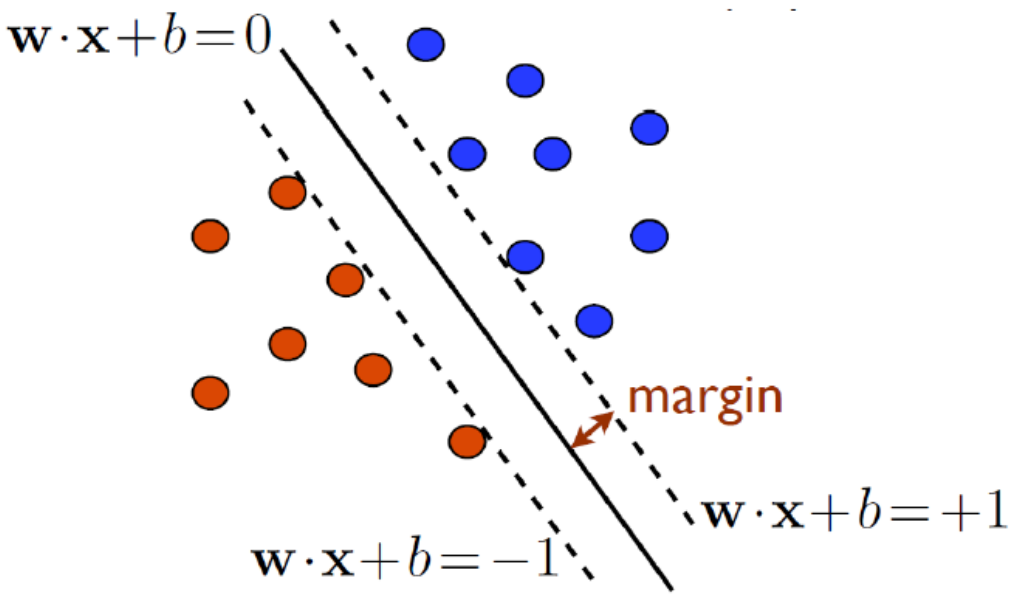
❖ SVM 분류 – Hard Margin

- **Objective function**

$$\min \ \frac{1}{2}\|\mathbf{w}\|^2 \quad = \quad \max \frac{1}{||\vec{w}||} \leftrightarrow \min ||\vec{w}|| \leftrightarrow \min \frac{1}{2}||\vec{w}||^2$$

- **Constraints**

$$s.t. \ \ y_i(\mathbf{w}^{\mathrm{T}}\mathbf{x}_i + b) \geq 1$$



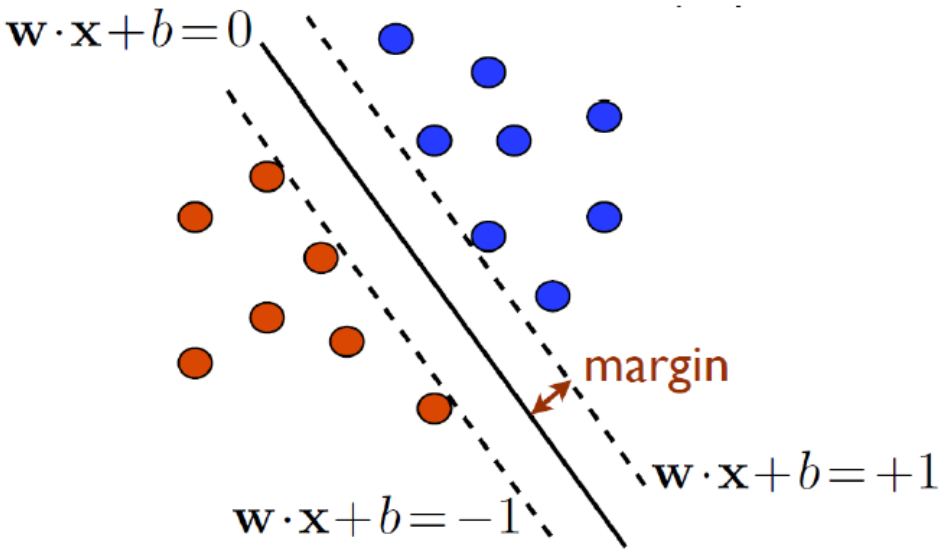
2. Support Vector Machine

❖ SVM 분류

- 마진은 평행한 초평면 사이의 거리를 뜻함
- 평행한 두 초평면

$$\vec{w} \cdot \vec{x} + b = -1 \, , \; \vec{w} \cdot \vec{x} + b = +1$$

- 평행한 두 초평면에 마진을 구하는 식은 다음과 같음
  - $\vec{w} \cdot \vec{x} + (b+1) = 0 \, , \; \vec{w} \cdot \vec{x} + (b-1) = 0$
  - 두 초평면 사이의 거리식은 $D = |b_1 - b_2| / \|\vec{w}\|$
- 결과적으로 $D = 2/\|\vec{w}\|$ 이 성립됨
- 우리는 마진을 극대화하기를 원하기 때문에, $\|\vec{w}\|$ 최소화하거나
- $\frac{1}{2}\|\vec{w}\|^2$ 를 최소화 해야함

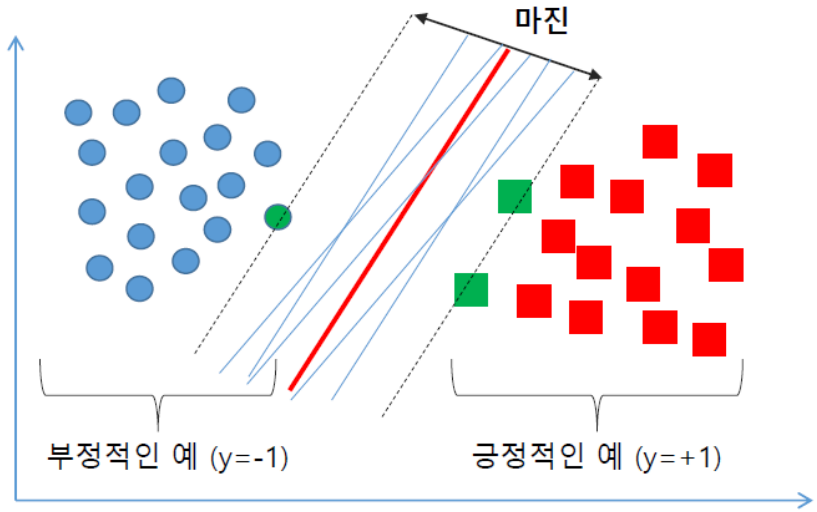


2. Support Vector Machine

❖ SVM 분류

$$\vec{x}_1, \vec{x}_2, \ldots, \vec{x}_N \in R^n$$
$$\vec{y}_1, \vec{y}_2, \ldots, \vec{y}_N \in \{-1, +1\}$$

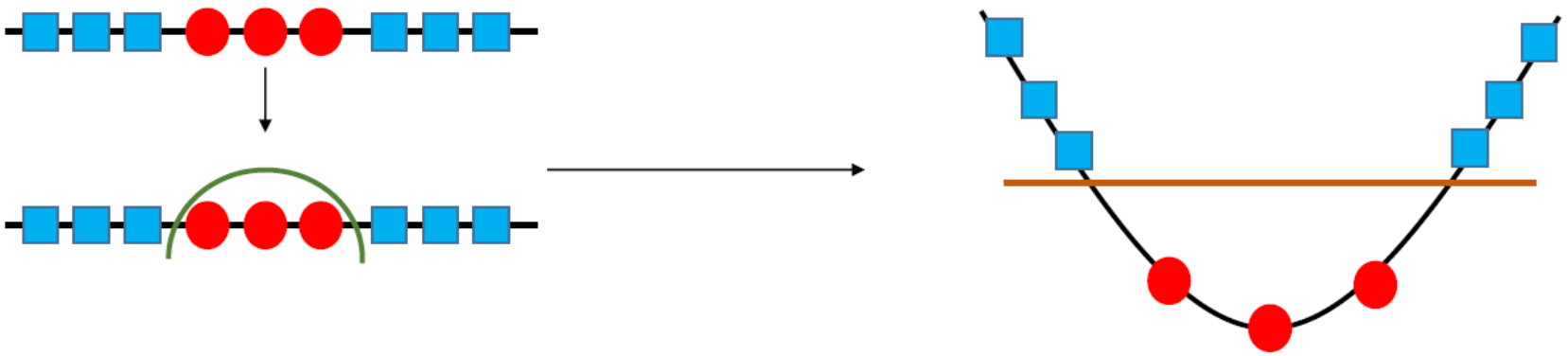
- 부정적인 예와 긍정적인 예를 분류할 수 있는 초평면을 찾을 때 다음과 같이 초평면이 무한히 존재함
- SVM은 경계에 있는 서포트벡터사이의 간격을 최대화하는 초평면을 찾음
- 경계상의 서포트벡터들이 잡음으로 인해 정확하지 않은 경우, SVM은 잘 분리하지 못함



2. Support Vector Machine

❖ SVM 분류 – 비선형 분류

- SVM은 선형분류와 더불어 비선형분류에서도 사용
- 비선형 분류를 하기 위해서는 주어진 데이터를 고차원 특징공간으로 사상하는 작업이 필요
- 효율적으로 실행하기위해서 커널트릭을 사용
- 이러한 Kernel을 이용하여 차원을 변경하게 되면,
  흩어져 있는 데이터에 대해서도 차원을 변경하여 간단하게 나눌 수 있는 장점을 가짐
- 주요 Kernel은 Linear Kernel, Polynomial Kernel, RBF(Radial Basis Function)이 있음



2. Support Vector Machine

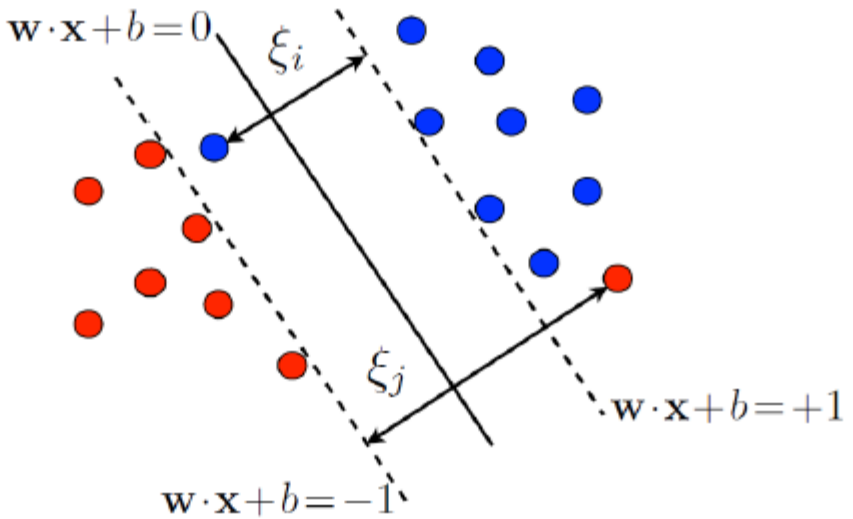
❖ SVM 분류 – Soft Margin

- Objective function

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

- Constraints

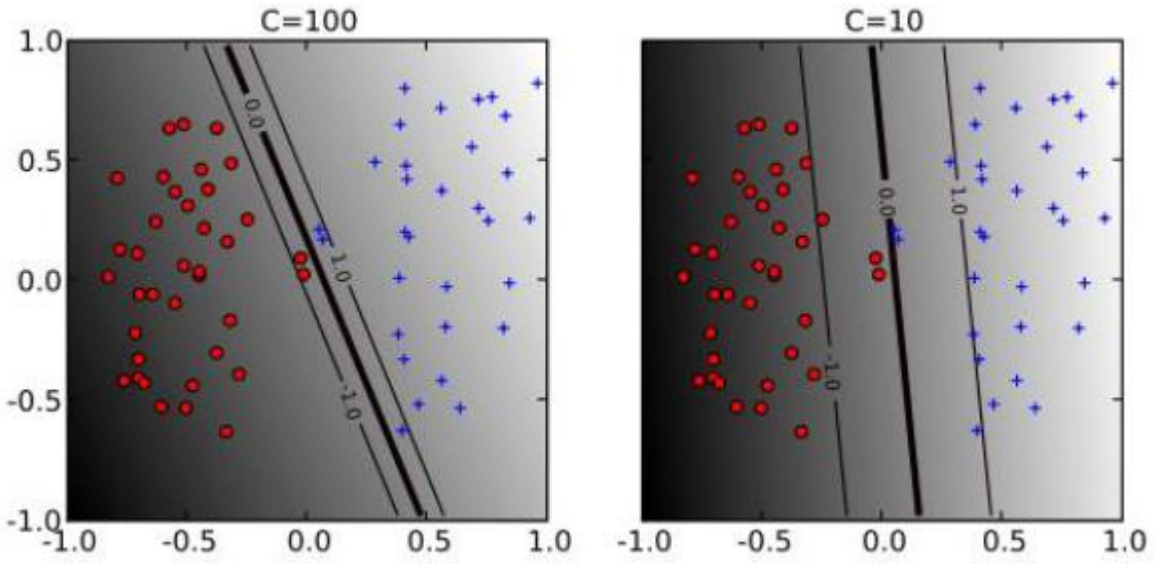
$$s.t. \ \ y_i\left(\mathbf{w}^{\mathrm{T}}\mathbf{x}_i + b\right) \geq 1 - \xi_i, \ \xi_i \geq 0, \ \ \forall i$$

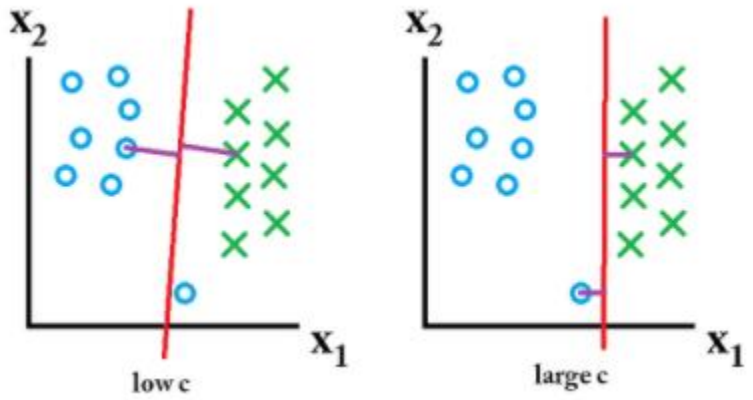


2. Support Vector Machine

❖ SVM 분류

- **C는 얼마나 많은 데이터 샘플이 다른 클래스에 놓이는 것을 허용하는지를 결정**
- 작을 수록 많이 허용하고, 클 수록 적게 허용
- C값을 낮게 설정하면 이상치들이 있을 가능성을 크게 잡아 일반적인 결정 경계를 찾아내고,
- 높게 설정하면 반대로 이상치의 존재 가능성을 작게 봐서 좀 더 세심하게 결정 경계

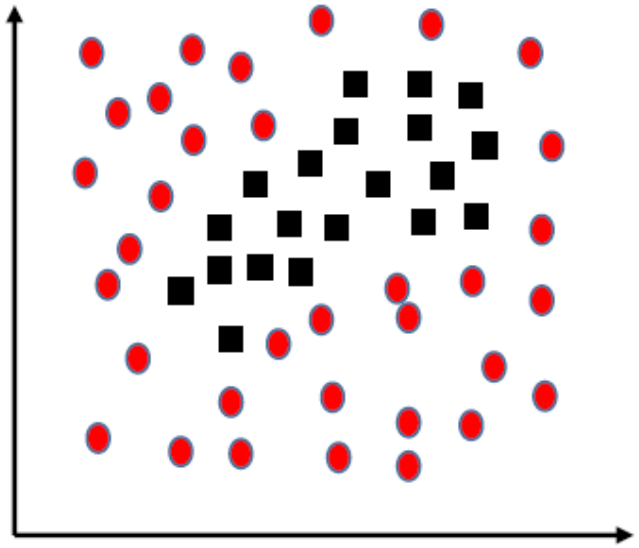




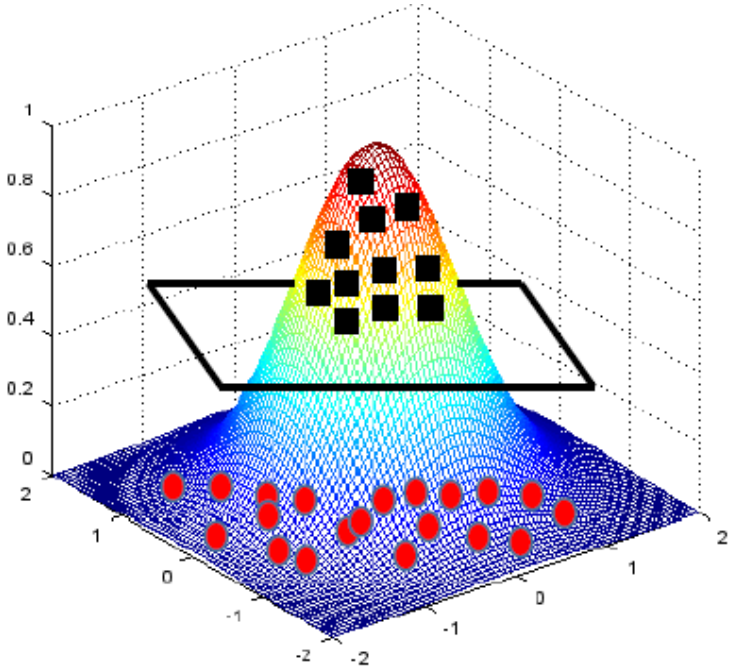
2. Support Vector Machine

❖ SVM 커널트릭

- **선형으로 분리 할 수 있는 데이터가 아닐때 사용**
- **커널 기법은 주어진 데이터를 고차원 특징 공간으로 사상**



데이터가 입력 공간에서
선형으로 분리되지 않음



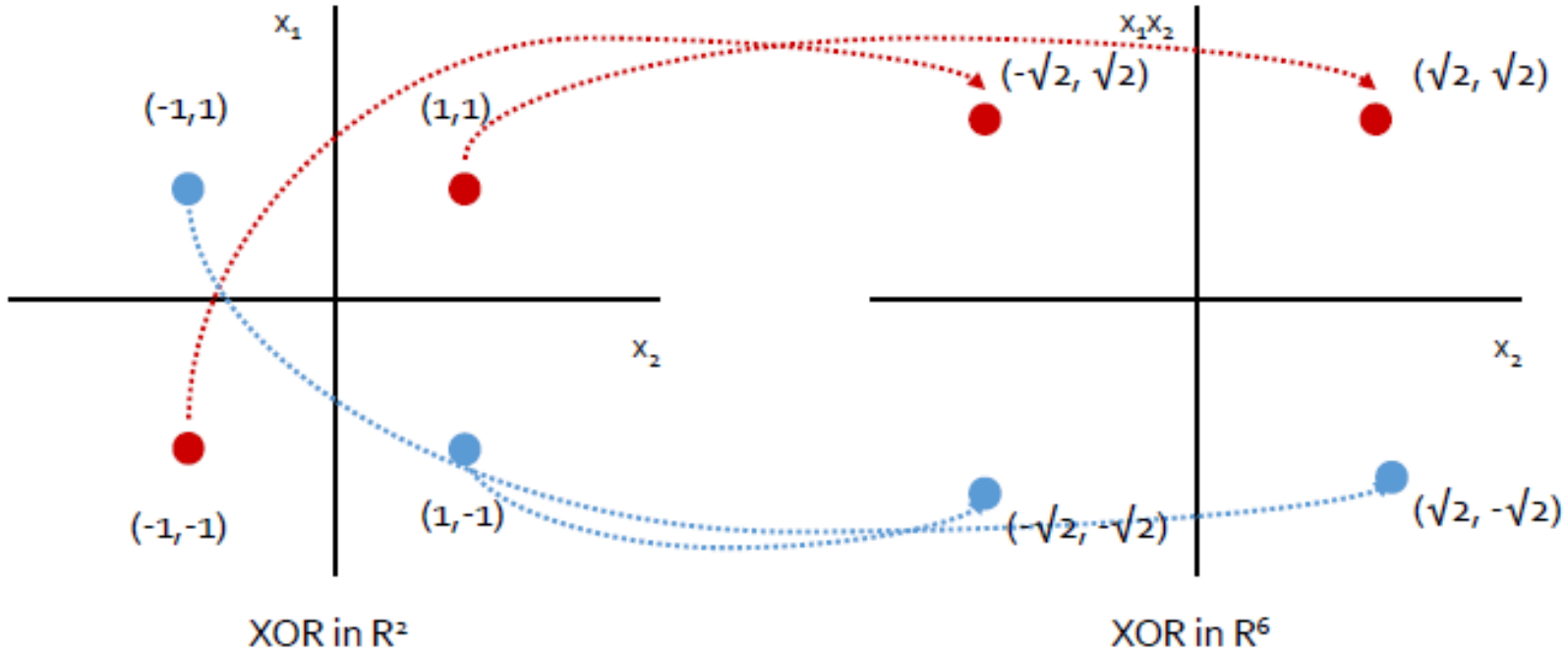
커널에 의해 얻어진 특정 공간
에서 선형으로 분리 가능함

2. Support Vector Machine

❖ SVM 커널트릭

- 선형으로 분리 할 수 있는 데이터가 아닐때 사용
- 커널 기법은 주어진 데이터를 고차원 특징 공간으로 사상

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

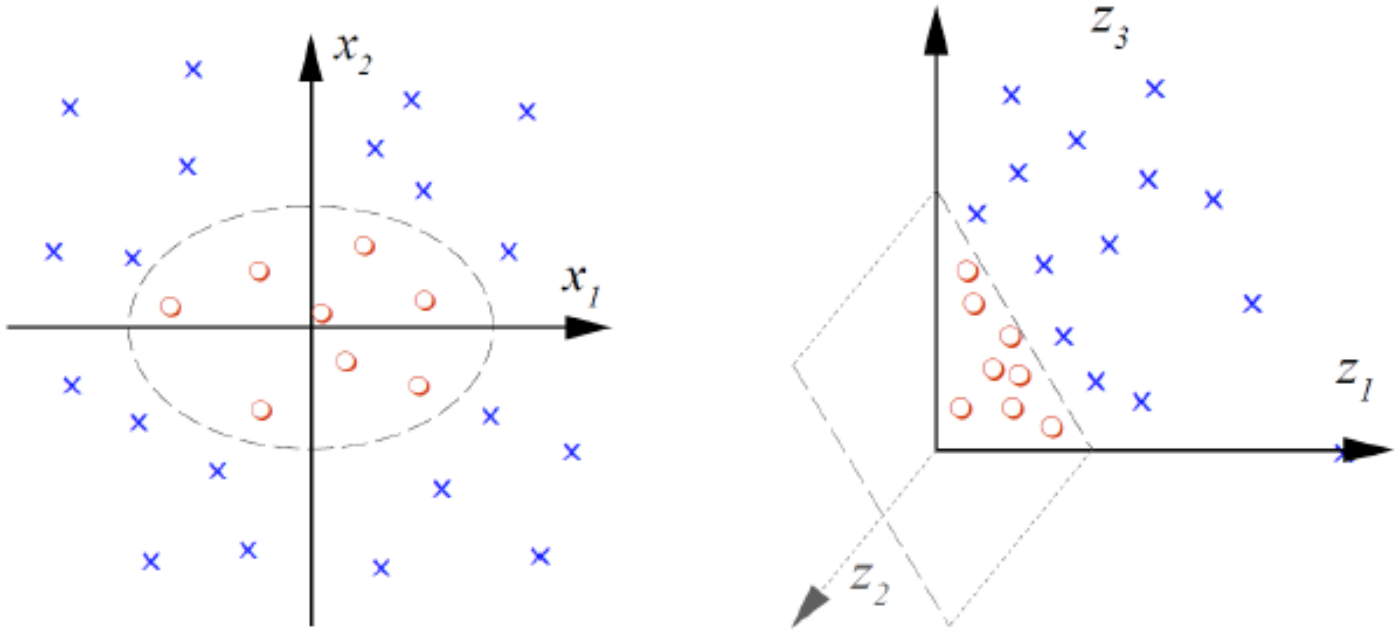


2. Support Vector Machine

❖ SVM 커널트릭

- 선형으로 분리 할 수 있는 데이터가 아닐때 사용
- 커널 기법은 주어진 데이터를 고차원 특징 공간으로 사상

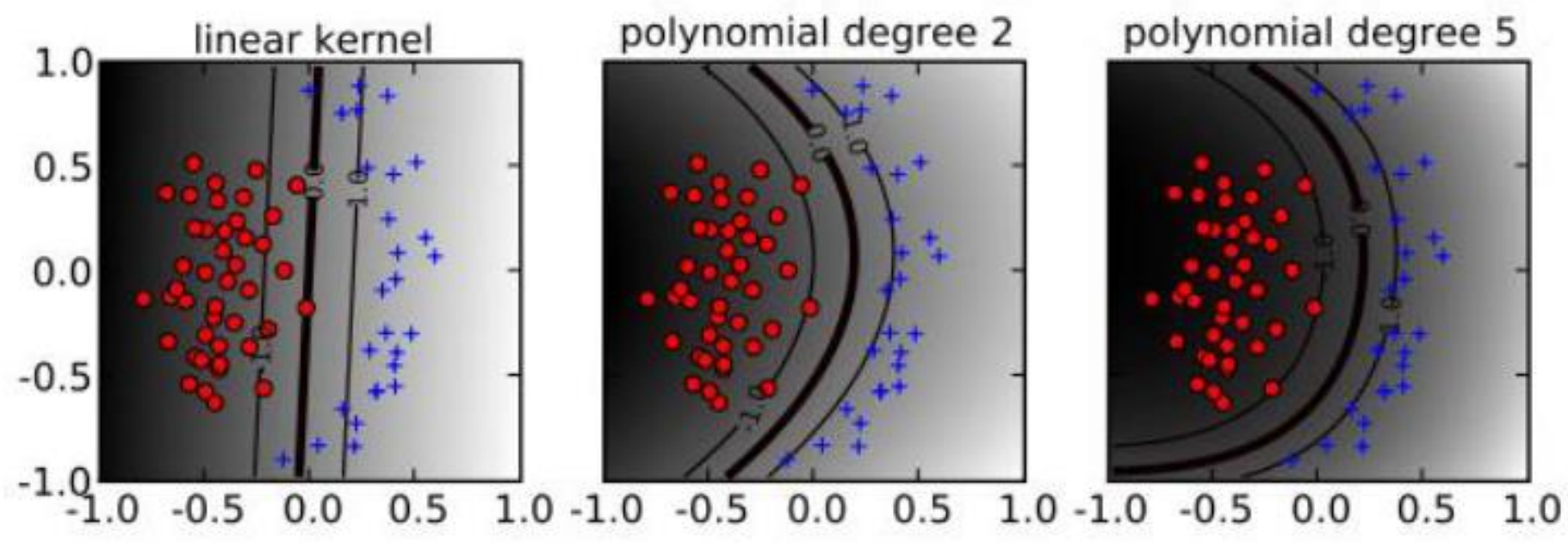
$$\Phi : \mathbb{R}^2 \to \mathbb{R}^3$$
$$(x_1, x_2) \mapsto (z_1, z_2, z_3) := (x_1^2, \sqrt{2}\, x_1 x_2, x_2^2)$$



2. Support Vector Machine

❖ SVM 커널트릭

- 선형으로 분리 할 수 있는 데이터가 아닐때 사용
- 커널 기법은 주어진 데이터를 고차원 특징 공간으로 사상



2. Support Vector Machine

❖ SVM 커널트릭

- **커널은 일부공간에서 점으로 표현**

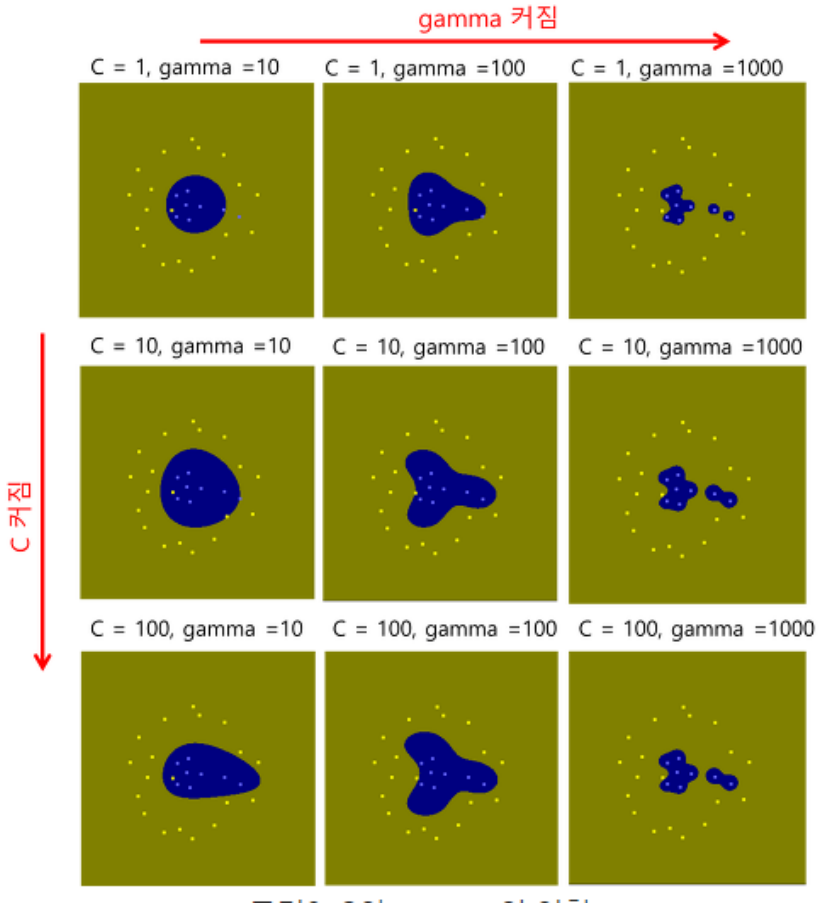
$$K(\vec{x}_i, \vec{x}_j) = \Phi(\vec{x}_i) \cdot \Phi(\vec{x}_j)$$

$K(\vec{x}_i, \vec{x}_j) = \vec{x}_i \cdot \vec{x}_j$	Linear kernel
$K(\vec{x}_i, \vec{x}_j) = \exp(-r\|\vec{x}_i - \vec{x}_j\|^2)$	Gaussian kernel
$K(\vec{x}_i, \vec{x}_j) = \exp(-r\|\vec{x}_i - \vec{x}_j\|)$	Exponential kernel
$K(\vec{x}_i, \vec{x}_j) = (p + \vec{x}_i \cdot \vec{x}_j)^q$	Polynomial kernel
$K(\vec{x}_i, \vec{x}_j) = (p + \vec{x}_i \cdot \vec{x}_j)^q \exp(-r\|\vec{x}_i - \vec{x}_j\|^2)$	Hybrid kernel
$K(\vec{x}_i, \vec{x}_j) = \tanh(k\vec{x}_i \cdot \vec{x}_j - \delta)$	Sigmoidal

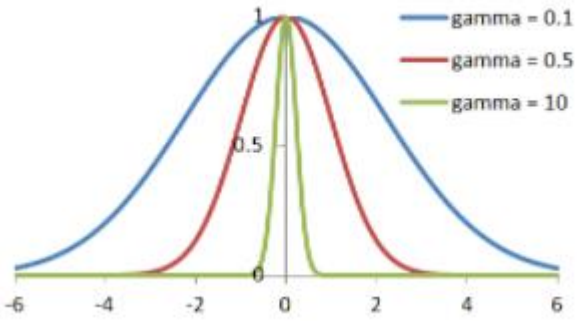
2. Support Vector Machine

❖ SVM 커널트릭

- **gamma는 하나의 데이터 샘플이 영향력을 행사하는 거리를 결정**



C는 데이터 샘플들이 다른 클래스에 놓이는 것을 허용결정
gamma는 결정 경계의 곡률을 결정



2. Sklearn 소개

❖ 서포트벡터머신

- from sklearn import svm

```
clf = svm.SVC(kernel='linear')
clf.fit(x_train, y_train)
clf.predict(x_test)
clf.score(x_test, y_test)
```

2. Sklearn 소개

❖ 서포트벡터머신

- from sklearn import svm import SVC
- model = SVC(kernel='linear').fit(X, y)
- model = SVC(kernel='rbf', C=10.0, gamma=0.10)
- model.fit(X_train, y_traim)
- y_pred = model.predict(X_test)
- print('Accuracy: %.2f' % accuracy_score(y_test, y_pred))

2. Support Vector Machine

❖ 파이썬 SklearnSVM

- **SVM에서 사용되는 Parameter는 Kernel, C, gamma 대표적으로 많이 사용하는 총 세가지 종류가 있음**
- **Kernel**
  - 기본값으로rbf를 설정(값은 특정중심에서 거리에 의존하는 함수값)
  - rbf(Radial Basis Function, linear, poly(Polynomial) 총 3가지가 있음
- **C**
  - 기본값으로1.0 설정
  - C 값을 낮추면 초평면이 매끄러워지고, 값을 높이면 서포트벡터들을 더 잘 분류함(Classifying training points correctly)
- **Gamma**
  - 기본값으로auto 설정
  - Gamma 값을 낮추면 초평면에서 멀리 떨어진 서포트벡터들의 영향이 낮고 값을 높이면 멀리 떨어진 요소들의 값이 영향이 큼
  - 값을 높일 경우 초평면에 인접한 서포트벡터들의 영향(Weight)가 커지기 때문에 초평면이 울퉁불퉁(Uneven)하게 됨

2. Support Vector Machine

❖ 파이썬 SklearnSVM Parameter

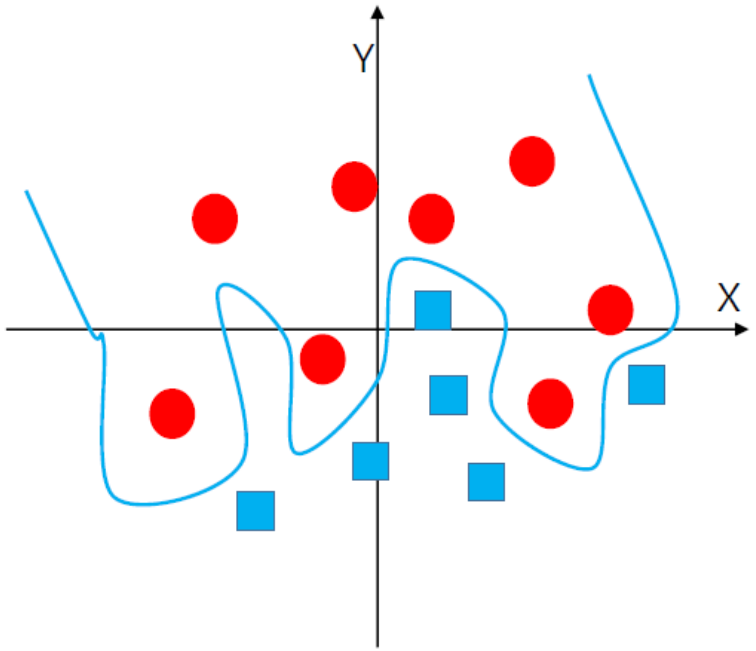
- **SVM 파라미터를 튜닝하기 위해서 GridSearchCV를 활용**

Parameter	type	설명
C	float (default = 1.0)	정규화 매개 변수, 정규화의 매개 변수는 C에 반비례한다.
kernel	string (default='rbf')	SVM 알고리즘에 사용될 커널 유형을 지정한다. 'linear', 'ploy', 'rbf', 'sigmoid', 'percomputed' , 'callable' 등의 커널을 지원한다.
degree	int (default = 3)	다항식 커널 함수의 각도로 'ploy' 커널을 사용했을 때 사용된다. 다른 커널에서는 무시된다.
gamma	{'scale', ' auto'} or float (default = 'scale')	'rbf', 'ploy', 'sigmoid' 커널에서의 커널 계수이디/
coef0	float (default = 0.0)	'ploy', 'sigmoid' 에서의 독립구간
shrinking	boolean (default = True)	휴리스틱 값이 줄어드는지의 여부
probability	boolean (default = False)	확률 추정을 활성화할 지의 여부
tol	flaot (default = 1e-3)	중지 기준에 대한 오차허용 값
cache_size	float	커널 캐시의 크기
class_weight	dict, ' balanced'	클래스 가중치, 클래스 불균형에 가중치를 주는 기능의 조정
verbose	bool (default = False)	출력 자세히 보기의 여부
max_iter	int (default = -1)	반복 횟수 제한 (제한이 없으면 -1)
decision_function_shape	'ovo', 'ovr' (default = 'ovr')	결정 함수의 모양을 결정한다.
break_ties	bool (default = False)	True, 결정 함수의 모양이 'ovr', 클래스 수가 2보다 큰 경우 신뢰도 값에 따라 연결이 끊어진다.
random_state	int (defualt = None)	데이터를 섞을 때 사용되는 의사 난수 생성기의 seed값

2. Support Vector Machine

❖ SVM Overfitting

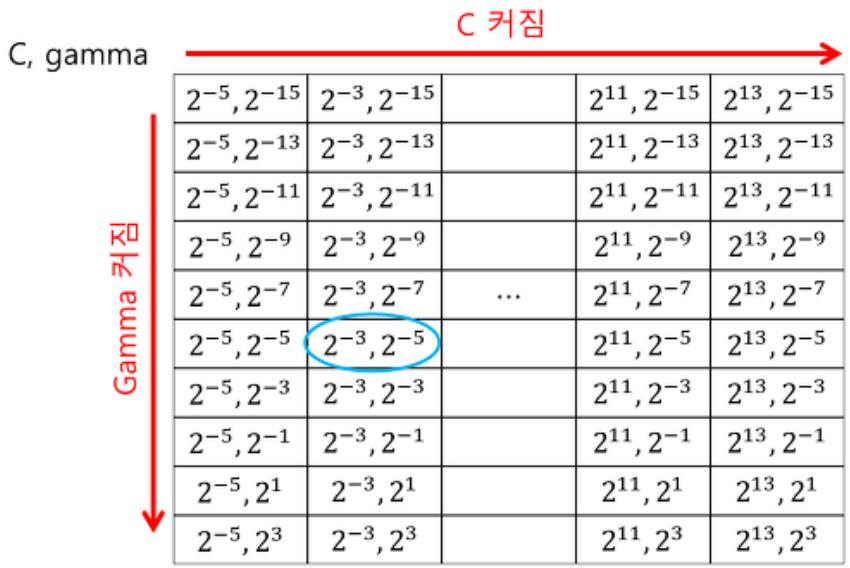
- 초평면을 정한것을 보고 Overfitting 되었다 고함
- Machine Learning에서는 이러한 경우를 피해야함
- SVM Parameters인Kernel, C, Gamma는Overfitting에 영향을 주는 요소 중 일부



2. Support Vector Machine

❖ GridSearchCV

- **train/ test split은 결과가 불안정하고 테스트 데이터의 분할에 크게 의존**
- **일반적인 train과 test에 대한 성능을 더 잘 평가하기 위해 한번만 나누지 않고, 교차검증과 Search를 통해**
  - **최적의 파라미터를 찾음**
  - **교차검증도**



GridSearchCV를 사용하면 최적의 parameter와 그때의 score값을 한 번에 찾음.

2. Support Vector Machine

❖ SVM 실습

```
1  from sklearn.svm import SVC
2  classifier = SVC(kernel = 'linear')
3  training_points = [[1, 2], [1, 5], [2, 2], [7, 5], [9, 4], [8, 2]]
4  labels = [1, 1, 1, 0, 0, 0]
5  classifier.fit(training_points, labels)
```

```
SVC(C=1.0, break_ties=False, cache_size=200, class_weight=None, coef0=0.0,
    decision_function_shape='ovr', degree=3, gamma='scale', kernel='linear',
    max_iter=-1, probability=False, random_state=None, shrinking=True,
    tol=0.001, verbose=False)
```

```
1  print(classifier.predict([[3, 2]]))
2
```

```
[1]
```

2. Support Vector Machine

❖ SVM 실습

```
from sklearn.datasets import load_iris
from sklearn.svm import SVC
from sklearn.model_selection import train_test_split, cross_val_score

iris = load_iris()

col1 = 0
col2 = 1

X = iris.data[:,:2] # 시각화를 위해 속성 2개만 선정
y = iris.target

# print(X, y)

# 학습용/테스트용 데이터 분리
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state= 42)
X_train.shape, X_test.shape, y_train.shape, y_test.shape

# 모델 정의
model = SVC()

# 학습시키기
model.fit(X_train, y_train)

# 평가하기
score1 = model.score(X_train, y_train)
print(score1)
```

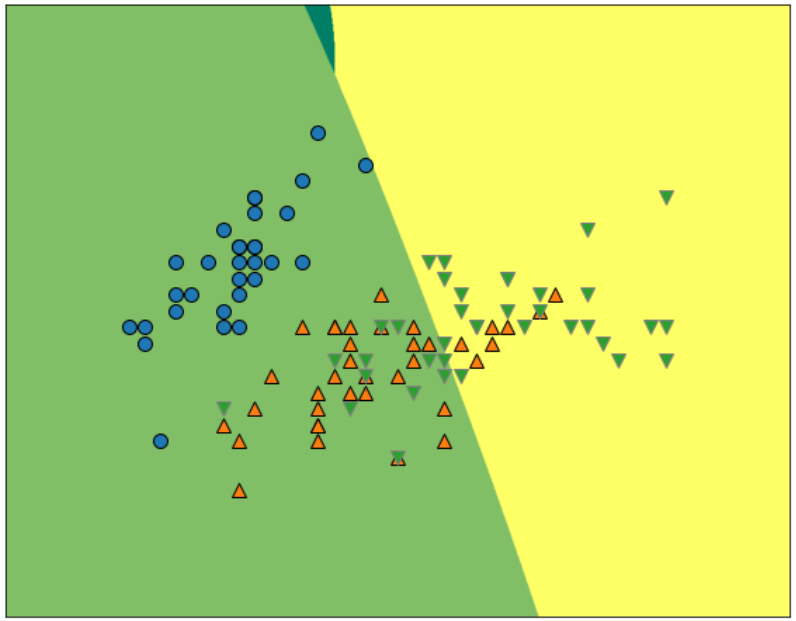
```
0.8095238095238095
```

2. Support Vector Machine

❖ SVM 실습

```
# !pip install mglearn
import mglearn

plt.figure(figsize=[10,8])
mglearn.plots.plot_2d_classification(model, X_train, cm='summer')
mglearn.discrete_scatter(X_train[:,0], X_train[:,1], y_train)
```

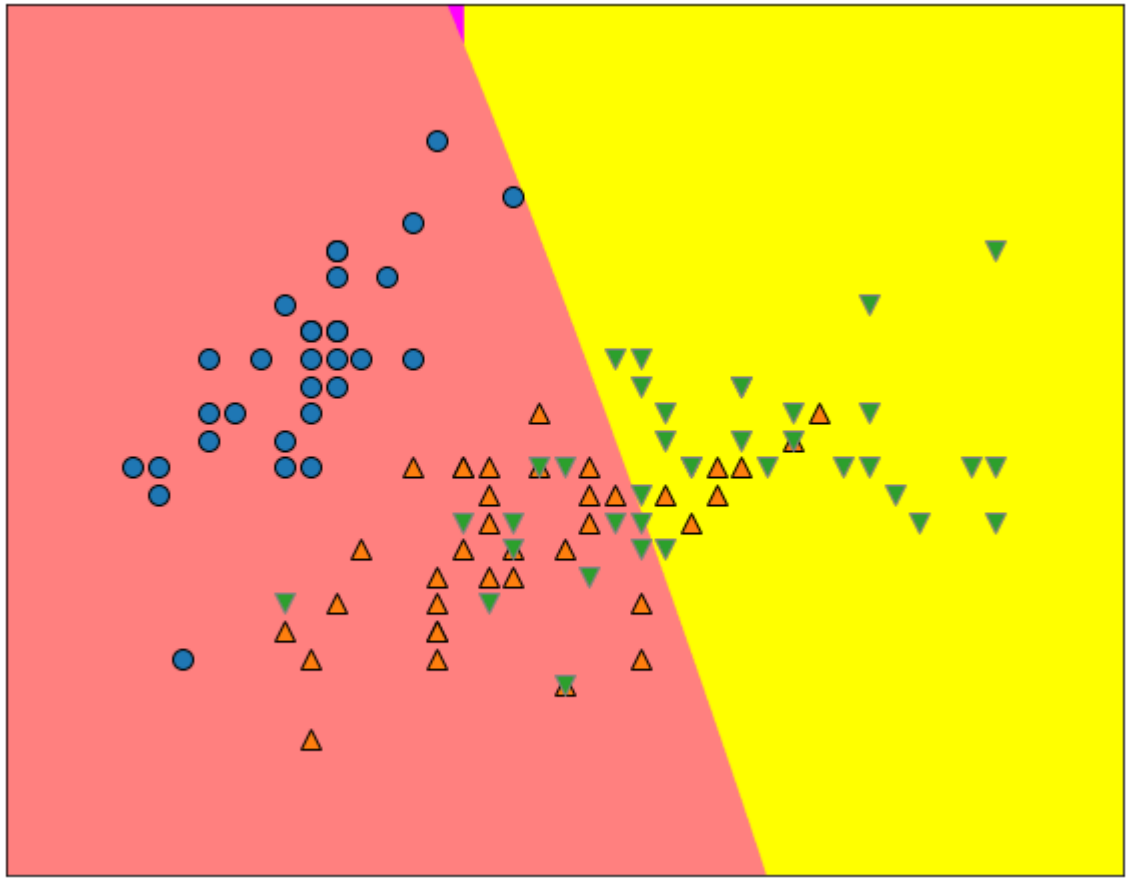
```
cmaps['Sequential (2)'] = [
            'binary', 'gist_yarg', 'gist_gray', 'gray', 'bone', 'pink',
            'spring', 'summer', 'autumn', 'winter', 'cool', 'Wistia',
            'hot', 'afmhot', 'gist_heat', 'copper']
```



2. Support Vector Machine

❖ SVM 실습

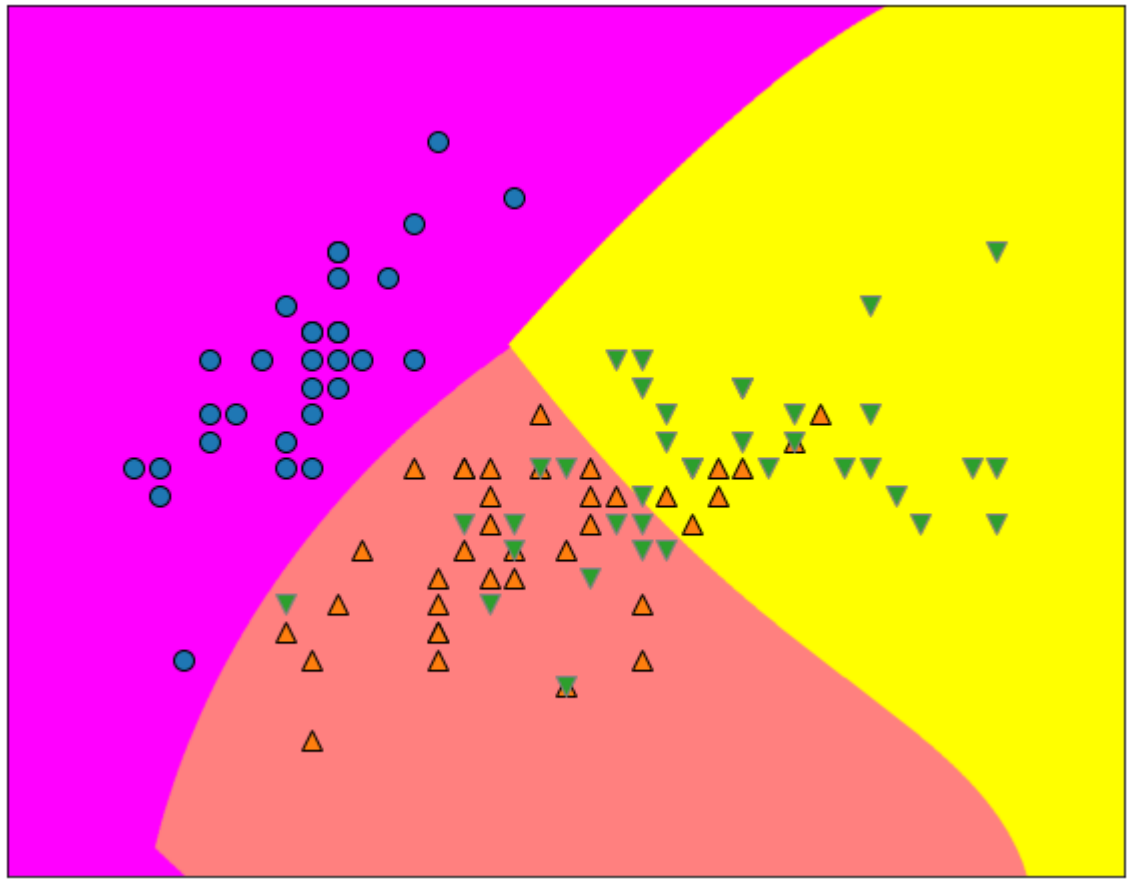
```
# 모델 정의
model = SVC(C=0.05) # C값에 민감하게 그래프가 변한다.

# 학습시키기
model.fit(X_train, y_train)

# 평가하기
score = model.score(X_train, y_train)
display(score)

plt.figure(figsize=[10,8])
mglearn.plots.plot_2d_classification(model, X_train,  cm='spring')
mglearn.discrete_scatter(X_train[:,0], X_train[:,1], y_train)
```



2. Support Vector Machine

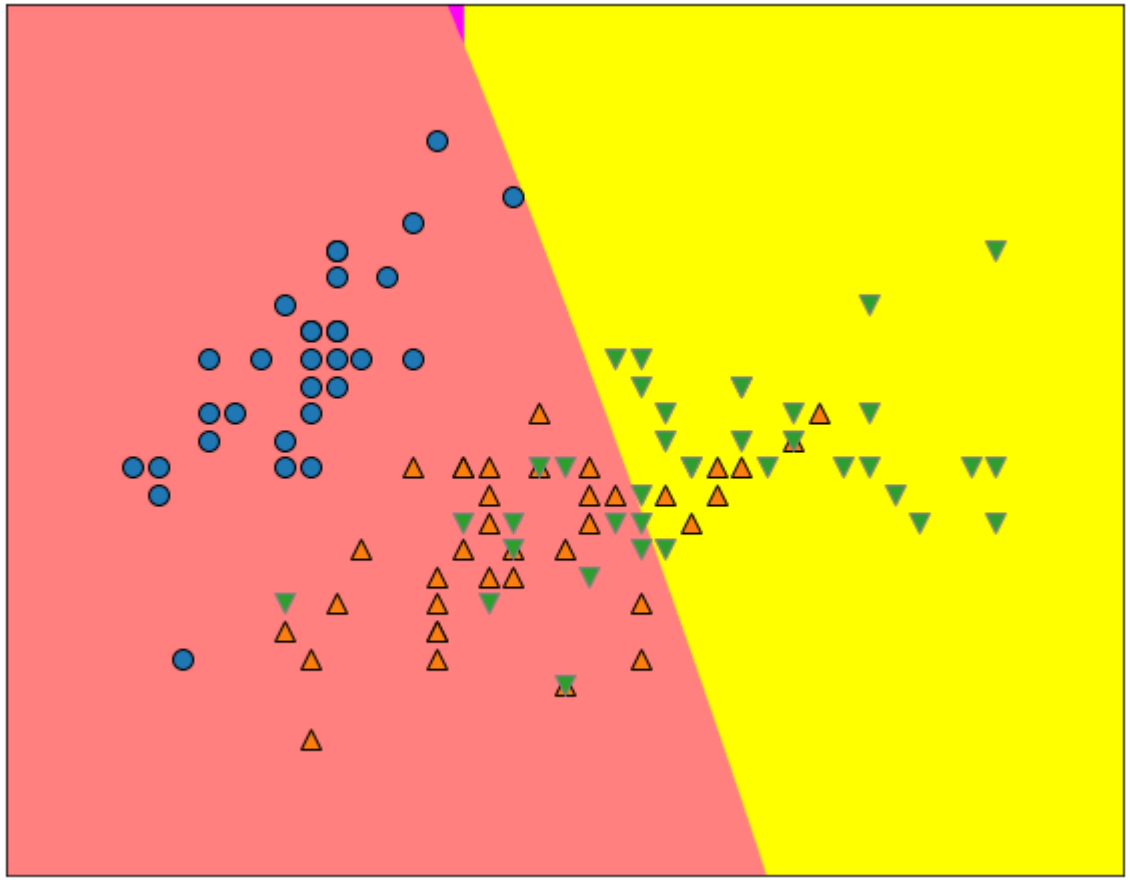
❖ SVM 실습

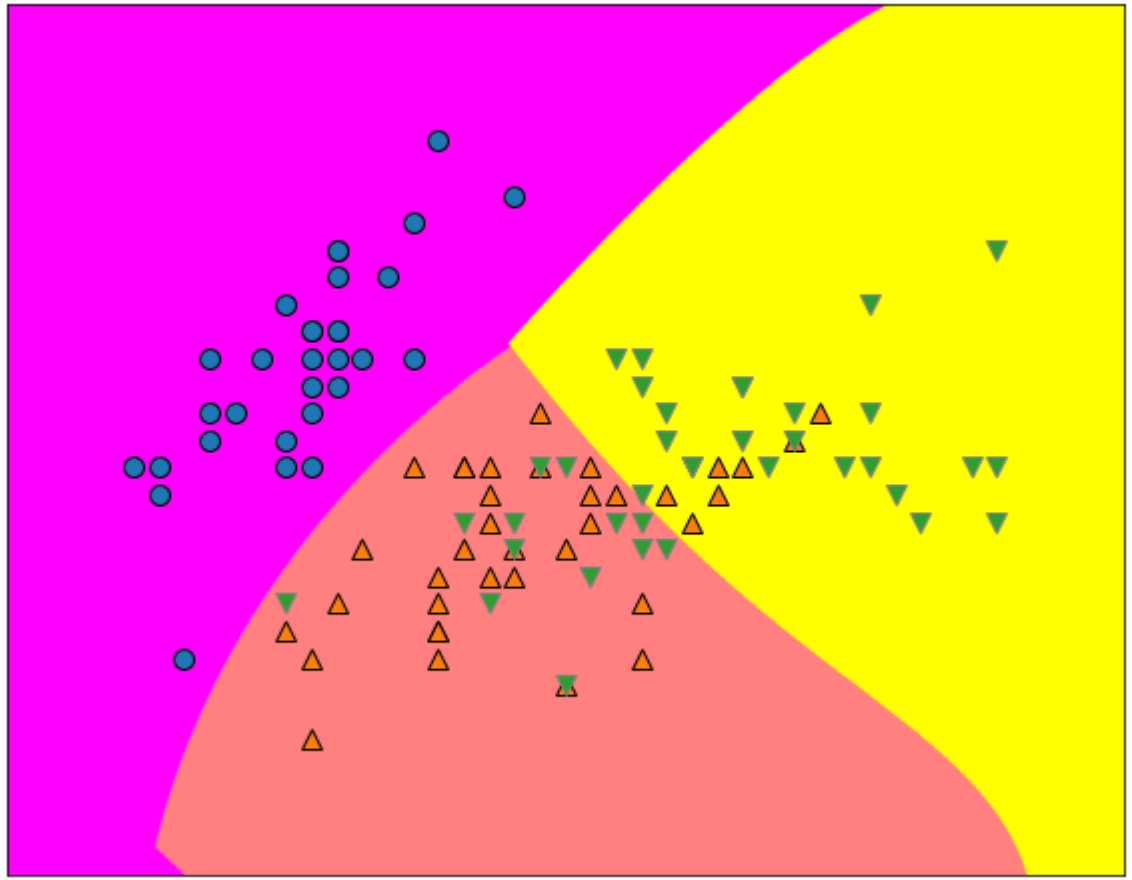
```
# 모델 정의
model = SVC(C=10) # C값에 민감하게 그래프가 변한다.

# 학습시키기
model.fit(X_train, y_train)

# 평가하기
score = model.score(X_train, y_train)
display(score)

plt.figure(figsize=[10,8])
mglearn.plots.plot_2d_classification(model, X_train, cm='spring')
mglearn.discrete_scatter(X_train[:,0], X_train[:,1], y_train)
```



2. Support Vector Machine

❖ SVM 실습

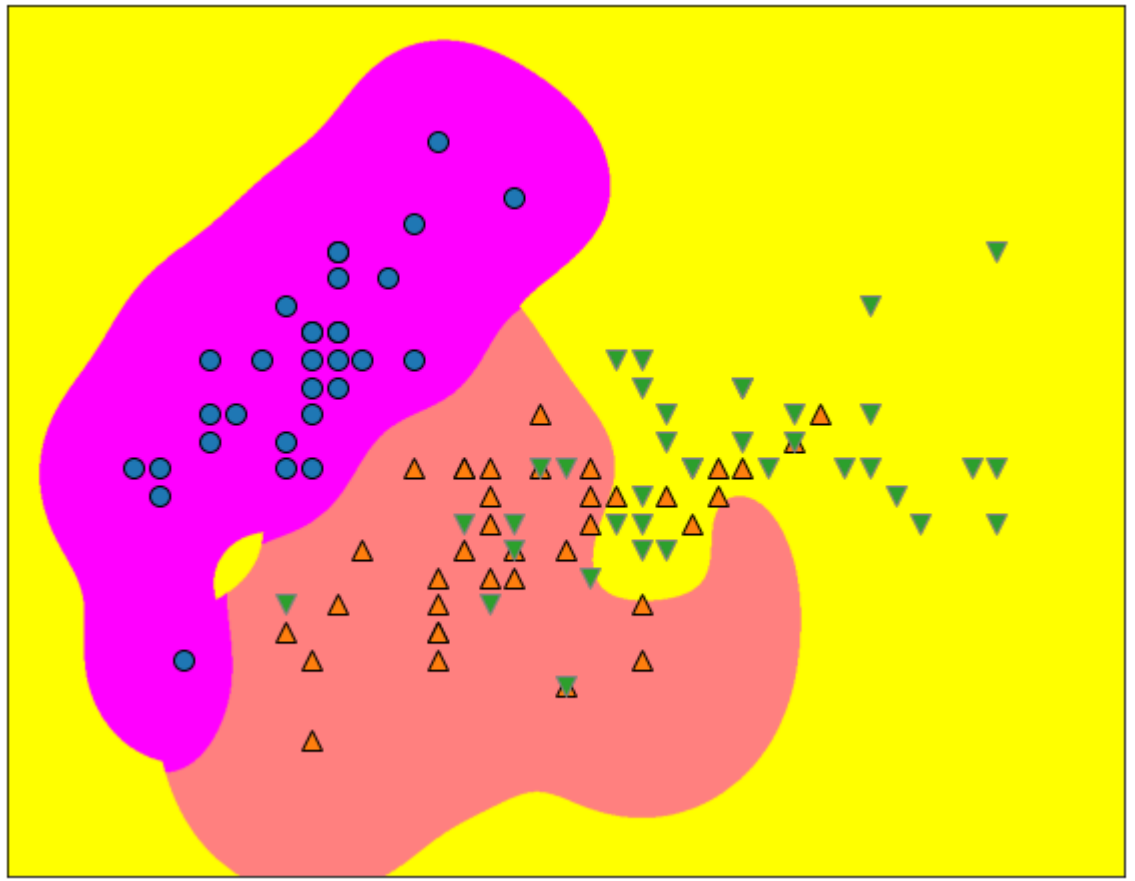




2. Support Vector Machine

❖ SVM 실습

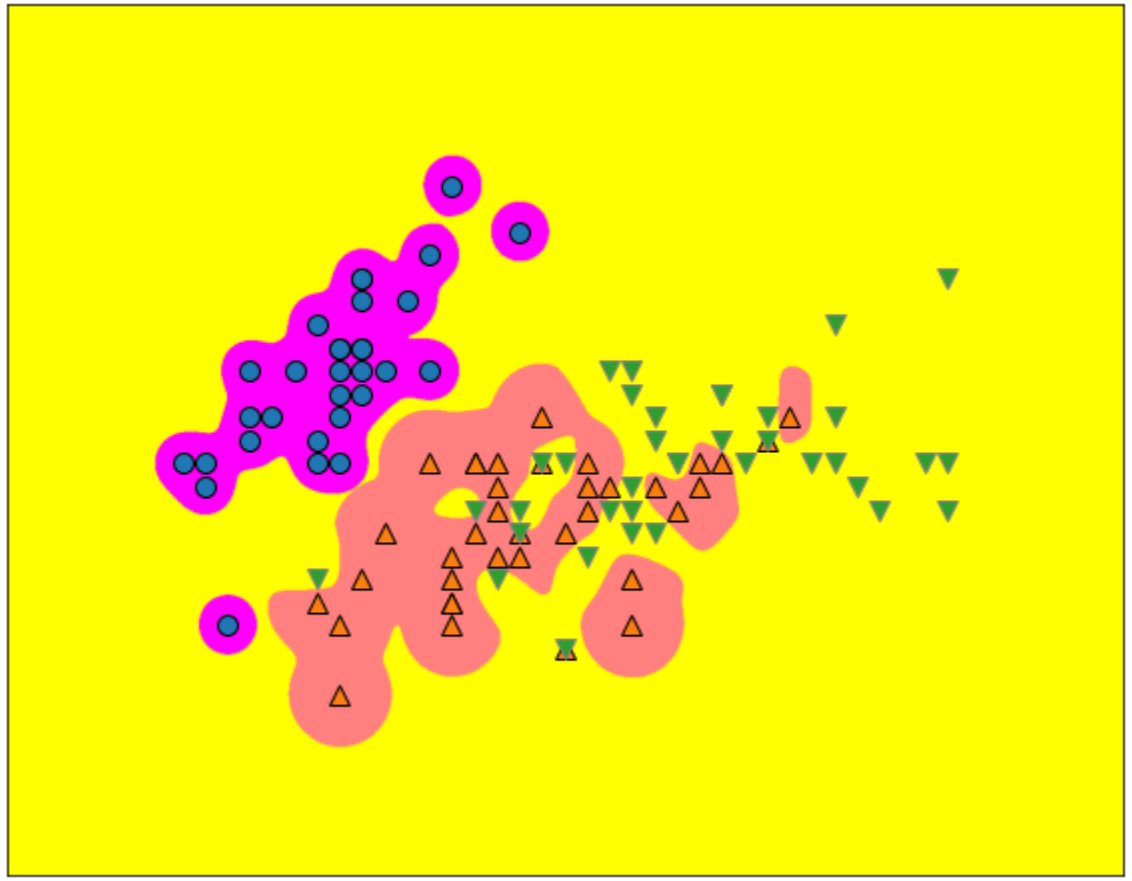
```
# 모델 정의
model = SVC(gamma=10) # C값에 민감하게 그래프가 변한다.

# 학습시키기
model.fit(X_train, y_train)

# 평가하기
score = model.score(X_train, y_train)
display(score)

plt.figure(figsize=[10,8])
mglearn.plots.plot_2d_classification(model, X_train,  cm='spring')
mglearn.discrete_scatter(X_train[:,0], X_train[:,1], y_train)
```



2. Support Vector Machine

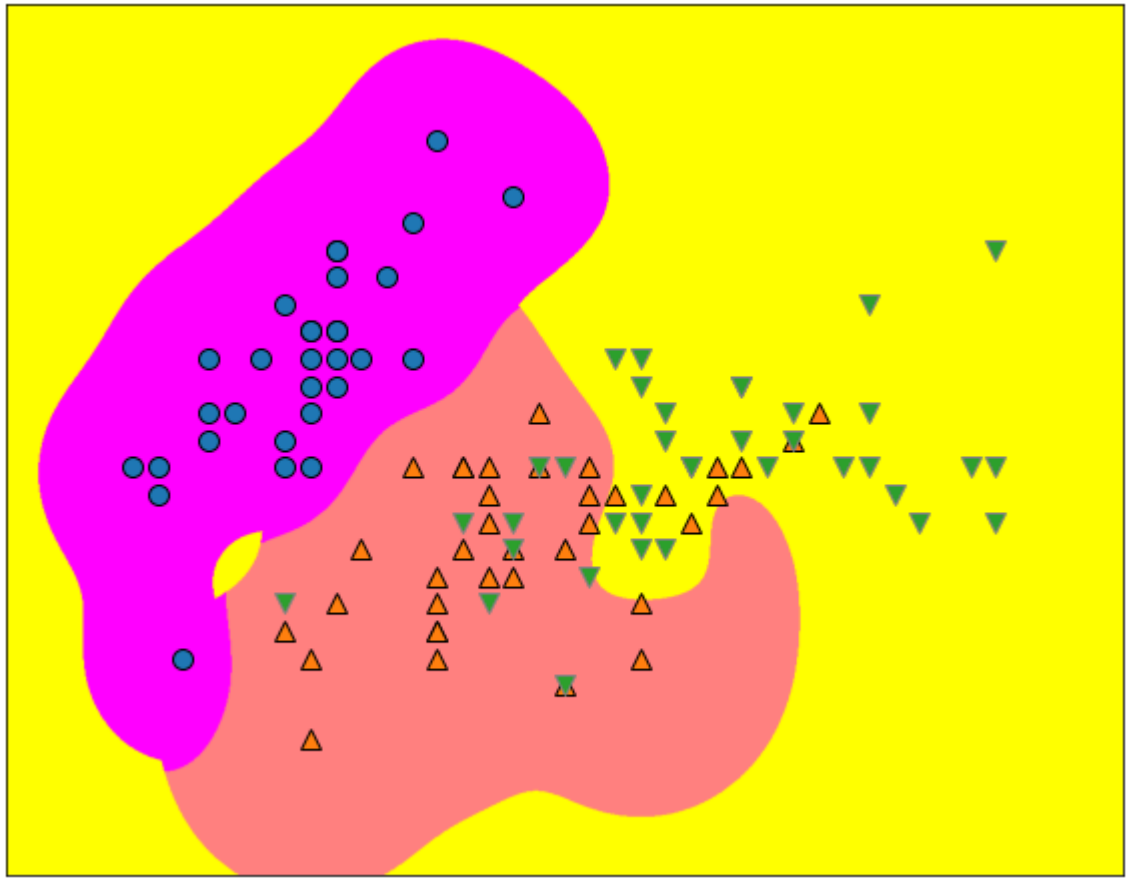
❖ SVM 실습

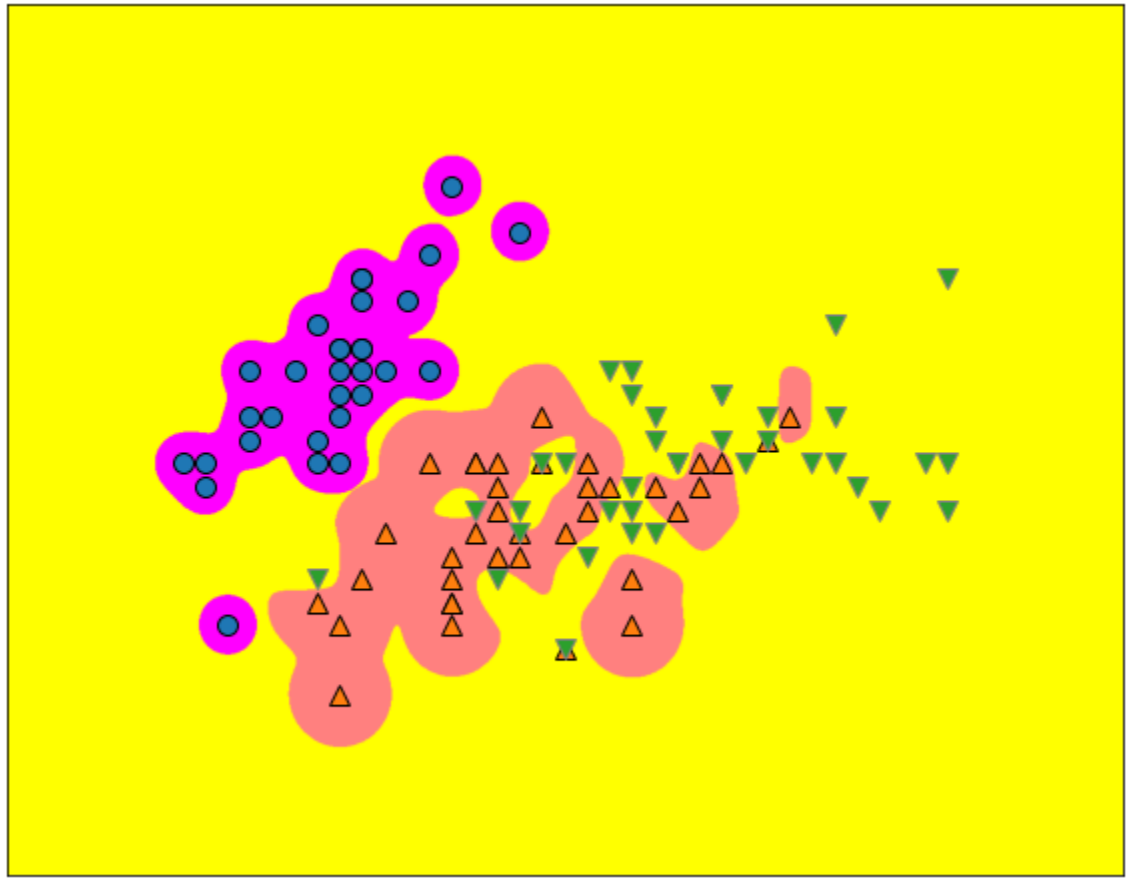
```
# 모델 정의
model = SVC(gamma=100) # C값에 민감하게 그래프가 변한다.

# 학습시키기
model.fit(X_train, y_train)

# 평가하기
score = model.score(X_train, y_train)
display(score)

plt.figure(figsize=[10,8])
mglearn.plots.plot_2d_classification(model, X_train, cm='spring')
mglearn.discrete_scatter(X_train[:,0], X_train[:,1], y_train)
```



2. Support Vector Machine

❖ SVM 실습

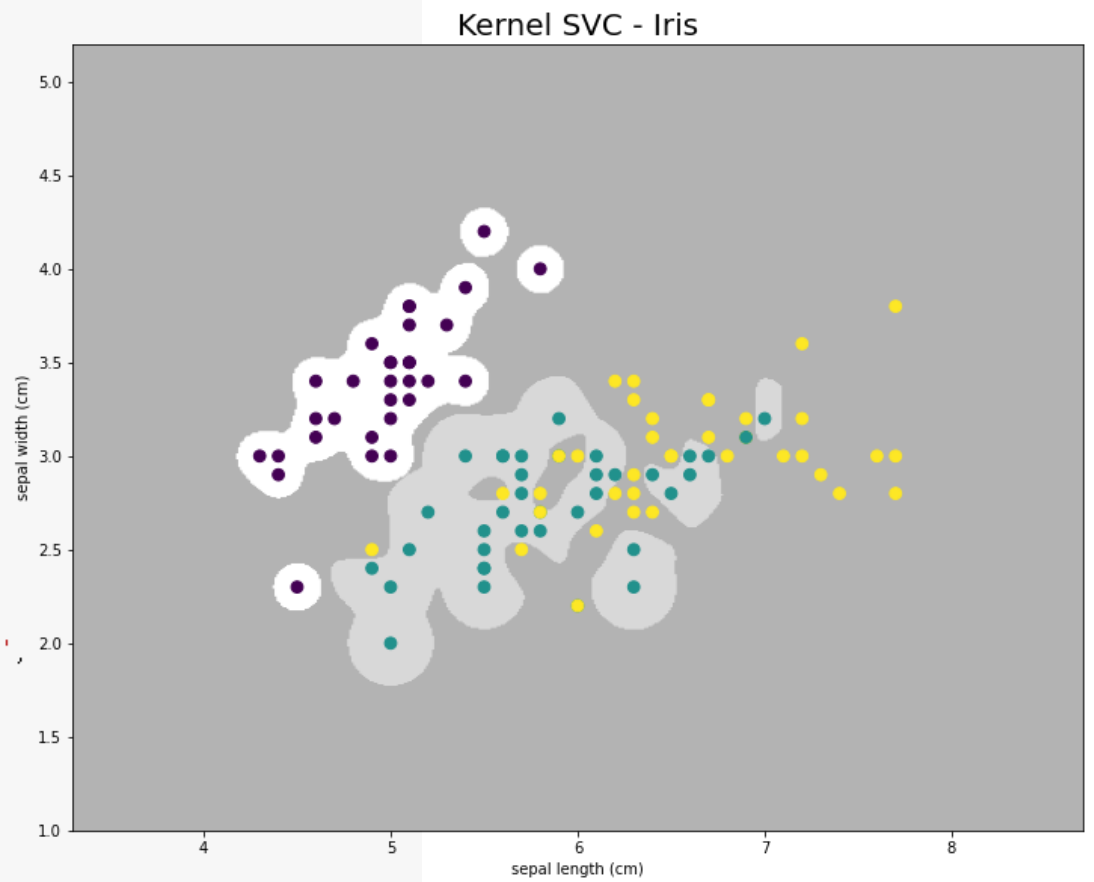




2. Support Vector Machine

❖ SVM 실습

```
import matplotlib.pyplot as plt
import numpy as np

scale = 500
xmax = X_train[:,0].max()+1
xmin = X_train[:,0].min()-1
ymax = X_train[:,1].max()+1
ymin = X_train[:,1].min()-1

xx = np.linspace(xmin,xmax,scale)
yy = np.linspace(ymin,ymax,scale)
data1, data2 = np.meshgrid(xx,yy)


X_grid = np.c_[data1.ravel(), data2.ravel()]
print(data1.ravel())
pred_y = model.predict(X_grid)

fig=plt.figure(figsize=[12,10])

CS = plt.imshow(pred_y.reshape(scale,scale), interpolation=None, origin='lower',
                extent=[xmin,xmax,ymin,ymax], alpha=0.3, cmap='gray_r')

# draw X_train
plt.scatter(X_train[:,0], X_train[:,1], c=y_train, s=60)

plt.xlabel(iris.feature_names[col1])
plt.ylabel(iris.feature_names[col2])
plt.title('Kernel SVC - Iris',fontsize=20)
```



2. Support Vector Machine

❖ SVM 실습

```
from sklearn.datasets import load_breast_cancer
breast_cancer_data = load_breast_cancer()


import pandas as pd
X_Data = pd.DataFrame(breast_cancer_data.data)
y = pd.DataFrame(breast_cancer_data.target)
```

```
# X_Data.info()
X_Data.describe()
```

	0	1	2	3	4	5	6	7	8	9	.
count	569.000000	569.000000	569.000000	569.000000	569.000000	569.000000	569.000000	569.000000	569.000000	569.000000	.
mean	14.127292	19.289649	91.969033	654.889104	0.096360	0.104341	0.088799	0.048919	0.181162	0.062798	.
std	3.524049	4.301036	24.298981	351.914129	0.014064	0.052813	0.079720	0.038803	0.027414	0.007060	.
min	6.981000	9.710000	43.790000	143.500000	0.052630	0.019380	0.000000	0.000000	0.106000	0.049960	.
25%	11.700000	16.170000	75.170000	420.300000	0.086370	0.064920	0.029560	0.020310	0.161900	0.057700	.
50%	13.370000	18.840000	86.240000	551.100000	0.095870	0.092630	0.061540	0.033500	0.179200	0.061540	.
75%	15.780000	21.800000	104.100000	782.700000	0.105300	0.130400	0.130700	0.074000	0.195700	0.066120	.
max	28.110000	39.280000	188.500000	2501.000000	0.163400	0.345400	0.426800	0.201200	0.304000	0.097440	.

2. Support Vector Machine

❖ SVM 실습

```
1  import sklearn.svm as svm
2  import sklearn.metrics as mt
3  from sklearn.model_selection import cross_val_score, cross_validate
4
5  # SVM, kernel = 'linear'로 선형분리 진행
6
7  svm_clf =svm.SVC(kernel = 'linear')
8
9  # 교차검증
10 scores = cross_val_score(svm_clf, X_Data, y, cv = 5)
11 scores
12
13 print('교차검증 평균: ', scores.mean())
14 print(pd.DataFrame(cross_validate(svm_clf, X_Data, y, cv =5)))
```

```
교차검증 평균:  0.9455364073901569
```

```
   fit_time  score_time  test_score
0  0.795871    0.000999    0.947368
1  1.850051    0.000999    0.929825
2  1.197795    0.000999    0.973684
3  0.604382    0.001996    0.921053
4  1.052186    0.000998    0.955752
```

2. Support Vector Machine

❖ SVM 실습

```
# SVM, kernel = 'rbf'로 비선형분리 진행

svm_clf =svm.SVC(kernel = 'rbf')

# 교차검증

scores = cross_val_score(svm_clf, X_Data, y, cv = 5)
scores

print(pd.DataFrame(cross_validate(svm_clf, X_Data, y, cv =5)))

print('cross Mean: ', scores.mean())

```

```
   fit_time  score_time  test_score
0  0.005983    0.001975    0.850877
1  0.005964    0.001024    0.894737
2  0.005984    0.001967    0.929825
3  0.005984    0.002016    0.947368
4  0.005963    0.002017    0.938053
cross Mean:  0.9121720229777983
```

2. Support Vector Machine

❖ SVM 실습

```
from sklearn.preprocessing import StandardScaler

scaler = StandardScaler()
scaler.fit(X_Data)
X_scaled = scaler.transform(X_Data)

from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X_scaled, y, test_size = 0.3, random_state = 42)

import sklearn.svm as svm
import sklearn.metrics
from sklearn.model_selection import cross_val_score, cross_validate

svm_clf =svm.SVC(kernel = 'linear')

scores = cross_val_score(svm_clf, X_scaled, y, cv = 5)
scores

print(pd.DataFrame(cross_validate(svm_clf, X_scaled, y, cv =5)))

print('교차검증 평균: ', scores.mean())

```

```
   fit_time  score_time  test_score
0  0.003988    0.000997    0.956140
1  0.003984    0.000997    0.982456
2  0.003997    0.001000    0.964912
3  0.003987    0.000000    0.964912
4  0.002994    0.000995    0.982301
교차검증 평균:  0.9701443875174661
```

2. Support Vector Machine

❖ SVM 실습

```
from sklearn.model_selection import GridSearchCV

# 테스트하고자 하는 파라미터 값들을 사전타입으로 정의

svm_clf = svm.SVC(kernel = 'linear',random_state=42)
parameters = {'C': [0.001, 0.01, 0.1, 1, 10, 25, 50, 100]}

grid_svm = GridSearchCV(svm_clf,
                        param_grid = parameters, cv = 5)

grid_svm.fit(X_train, y_train)
print(grid_svm.best_params_)          # 좋은 파라미터를 보여줌,
print(grid_svm.best_score_)


result = pd.DataFrame(grid_svm.cv_results_['params'])
result['mean_test_score'] = grid_svm.cv_results_['mean_test_score']
result.sort_values(by='mean_test_score', ascending=False)

```

	C	mean_test_score
3	1.000	0.974810
2	0.100	0.972310
4	10.000	0.964778
7	100.000	0.959810
5	25.000	0.959778
6	50.000	0.957310
1	0.010	0.952215
0	0.001	0.927057

```
model=grid_svm.best_estimator_  # 최적의 파라미터로 모델 생성
y_pred=model.predict(X_test)

from sklearn import metrics

print("Accuracy:", metrics.accuracy_score(y_test, y_pred))

```

```
Accuracy: 0.9766081871345029
```

2. Support Vector Machine

❖ SVM 실습

```
from sklearn.model_selection import GridSearchCV

# 테스트하고자 하는 파라미터 값들을 사전타입으로 정의

svm_clf = svm.SVC(kernel = 'rbf',random_state=100)
parameters = {'C': [0.001, 0.01, 0.1, 1, 10, 25, 50, 100],
              'gamma':[0.001, 0.01, 0.1, 1, 10, 25, 50, 100]}

grid_svm = GridSearchCV(svm_clf,
                        param_grid = parameters, cv = 5)


grid_svm.fit(X_train, y_train)
print(grid_svm.best_params_)          # 좋은 파라미터를 보여줌,
print(grid_svm.best_score_)

model=grid_svm.best_estimator_  # 최적의 파라미터로 모델 생성
y_pred=model.predict(X_test)

from sklearn import metrics

print("Accuracy:", metrics.accuracy_score(y_test, y_pred))


result = pd.DataFrame(grid_svm.cv_results_['params'])
result['mean_test_score'] = grid_svm.cv_results_['mean_test_score']
result.sort_values(by='mean_test_score', ascending=False)
```

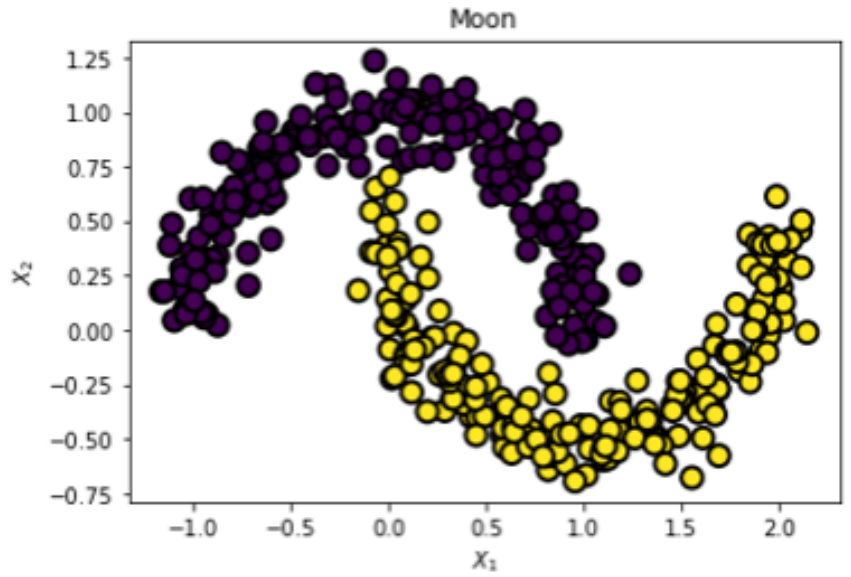
```
{'C': 50, 'gamma': 0.001}
0.9748417721518987
Accuracy: 0.9824561403508771
```

	C	gamma	mean_test_score
48	50.0	0.001	0.974842
40	25.0	0.001	0.972310
33	10.0	0.010	0.967278
57	100.0	0.010	0.967278
56	100.0	0.001	0.967247
...	...	...	...
28	1.0	10.000	0.625633
29	1.0	25.000	0.625633
30	1.0	50.000	0.625633
31	1.0	100.000	0.625633
63	100.0	100.000	0.625633

2. Support Vector Machine

❖ SVM 실습

```
from sklearn.datasets import make_moons
import matplotlib.pyplot as plt

plt.title("Moon")
X, y = make_moons(n_samples=400, noise=0.1, random_state=0)
plt.scatter(X[:, 0], X[:, 1], marker='o', c=y, s=100,
            edgecolor="k", linewidth=2)
plt.xlabel("$X_1$")
plt.ylabel("$X_2$")
plt.show()
```



2. Support Vector Machine

❖ SVM 실습

```
import sklearn.svm as svm
import sklearn.metrics as mt
from sklearn.model_selection import cross_val_score, cross_validate
from sklearn.model_selection import train_test_split, cross_val_score
import pandas as pd

X_train, X_test, y_train, y_test = train_test_split(X, y,
                                                    test_size = 0.3, random_state = 100)


svm_clf =svm.SVC(kernel = 'linear', random_state=100)

scores = cross_val_score(svm_clf, X, y, cv = 5)
scores



print(pd.DataFrame(cross_validate(svm_clf, X, y, cv =5)))
print('linear: ', scores.mean())


svm_clf =svm.SVC(kernel = 'rbf')

scores = cross_val_score(svm_clf, X, y, cv = 5)
scores

print(pd.DataFrame(cross_validate(svm_clf, X, y, cv =5)))
print('nonlinear: ', scores.mean())

```

```
   fit_time  score_time  test_score
0  0.000997    0.000998      0.8500
1  0.000998    0.000000      0.8875
2  0.000998    0.000000      0.8375
3  0.000997    0.000997      0.8625
4  0.000998    0.000000      0.9500
linear:  0.8775000000000001
   fit_time  score_time  test_score
0  0.000998    0.000000      1.0000
1  0.000999    0.000000      1.0000
2  0.000000    0.000998      1.0000
3  0.000997    0.000000      0.9750
4  0.000000    0.001000      0.9875
nonlinear:  0.9925
```

감사합니다