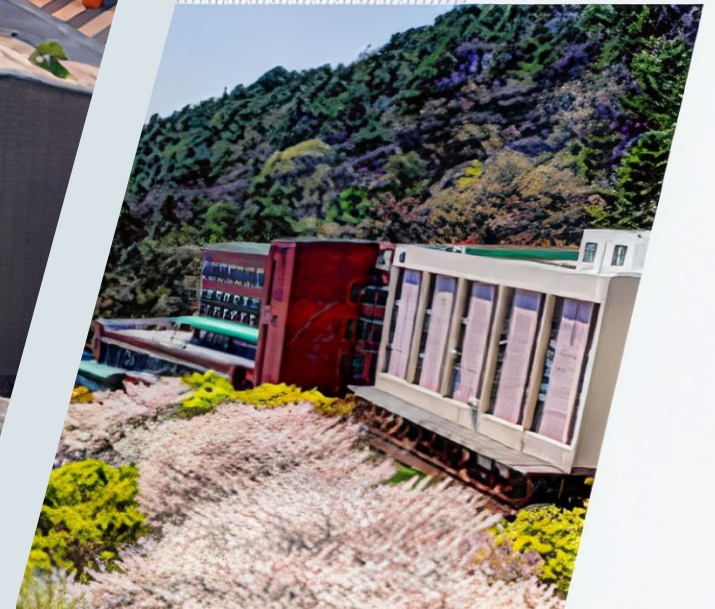


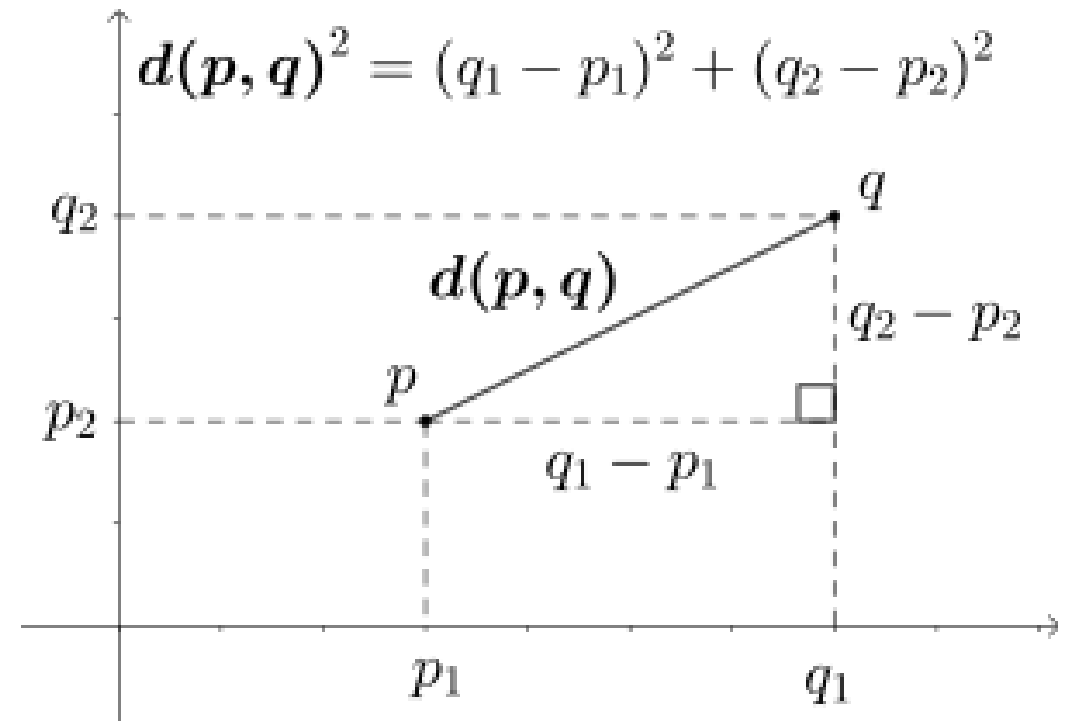
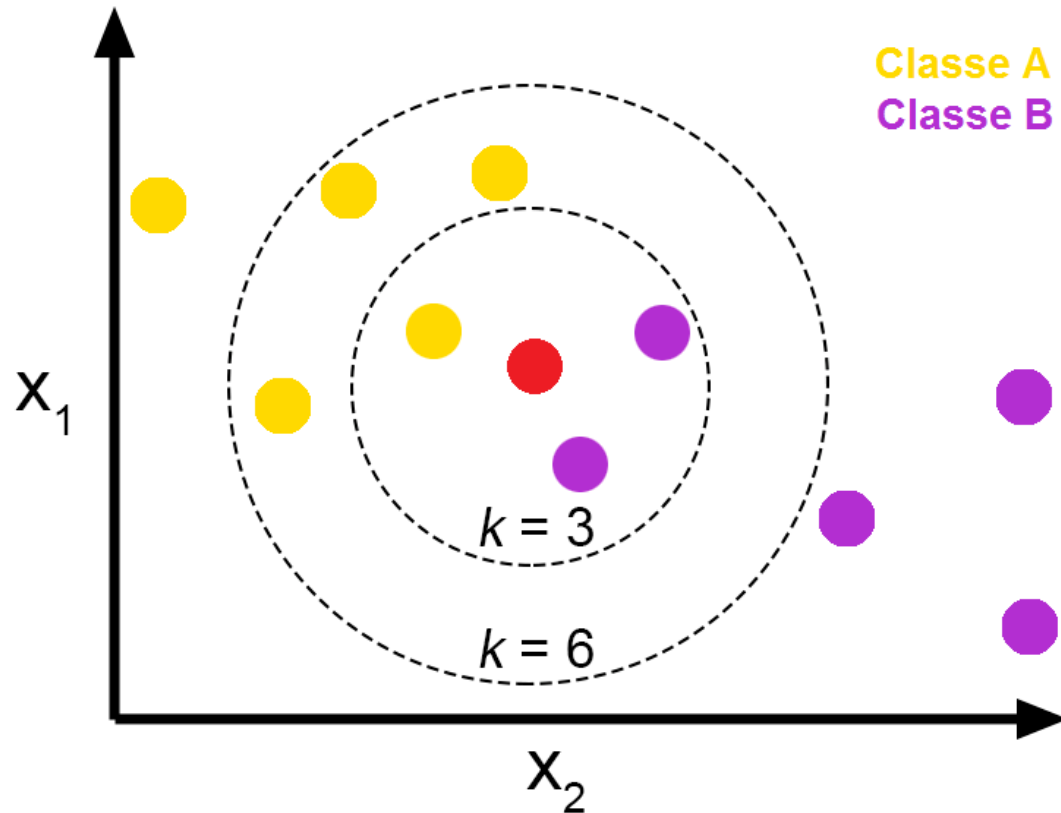
K Nearest Neighbors – 실습

컴퓨터AI공학부
2024년 2학기 머신러닝



Review – K Nearest Neighbors

- 목적: 새로운 샘플에서 가장 인접한 k개 샘플의 class에 따라 현재 class 분류

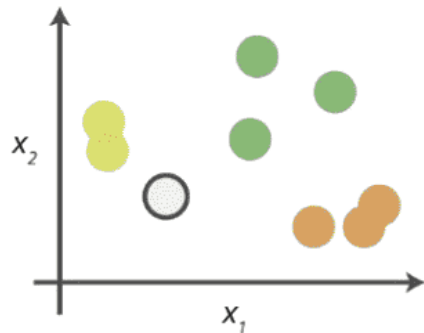


Euclidean distance (L2 distance)

Review – K Nearest Neighbors

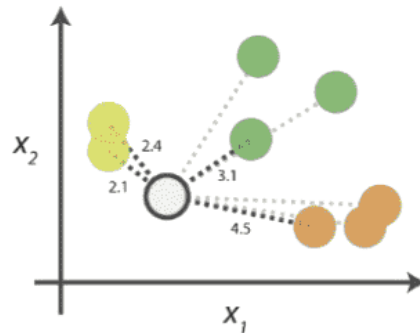
- 목적: 새로운 샘플에서 가장 인접한 k개 샘플의 class에 따라 현재 class 분류

0. Look at the data







Say you want to classify the grey point into a class. Here, there are three potential classes - lime green, green and orange.

1. Calculate distances









Start by calculating the distances between the grey point and all other points.

2. Find neighbours

Point Distance			
	2.1	→	1st NN
	2.4	→	2nd NN
	3.1	→	3rd NN
	4.5	→	4th NN

Next, find the nearest neighbours by ranking points by increasing distance. The nearest neighbours (NNs) of the grey point are the ones closest in dataspace.

3. Vote on labels

Class	# of votes	
	2	➔ Class  wins the vote! Point  is therefore predicted to be of class  .
	1	
	1	

Vote on the predicted class labels based on the classes of the k nearest neighbours. Here, the labels were predicted based on the k=3 nearest neighbours.

[실습] K Nearest Neighbors (KNN)

▪ Basecode 다운로드: LMS 강의게시판 → 14주차

▼ K Nearest Neighbors (KNN)

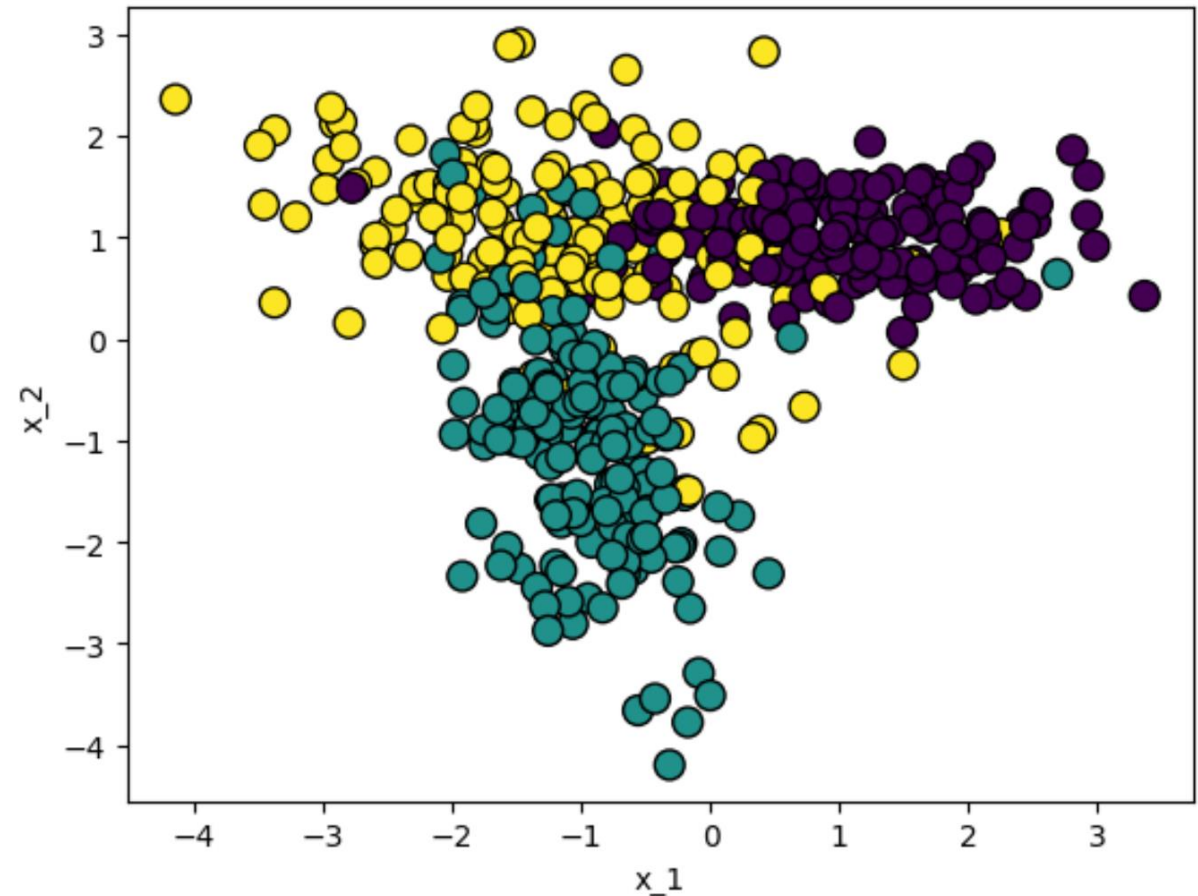
```
[14] import pandas as pd
import numpy as np
import matplotlib.pyplot as plt

from sklearn.datasets import make_classification
from sklearn.model_selection import train_test_split
```

▼ Dataset

```
[54] X, y = make_classification(n_samples=500,
                               n_features=2,
                               n_classes=3,
                               n_clusters_per_class=1,
                               n_informative=2,
                               n_redundant=0,
                               random_state=40)

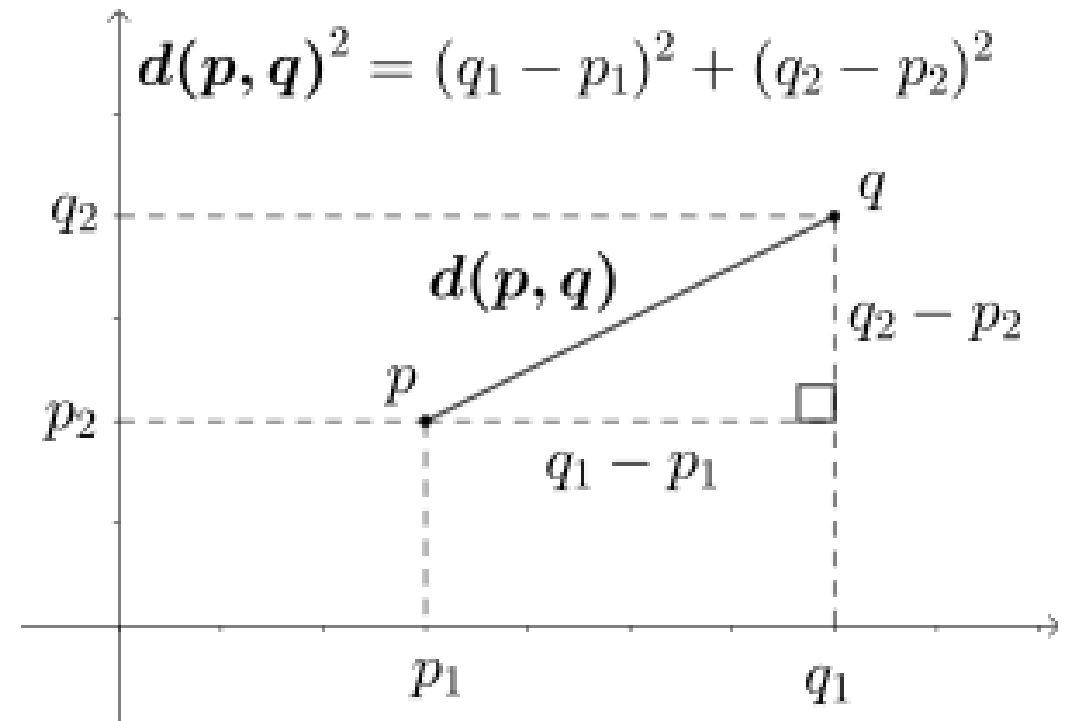
plt.scatter(X[:, 0], X[:, 1], marker='o', c=y, s=100, edgecolor="k", linewidth=1)
plt.xlabel("x_1")
plt.ylabel("x_2")
plt.show()
```



[실습] K Nearest Neighbors (KNN)

- Euclidian Distance 함수, KNN 모델 작성

```
def L2_distance(x1, x2):  
    return np.sqrt(np.sum((x1 - x2) ** 2))  
  
class KNN:  
    def __init__(self, k=3):  
        # initialization  
  
    def fit(self, X, y):  
        # Storage training datas  
  
    def predict(self, X):  
        # Prediction
```



Euclidean distance (L2 distance)

[실습] K Nearest Neighbors (KNN)

- 예측 및 성능 평가

▽ Prediction

```
[56] model = KNN()  
      model.fit(X_train, y_train)  
      y_pred = model.predict(X_test)  
  
      accuracy = np.sum(y_pred == y_test) / len(y_test)  
      print(accuracy)
```

0.81

[실습] K Nearest Neighbors (KNN)

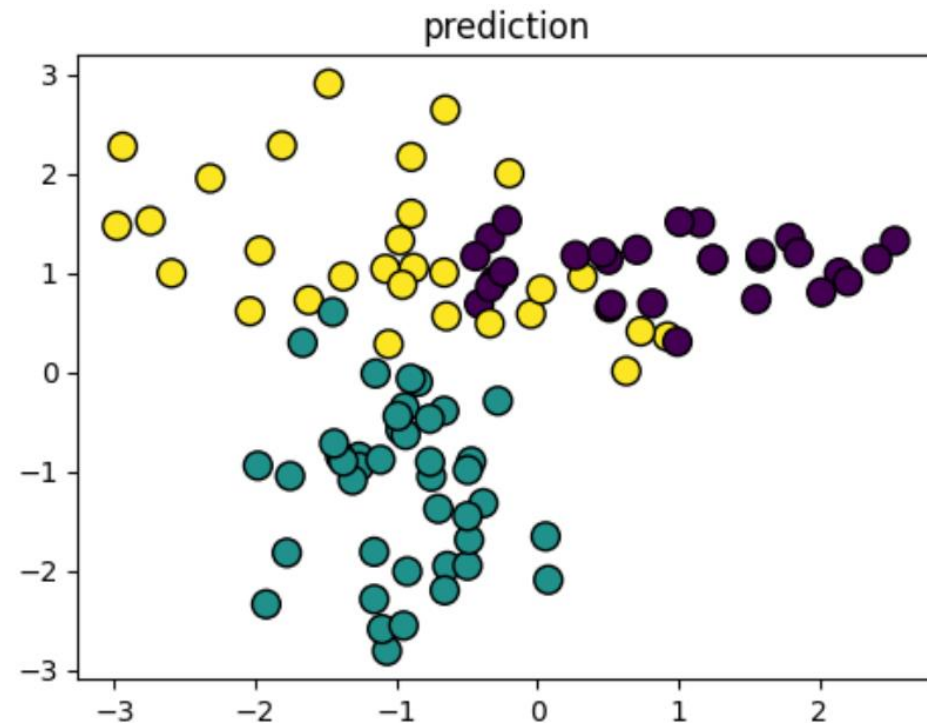
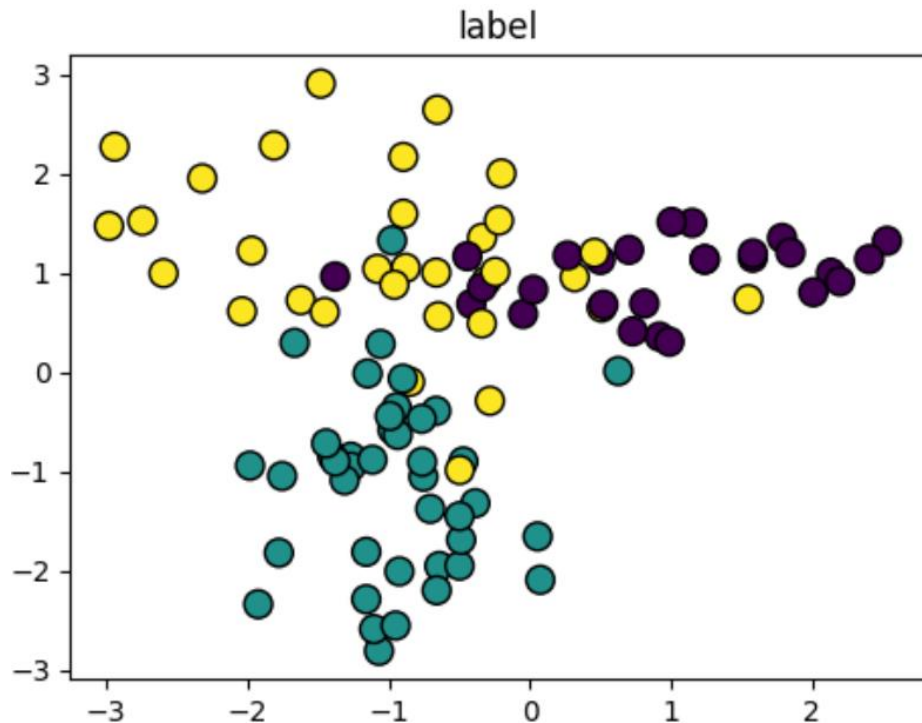
■ 예측 결과 시각화

- Label: 정답 데이터
- Prediction: 예측 값

```
[20] plt.figure(figsize=(12,6))

plt.subplot(1, 2, 1)
plt.title("label")
plt.scatter(X_test[:, 0], X_test[:, 1], marker='o', c=y_test, s=100, edgecolor="k", linewidth=1)

plt.subplot(1, 2, 2)
plt.title("prediction")
plt.scatter(X_test[:, 0], X_test[:, 1], marker='o', c=y_pred, s=100, edgecolor="k", linewidth=1)
plt.show()
```

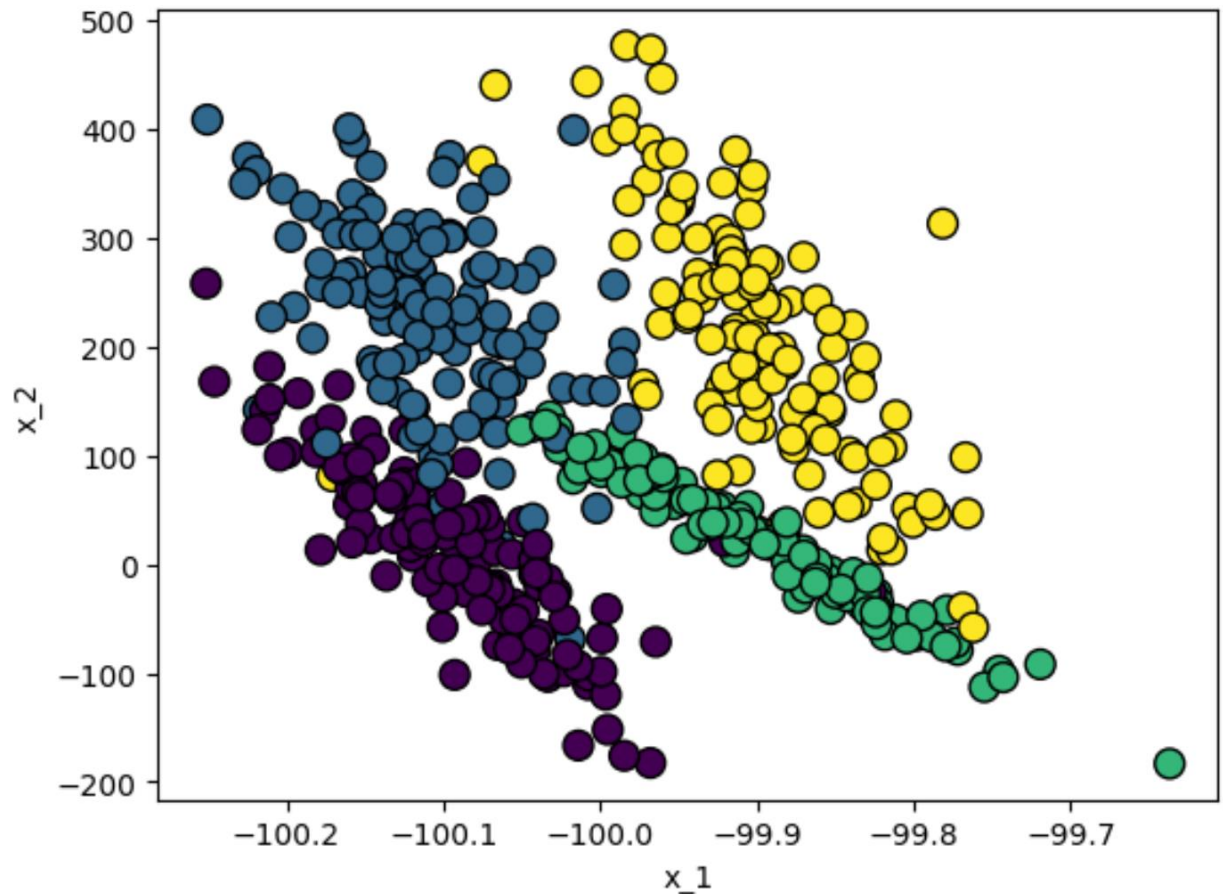


기말고사 연습 문제[1/3]

▪ K Nearest Neighbors (KNN)

- x1, x2 데이터에 대해 각각 Gaussian 정규화 수행 후 KNN 학습 및 예측 수행

```
X, y = make_classification(n_samples=500,  
                           n_features=2,  
                           n_classes=4,  
                           n_clusters_per_class=1,  
                           n_informative=2,  
                           n_redundant=0,  
                           random_state=40)  
  
X[:, 0] = X[:, 0] * 0.1 - 100  
X[:, 1] = X[:, 1] * 100 + 120  
  
plt.scatter(X[:, 0], X[:, 1], marker='o', c=y, s=100, edgecolor="k", linewidth=1)  
plt.xlabel("x_1")  
plt.ylabel("x_2")  
plt.show()
```

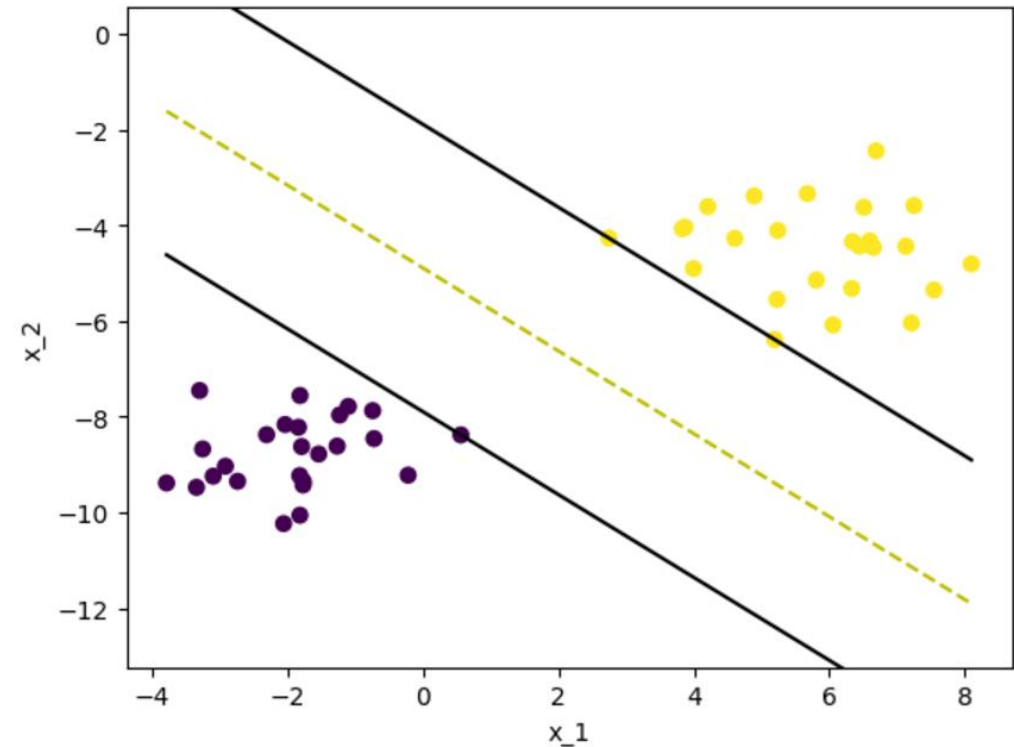
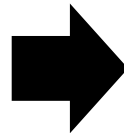
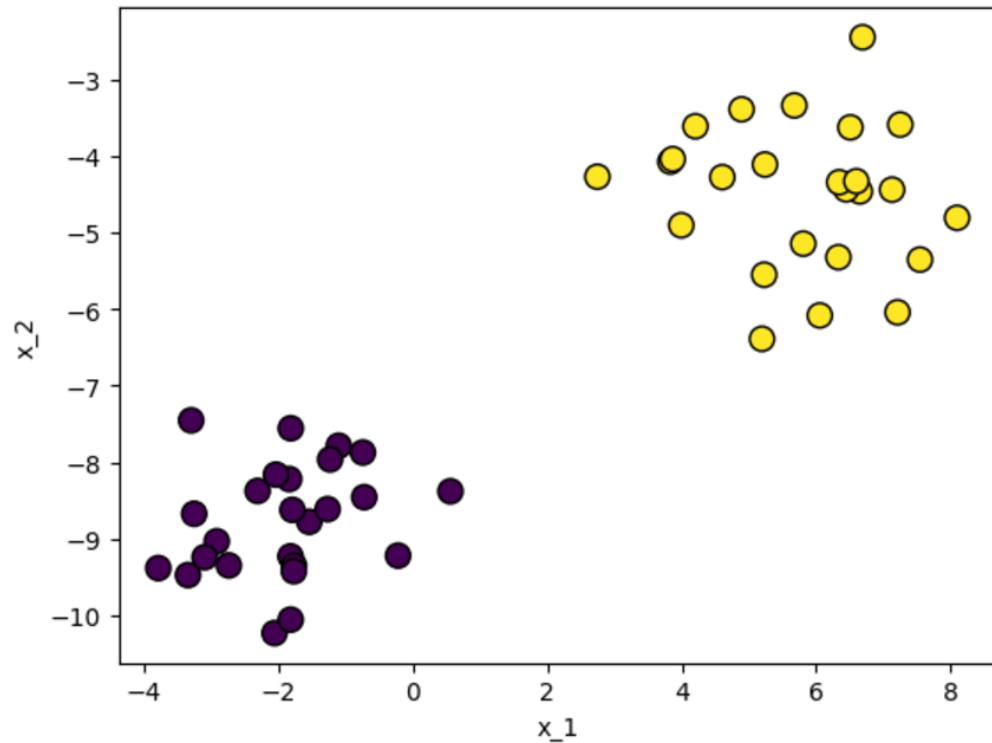


기말고사 연습 문제[2/3]

Support Vector Machine (SVM)

- Cross entropy loss를 이용해 gradient decent 수행 후 결과 시각화

```
X, y = make_blobs(n_samples=50, n_features=2, centers=2, cluster_std=1.05, random_state=40)
plt.scatter(X[:, 0], X[:, 1], marker='o', c=y, s=100, edgecolor="k", linewidth=1)
plt.xlabel("x_1")
plt.ylabel("x_2")
plt.show()
```

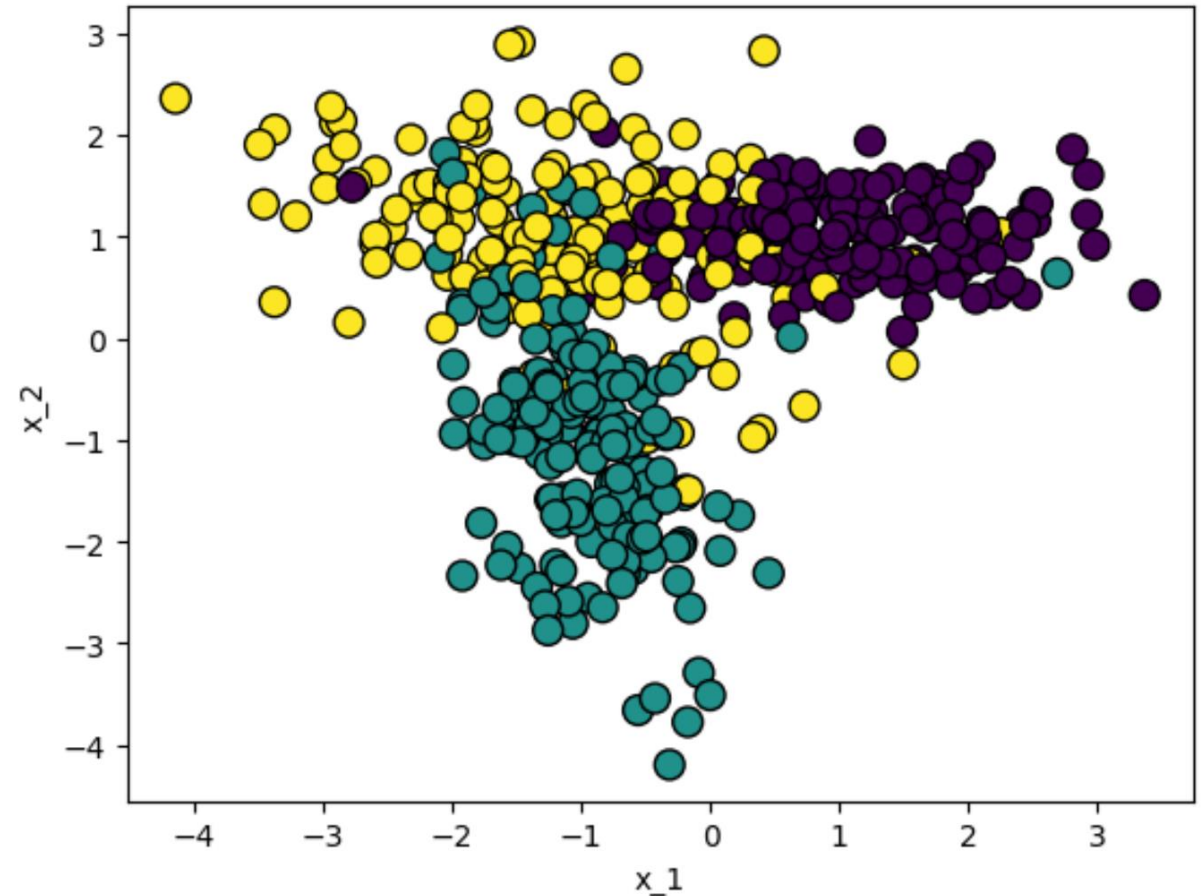


기말고사 연습 문제[3/3]

Support Vector Machine (SVM)

- 3개 클래스를 가지는 데이터에 대해 분류를 수행하는 SVM 모델 생성

```
[54] X, y = make_classification(n_samples=500,  
                                n_features=2,  
                                n_classes=3,  
                                n_clusters_per_class=1,  
                                n_informative=2,  
                                n_redundant=0,  
                                random_state=40)  
  
plt.scatter(X[:, 0], X[:, 1], marker='o', c=y, s=100, edgecolor="k", linewidth=1)  
plt.xlabel("x_1")  
plt.ylabel("x_2")  
plt.show()
```



Questions & Answers

Dongsan Jun (dsjun@dau.ac.kr)

Image Signal Processing Laboratory (www.donga-ispl.kr)

Dept. of Computer Engineering

Dong-A University, Busan, Rep. of Korea