머신러닝응용 제06강

K-NN & Naïve Bayes

첨단공학부 김동하교수

제06강 K-NN & Naïve Bayes

1	K-NN 방법론에 대해 학습한다.
2	유사도를 측정하는 다양한 거리에 대해 학습한다.
3	Naïve Bayes 방법론에 대해 학습한다.
4	Gaussian NB와 QDA의 관계를 파악한다.

핵심 단어

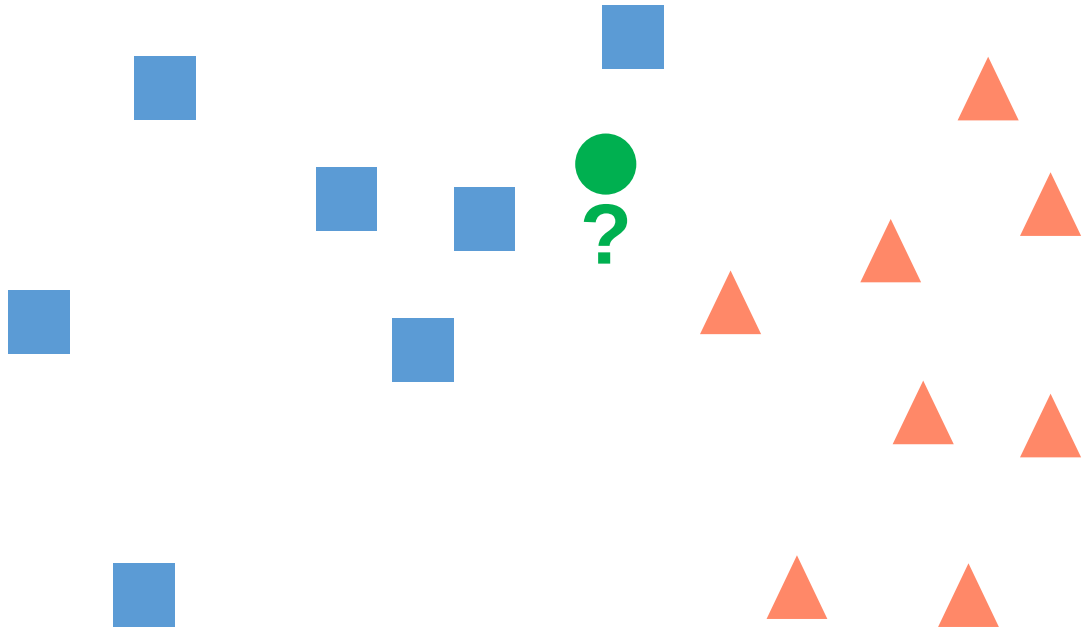
- Distance
- K-NN
- Naïve Bayes
- Gaussian Naïve Bayes

06강. K-NN & Naïve Bayes

01. K-NN Classifier

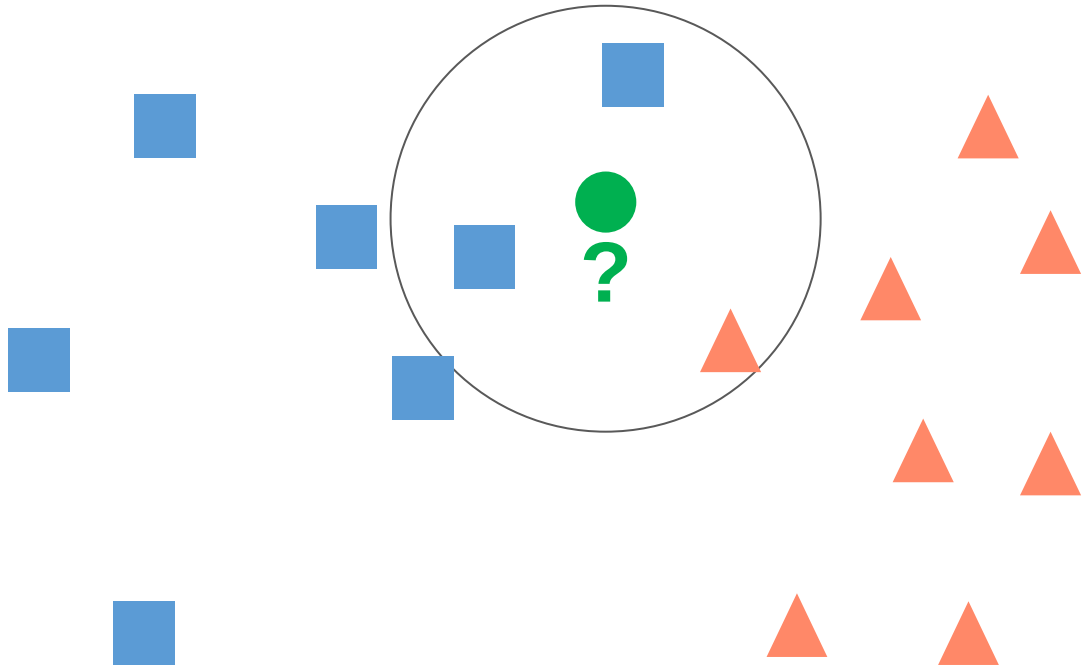
1) K-Nearest Neighbors

- 분류 문제: 다음 데이터는 어느 집단에 속할까?



1) K-Nearest Neighbors

- 가장 가까운 K개 중 다수가 속한 집단으로 분류하자.
  - K-Nearest Neighbors



K=3

■ : 2

▲ : 1

● : ■

1) K-Nearest Neighbors

K-NN 이란?

- 가장 가까이 있는 K개의 자료들 (K-nearest neighbors)를 이용해 분류나 예측 결과를 이끌어내는 기법

K-NN의 절차

- 각 데이터에 대한 거리 계산
- 가장 가까운 K개의 데이터 결정
- 최종 예측값 결정

2) K-NN로 예측하기

- x의 가장 가까운 데이터들을 $\tilde{x}_1, \cdots, \tilde{x}_k$라 하고 이들의 라벨을 각각 $\tilde{y}_1, \cdots, \tilde{y}_k$라 하자.
- 회귀 문제의 경우

$$\hat{y} = \frac{1}{k}\sum_{i=1}^{k} \tilde{y}_i$$

- 분류 문제의 경우
  - $\tilde{y}_1, \cdots, \tilde{y}_k$ 의 값들 중 최빈값을 이용해서 예측한다.

3) K-NN vs Linear model

모형의 형태

- 선형모형은 자료를 생성하는 과정이 선형이라 가정
- K-NN은 특별한 가정이 없다.

모형의 학습

- 선형모형은 최소제곱법을 통해 모수를 학습한다.
- K-NN은 모형을 학습할 필요가 없다.
  - 가까운 데이터만을 구하면 됨.

4) 다양한 거리

- K-NN 방법론을 결정하는 가장 중요한 요소는 데이터 사이의 유사도를 결정하는 거리이다.
  - 거리의 정의가 달라지면 예측 결과도 달라짐.
  - 거리를 어떻게 정의하는지가 중요.

4) 다양한 거리

수치형 데이터 사이의 거리

- 2차원의 경우: $(a_1, a_2), (b_1, b_2)$

- Euclidean distance

$$\sqrt{(a_1 - b_1)^2 + (a_2 - b_2)^2}$$

- Manhattan distance

$$|a_1 - b_1| + |a_2 - b_2|$$

4) 다양한 거리

수치형 데이터 사이의 거리

- Minkowski distance

$$(|a_1 - b_1|^q + |a_2 - b_2|^q)^{1/q}$$

- 대표적으로 Euclidean 거리를 사용.

- 다양한 설명변수들의 척도를 균등하게 하기 위하여 거리 계산에 앞서 데이터 표준화 작업을 먼저 하는 것이 좋다.

4) 다양한 거리

- 범주형 데이터 사이의 거리
  - 범주형 데이터 사이의 유사도를 측정하는 다양한 거리가 존재한다.
    - Hamming distance
    - Jaccard distance
    - 등등

5) K의 결정

- K가 너무 작으면 지엽적인 정보만을 이용하여 예측
- K가 너무 크면 데이터에 상관없이 비슷한 결과를 예측
- 일반적으로는 1에서 20 사이의 범위에서 최적의 K를 결정
  - 검증 데이터, 교차 검증법을 이용
  - 오분류율이 가장 낮은 K를 선택

5) K-NN의 확장

- K-NN은 세개 이상의 다범주 분류 문제에서도 쉽게 적용 가능.
  - 거리를 계산하여 K개의 이웃을 선정
  - 선정한 이웃의 라벨값의 최빈값을 예측값으로 사용

5) K-NN의 확장

- 회귀 모형으로도 쉽게 확장 가능.
  - 거리를 계산하여 K개의 이웃을 선정
  - 선정한 이웃의 종속변수값들의 평균을 예측값으로 사용.
  - 최적의 K를 계산할 때 오분류율 대신 제곱손실값을 사용.

6) K-NN의 장단점

장점

- 단순하고, 모수에 대한 가정이 없어 쉽게 이용 가능.
- 충분히 많은 학습 데이터가 있을 경우 좋은 성능.

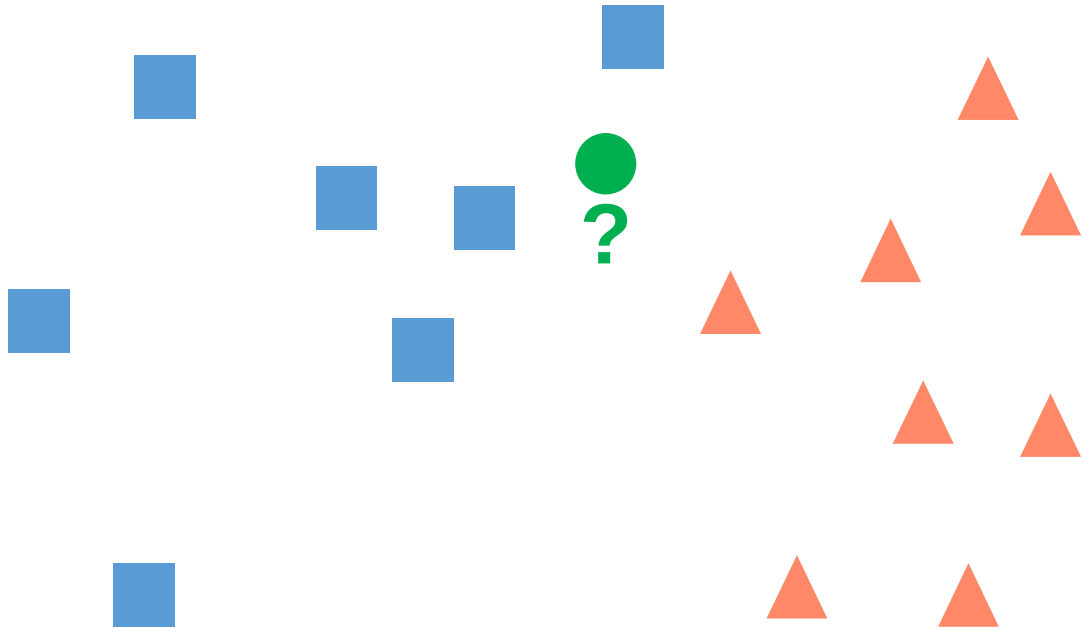
단점

- 이웃을 파악하는데 많은 계산 시간 소요.
- 설명변수의 차원수가 커질 경우 일반적으로 성능 저하.

02. Naïve Bayes

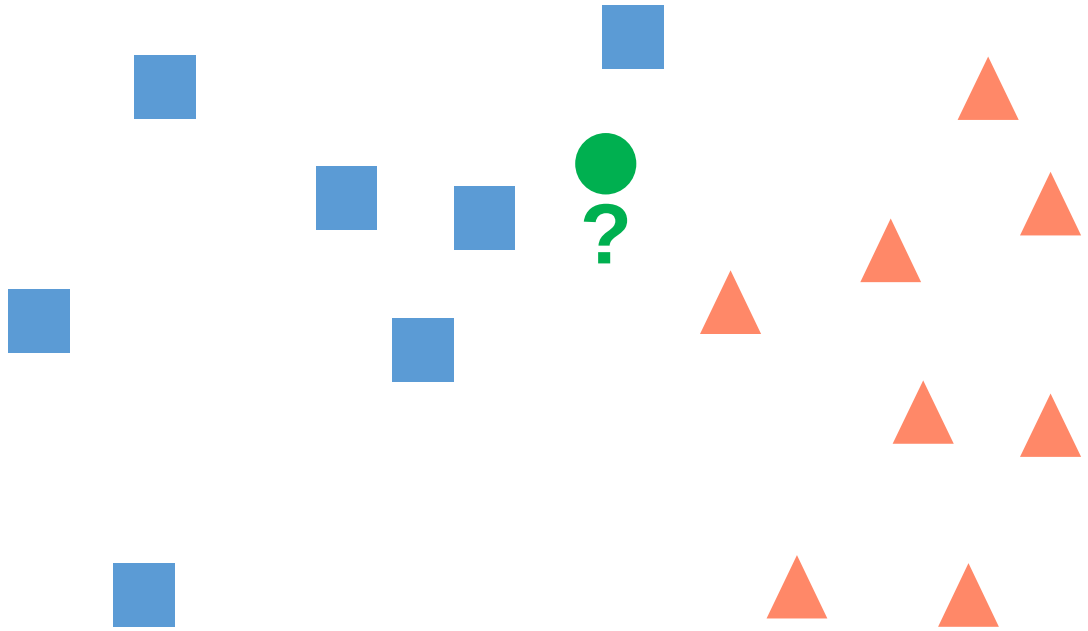
1) Naïve Bayes란?

- 분류 문제: 다음 데이터는 어느 집단에 속할까?



1) Naïve Bayes란?

- 베이즈 정리에 근거하여 해당 데이터에 대한 종속변수의 확률을 통해 분류하자.
  - Naïve Bayes



2) 베이즈 정리

- 조건부 확률을 이용해 사전 확률과 사후 확률의 관계를 밝히는 정리.
- A: 어떤 사건
- $B_1, \cdots, B_k$: 배반 사건들
  - $P\left(\cup_{i=1}^{k} B_i\right) = 1$를 만족한다고 가정.
- 이 때, 다음의 등식이 성립한다.

$$P(B_j|A) = \frac{P(A \cap B_j)}{P(A)} = \frac{P(A|B_j)P(B_j)}{\sum_{i=1}^{k} P(A|B_i)P(B_i)}$$

3) Bayes classifier

- X: 설명변수
- Y: 종속변수 (편의상 이진 변수를 가정)
- 베이즈 분류기는 다음과 같이 정의된다.

$$\hat{y} = argmax_{y \in \{0,1\}} P(X = x | Y = y) P(Y = y)$$

4) Naïve Bayes

- X가 p차원이라 하자.
  - $X = (X_1, \cdots, X_p)$

- Naïve Bayes는 종속변수가 주어졌을 때 설명 변수 각각의 원소가 독립이라는 가정을 한다.
  - $P(X = (x_1, \cdots, x_p)|Y = y) = \prod_{j=1}^{p} P(X_j = x_j|Y = y)$

4) Naïve Bayes

- 즉, Naïve Bayes는 다음의 결합 분포를 가정한다.

$$P(X = x, Y = y) = P(Y = y) \prod_{j=1}^{p} P(X_j = x_j | Y = y)$$

- Naïve Bayes를 이용해서 다음과 같이 예측한다.

$$\hat{y} = argmax_{y \in \{0,1\}} P(Y = y) \prod_{j=1}^{p} P(X_j = x_j | Y = y)$$

5) Gaussian Naïve Bayes

- $P(X_j = x_j | Y = y)$를 정규 분포로 모형화

$$Y \sim Ber(\pi)$$

$$X_j | Y = y \sim N(\mu_{jy}, \sigma_{jy}^2)$$

- 추정해야 하는 모수:
  - $\pi, \mu_{jy}, \sigma_{jy}^2, \mathrm{j} = 1, \cdots, p, and\ y = 0,1$

5) Gaussian Naïve Bayes

- 학습 데이터를 이용한 음의 로그우도함수를 최소화하는 모수를 추정한다.
  - 학습 데이터: $(x_1, y_1), \cdots, (x_n, y_n)$
  - 음의 로그우도 함수

$$-\sum_{i=1}^{n} \log P(Y = y_i, X = x_i)$$

6) Gaussian Naïve Bayes vs QDA

- Gaussian Naïve Bayes 모형은 QDA 모형의 특별한 형태이다.

$$Y \sim Ber(\pi)$$
$$X|Y = y \sim N(\mu_y, \Sigma_y)$$

  - $\mu_y = (\mu_{1y}, \cdots, \mu_{py})$
  - $\Sigma_y = diag(\sigma_{j1}^2, \cdots, \sigma_{jp}^2)$
  - 즉, GNB는 QDA에서 공분산 행렬을 대각행렬로 사용한 형태이다.

06강. K-NN & Naïve Bayes

03. Python을 이용한 실습

1) 데이터 설명

- Iris 데이터셋
  - 150개의 붓꽃에 대한 데이터
  - 꽃잎의 각 부분의 너비와 길이 등을 측정
  - sepal length: 꽃받침의 길이
  - sepal width: 꽃받침의 너비
  - petal length: 꽃잎의 길이
  - petal width: 꽃잎의 너비
  - species: 붓꽃의 종류 (setosa, versicolor, virginica)

1) 데이터 설명

분석 목표

- 붓꽃의 꽃받침과 꽃잎의 정보로 종류를 예측하는 K-NN, Naïve Bayes를 학습하자.

2) 환경설정

◆필요한 패키지 불러오기

```
import os
import numpy as np
import pandas as pd
import collections
from sklearn.model_selection import train_test_split
from sklearn.neighbors import KNeighborsClassifier
from sklearn.naive_bayes import GaussianNB
from sklearn.preprocessing import StandardScaler
from sklearn.metrics import confusion_matrix
import seaborn as sns
import matplotlib.pyplot as plt
```

3) 데이터 불러오기

◆sns 패키지에 내장되어있는 iris 데이터를 불러오자.

```
iris = sns.load_dataset('iris')
print(iris.shape)
iris.head()
```

```
(150, 5)
```

	sepal_length	sepal_width	petal_length	petal_width	species
0	5.1	3.5	1.4	0.2	setosa
1	4.9	3.0	1.4	0.2	setosa
2	4.7	3.2	1.3	0.2	setosa

4) 데이터 전처리

- 종속변수와 설명변수 구분
- 데이터를 분할하고, 설명변수값을 표준화한다.

```
X = iris[['sepal_length','sepal_width',\
                'petal_length','petal_width']]
y = iris.iloc[:,4]
```

```
X_train, X_test, y_train, y_test = \
train_test_split(X, y, test_size = 0.3, random_state=123)
```

```
scaler = StandardScaler()
scaler.fit(X_train)
X_train = pd.DataFrame(scaler.transform(X_train))
X_test = pd.DataFrame(scaler.transform(X_test))
```

5) K-NN 분석

- 검증자료를 활용하여 최적의 K값을 찾는다.
- 학습 자료를 다시 분할하여 일부를 검증자료로 사용.

```
X_tr_temp, X_val_temp, y_tr_temp, y_val_temp = \
train_test_split(X_train, y_train, test_size = 0.3, random_state=1234)
```

5) K-NN 분석

- 검증자료를 활용하여 최적의 K값을 찾는다.
- 학습 자료를 다시 분할하여 일부를 검증자료로 사용.

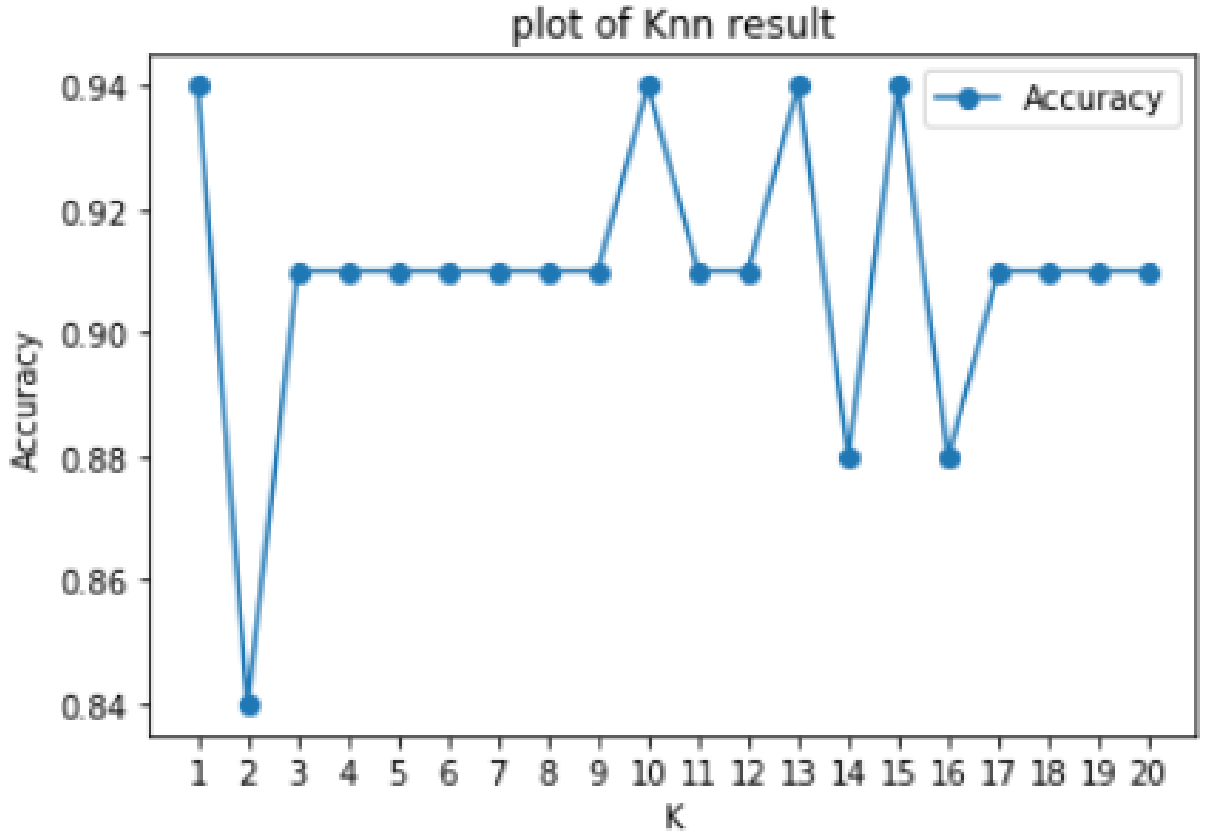
```
k_grid = range(1, 21, 1)
acc_list = []

for k_ in k_grid:
    model = KNeighborsClassifier(n_neighbors=k_)
    model.fit(X_tr_temp, y_tr_temp)
    y_pred = model.predict(X_val_temp)
    sub_acc = round((y_pred == y_val_temp).mean(), 2)
    acc_list.append(sub_acc)

KnnRes_df = pd.DataFrame({'k' : k_grid, 'Accuracy' : acc_list})
```

5) K-NN 분석

- 각 K마다 검증자료에서의 성능 확인하기.
  - 1이 적당해보임.



5) K-NN 분석

◆최적의 K를 이용하여 최종 K-NN 모형 학습.

```
KnnoptimalK = KnnRes_df['k'].iloc[KnnRes_df['Accuracy'].idxmax()]
print(KnnoptimalK)
knn_model = KNeighborsClassifier(n_neighbors=KnnoptimalK)
knn_model.fit(X_train, y_train)
```

5) K-NN 분석

◆ 이후에 시험 자료에 적합하여 정분류율 및 오차행렬을 구한다.

```
y_test_pred_knn = knn_model.predict(X_test)
confusion_matrix(y_test, y_test_pred_knn)
```

```
array([[18,  0,  0],
       [ 0,  9,  1],
       [ 0,  2, 15]])
```

```
knn_model.score(X_test, y_test)
```

```
0.9333333333333333
```

6) Gaussian NB 분석

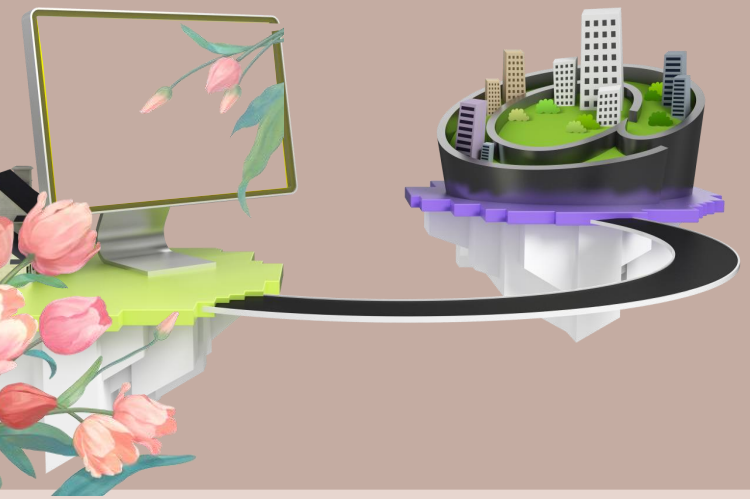
- Gaussian Naïve Bayes 모형 학습.
- 시험 데이터에 대입하여 분류 정확도와 오차행렬을 계산한다.

```
GNB_model = GaussianNB().fit(X_train, y_train)
y_test_pred_GNB = GNB_model.predict(X_test)
confusion_matrix(y_test, y_test_pred_GNB)
```

```
array([[18,  0,  0],
       [ 0, 10,  0],
       [ 0,  2, 15]])
```

```
GNB_model.score(X_test, y_test)
```

```
0.9555555555555556
```



다음시간안내

제07강

Principal Component Analysis