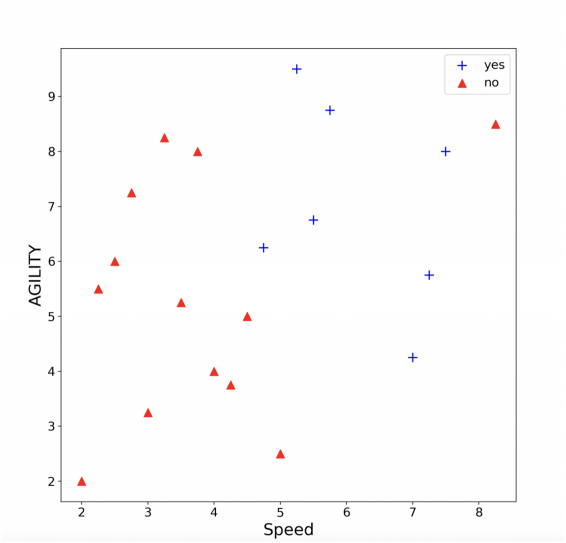
ML Day17 (Matplotlib) (K-Nearest Neighbor)

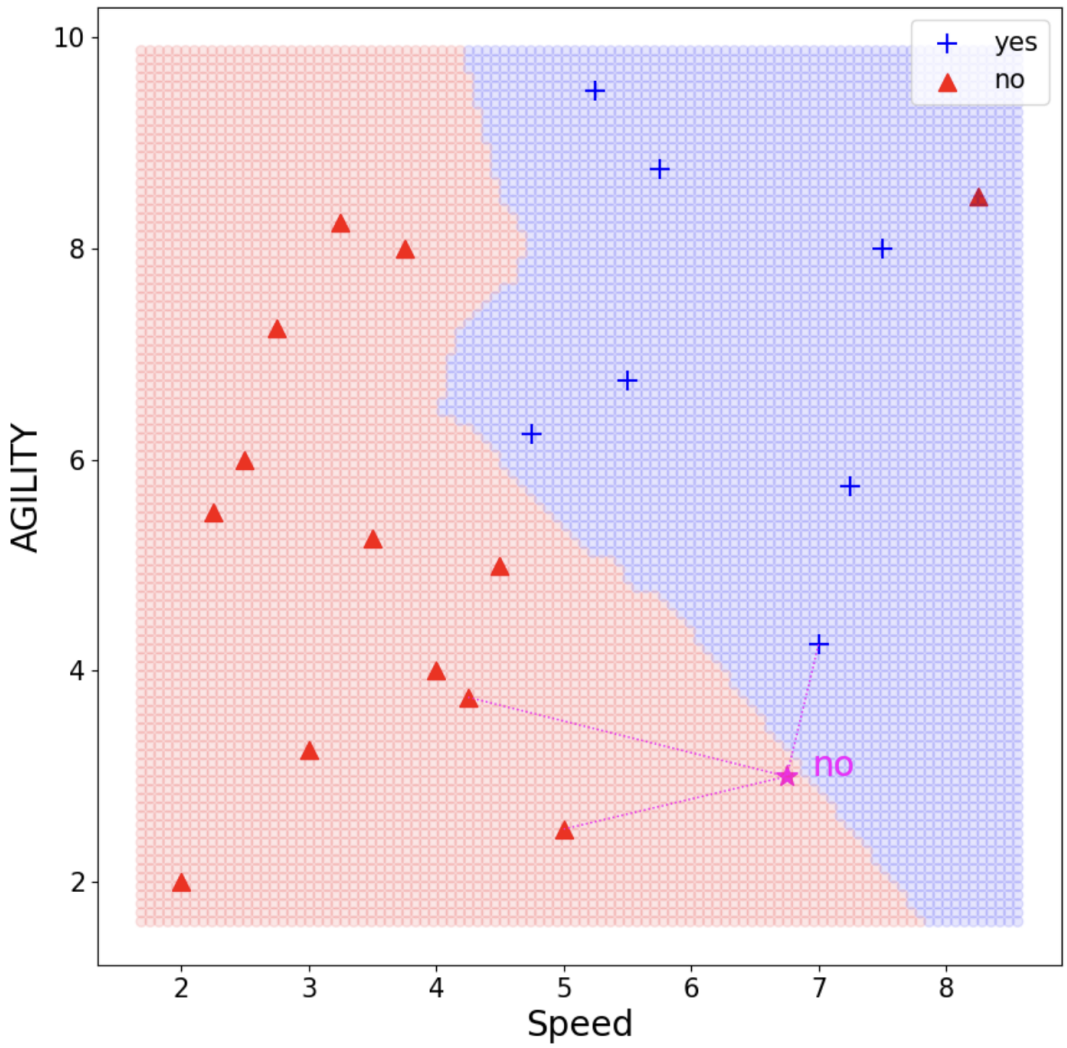
KNN Classification Fundamentals

ID	Speed	Agility	Draft	ID	Speed	Agility	Draft
1	2.50	6.00	no	11	2.00	2.00	no
2	3.75	8.00	no	12	5.00	2.50	no
3	2.25	5.50	no	13	8.25	8.50	no
4	3.25	8.25	no	14	5.75	8.75	yes
5	2.75	7.50	no	15	4.75	6.25	yes
6	4.50	5.00	no	16	5.50	6.75	yes
7	3.50	5.25	no	17	5.25	9.50	yes
8	3.00	3.25	no	18	7.00	4.25	yes
9	4.00	4.00	no	19	7.50	8.00	yes
10	4.25	3.75	no	20	7.25	5.75	yes



SBA용산 양정은 3

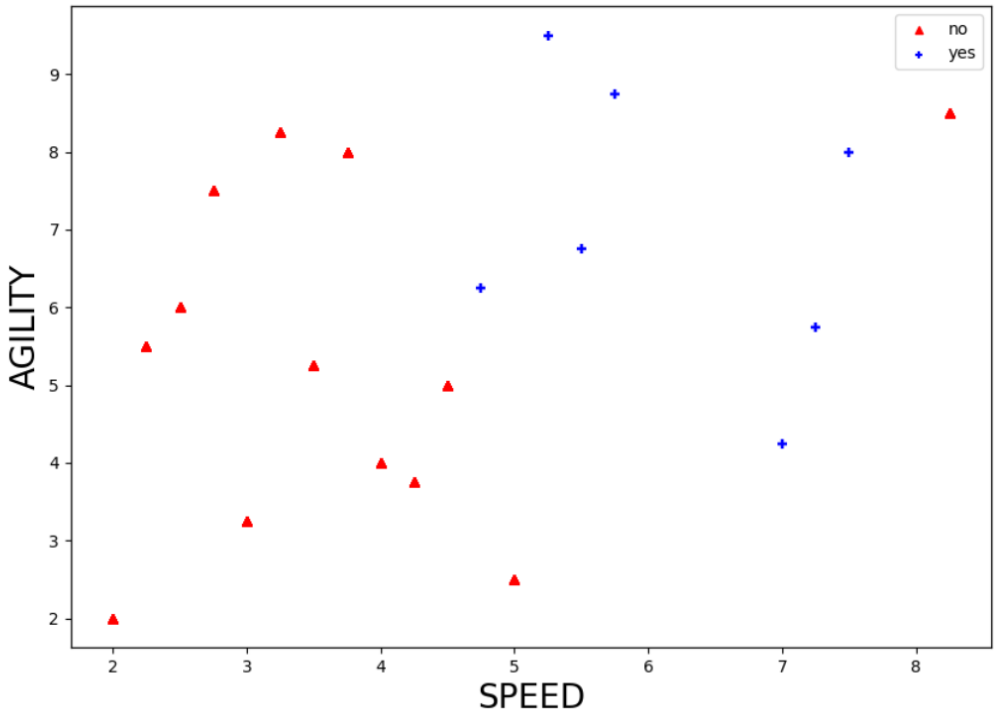
시각화 실습(3)



▼ **step (1)**

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

speed = np.array([2.50, 3.75, 2.25, 3.25, 2.75, 4.50, 3.50, 3.00, 4.00, 4.25, 2.00, 5.00, 8.25, 5.75, 4.75, 5.50, 5.25, 7.00, 7.50,
agility = np.array([6.00, 8.00, 5.50, 8.25, 7.50, 5.00, 5.25, 3.25, 4.00, 3.75, 2.00, 2.50, 8.50, 8.75, 6.25, 6.75, 9.50, 4.25, 8.(
draft = ['no', 'no', 'no', 'no', 'no', 'no', 'no', 'no', 'no', 'no', 'no', 'no', 'no', 'yes', 'yes', 'yes', 'yes', 'yes', 'yes', 'y
colors = ['red', 'blue']
marker_li = ['^', '+']
draft_name = ['no', 'yes']

fig, ax = plt.subplots(figsize=(10, 7))

for s in range(len(speed))[:13]:           # draft = 'no'인 13번째 id까지 for문으로 scatter를 그린다.
    ax.scatter(speed[:13], agility[:13],
               c=colors[0],
               s=30,
               marker=marker_li[0])
for s_1 in range(len(speed))[13:]:         # draft = 'yes'인 13 index 부터 for문으로 scatter를 그림
    ax.scatter(speed[13:], agility[13:],
               c=colors[1],
               s=30,
               marker=marker_li[1])

ax.set_xlabel('SPEED',
              fontsize=20)
```

```
ax.set_ylabel('AGILITY',
              fontsize=20)

ax2 = ax.twinx()

for i in range(2):
    ax2.scatter([], [],
                s=20,
                c=colors[i],
                marker=marker_li[i],
                label=draft_name[i])
ax2.legend(loc='upper right',
           bbox_to_anchor=(1, 1),
           fontsize=10,
           ncol=1)

ax2.tick_params(axis='y',
                right=False, labelright=False)

plt.show()
```



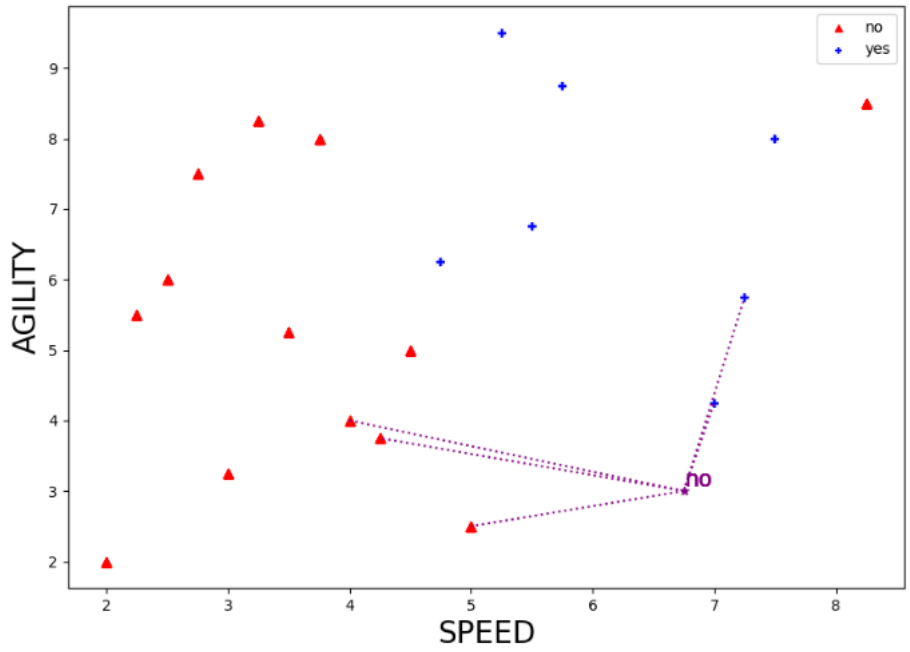

```
# 선생님 code
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt

athletes = [[2.50, 6.00, 'no'],
            [3.75, 8.00, 'no'],
            [2.25, 5.50, 'no'],
            [3.25, 8.25, 'no'],
            [2.75, 7.25, 'no'],
            [4.50, 5.00, 'no'],
            [3.50, 5.25, 'no'],
            [3.00, 3.25, 'no'],
            [4.00, 4.00, 'no'],
            [4.25, 3.75, 'no'],
            [2.00, 2.00, 'no'],
            [5.00, 2.50, 'no'],
            [8.25, 8.50, 'no'],
            [5.75, 8.75, 'yes'],
            [4.75, 6.25, 'yes'],
            [5.50, 6.75, 'yes'],
            [5.25, 9.50, 'yes'],
            [7.00, 4.25, 'yes'],
```

```
            [7.50, 8.00, 'yes'],
            [7.25, 5.75, 'yes']]

athletes_df = pd.DataFrame(athletes,
                           columns=['SPEED', 'AGILITY', 'DRAFT'])


X = athletes_df[['SPEED', 'AGILITY']].to_numpy()   # to_numpy : series, dataframe 형태의 데이터를 numpy 배열 객체인 ndarray로 반환하 메서드
y = athletes_df['DRAFT'].values                    # values : 배열을 반환하는 메서드 (dtype에 따라 ndarray, ndarray-like를 반환한다.

fig, ax = plt.subplots(figsize=(10, 10))

X_pos, X_neg = X[y == 'yes'], X[y == 'no']         # boolean indexing

ax.scatter(X_pos[:, 0], X_pos[:, 1], color='blue',
           marker='+', s=130, label='yes')
ax.scatter(X_neg[:, 0], X_neg[:, 1], color='red',
           marker='^', s=100, label='no')
ax.set_xlabel("Speed", fontsize=20)
ax.set_ylabel("AGILITY", fontsize=20)
ax.legend(fontsize=15)
ax.tick_params(labelsize=15)
```

▼ step (2)

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

speed = np.array([2.50, 3.75, 2.25, 3.25, 2.75, 4.50, 3.50, 3.00, 4.00, 4.25, 2.00, 5.00, 8.25, 5.75, 4.75, 5.50, 5.25, 7.00, 7.50,
agility = np.array([6.00, 8.00, 5.50, 8.25, 7.50, 5.00, 5.25, 3.25, 4.00, 3.75, 2.00, 2.50, 8.50, 8.75, 6.25, 6.75, 9.50, 4.25, 8.(
draft = ['no', 'no', 'no', 'no', 'no', 'no', 'no', 'no', 'no', 'no', 'no', 'no', 'no', 'yes', 'yes', 'yes', 'yes', 'yes', 'yes', 'y
colors = ['red', 'blue']
marker_li = ['^', '+']
test_data = [6.75, 3.00]
draft_name = ['no', 'yes']
fig, ax = plt.subplots(figsize=(10, 7))

# main scatter
for s in range(len(speed))[:13]:
    ax.scatter(speed[:13], agility[:13],
               color=colors[0],
               s=35,
               marker=marker_li[0])
for s_1 in range(len(speed))[13:]:
    ax.scatter(speed[13:], agility[13:],
               color=colors[1],
               s=35,
               marker=marker_li[1])

ax.set_xlabel('SPEED',
              fontsize=20)
ax.set_ylabel('AGILITY',
              fontsize=20)



e_distance_li = []
e_distance = 0
diff = 0

# test_data와 각 선수들의 speed,agility 간의 Euclidean distance / e-distance와 그에 해당하는 speed, agility 값을 list에 append
for s_idx in range(len(speed)):
    diff = (test_data[0] - speed[s_idx])**2 + (test_data[1] - agility[s_idx])**2
    e_distance = np.sqrt(diff)
    e_distance_li.append([e_distance, speed[s_idx], agility[s_idx], draft[s_idx]])

# 위에서 저장된 list를 e_distance 값을 기준으로 한 후 오름차순으로 정렬
s_dist_diff = sorted(e_distance_li, key=lambda e_distance_li: e_distance_li[0])

K = 5

# s_dist_diff[:K] 만큼 for문으로 plot을 그린다.(x = [test_data[0], s_dist_diff[x][1]], y = [test_data[1], s_dist_diff[x][2]])
for x in range(len(s_dist_diff))[:K]:
    ax.plot([test_data[0], s_dist_diff[x][1]], [test_data[1], s_dist_diff[x][2]],
            color='purple',
            linestyle=':')

# s_dist_diff[:K] 만큼 for문으로 [:K]까지 s_dist_diff[4]에 해당하는 draft값('yes', 'no')의 합산을 해준다.
for y in range(len(s_dist_diff))[:K]:
    yes_sum = 0
```

```
    no_sum = 0
    if s_dist_diff[y][3] == 'yes':
        yes_sum += 1
    else:
        no_sum += 1

# 위 code에서 구한 'yes'의 수와 'no'의 수 합산값을 비교하여 더 큰 draft값을 ax.test로출력
for _ in range(K):
    if yes_sum > no_sum:
        ax.text(x=test_data[0], y=test_data[1],
                va='bottom',
                ha='left',
                s='yes',
                c='purple',
                fontsize=15)
    else:
        ax.text(x=test_data[0], y=test_data[1],
                va='bottom',
                ha='left',
                s='no',
                c='purple',
                fontsize=15)

# test_data 표현
ax.scatter(test_data[0], test_data[1],
           c='purple',
           s=30,
           marker='*')


ax2 = ax.twinx()

# 비어있는 scatter 생성
for i in range(2):
    ax2.scatter([], [],
                s=20,
                c=colors[i],
                marker=marker_li[i],
                label=draft_name[i])
ax2.legend(loc='upper right',
           bbox_to_anchor=(1, 1),
           fontsize=10,
           ncol=1)
# 빈 scatter에 해당하는 y축 ticks, tickslabel 제거
ax2.tick_params(axis='y',
                right=False, labelright=False)

# xx, yy = np.meshgrid(speed, agility)

#for mesh_idx in range(2):
#    plt.scatter(xx, yy, marker='*', color=colors[mesh_idx], alpha=0.1)


plt.show()
```



▼ step (3)

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

speed = np.array([2.50, 3.75, 2.25, 3.25, 2.75, 4.50, 3.50, 3.00, 4.00, 4.25, 2.00, 5.00, 8.25, 5.75, 4.75, 5.50, 5.25, 7.00, 7.50,
agility = np.array([6.00, 8.00, 5.50, 8.25, 7.50, 5.00, 5.25, 3.25, 4.00, 3.75, 2.00, 2.50, 8.50, 8.75, 6.25, 6.75, 9.50, 4.25, 8.(
draft = ['no', 'no', 'no', 'no', 'no', 'no', 'no', 'no', 'no', 'no', 'no', 'no', 'no', 'yes', 'yes', 'yes', 'yes', 'yes', 'yes', 'y
colors = ['red', 'blue']
marker_li = ['^', '+']
test_data = [6.75, 3.00]
draft_name = ['no', 'yes']
fig, ax = plt.subplots(figsize=(10, 7))

# main scatter
for s in range(len(speed))[:13]:
    ax.scatter(speed[:13], agility[:13],
               color=colors[0],
               s=35,
               marker=marker_li[0])
for s_1 in range(len(speed))[13:]:
    ax.scatter(speed[13:], agility[13:],
               color=colors[1],
               s=35,
               marker=marker_li[1])

ax.set_xlabel('SPEED',
              fontsize=20)
ax.set_ylabel('AGILITY',
              fontsize=20)

e_distance_li = []
e_distance = 0
diff = 0

# test_data와 각 선수들의 speed,agility 간의 Euclidean distance / e-distance와 그에 해당하는 speed, agility 값을 list에 append
for s_idx in range(len(speed)):
    diff = (test_data[0] - speed[s_idx])**2 + (test_data[1] - agility[s_idx])**2
    e_distance = np.sqrt(diff)
    e_distance_li.append([e_distance, speed[s_idx], agility[s_idx], draft[s_idx]])

# 위에서 저장된 list를 e_distance 값을 기준으로 한 후 오름차순으로 정렬
s_dist_diff = sorted(e_distance_li, key=lambda e_distance_li: e_distance_li[0])

K = 3
```

```
# s_dist_diff[:K] 만큼 for문으로 plot을 그린다.(x = [test_data[0], s_dist_diff[x][1]], y = [test_data[1], s_dist_diff[x][2]])
for x in range(len(s_dist_diff))[:K]:
    ax.plot([test_data[0], s_dist_diff[x][1]], [test_data[1], s_dist_diff[x][2]],
            color='purple',
            linestyle=':')
# meshgrid
X = np.linspace(1, 10, 100)
Y = np.linspace(1, 10, 100)
X, Y = np.meshgrid(X, Y)

m_e_distance_li = []
m_e_distance = 0
m_diff = 0
####
for X_idx in range(len(X)):
    for Y_idx in range(len(Y)):
        m_diff = (X[X_idx][Y_idx] - speed[X_idx])**2 + (X[X_idx][Y_idx] - agility[X_idx])**2
        m_e_distance = np.sqrt(m_diff)
        m_e_distance_li.append([m_e_distance, speed[X_idx], agility[X_idx], draft[X_idx]])

m_dist_diff = sorted(m_e_distance_li, key=lambda m_e_distance_li: m_e_distance_li[0])

for h in range(len(m_dist_diff))[:K]:
    yes_sum_g = 0
    no_sum_g = 0
    if s_dist_diff[h][3] == 'yes':
        yes_sum_g += 1
    else:
        no_sum_g += 1
    for g in range(len(X)):
        if yes_sum_g > no_sum_g:
            ax.plot(X[g][0], Y[g][0],
                    c='blue',
                    linestyle=':',
                    alpha=0.3)
        else:
            ax.plot(X[g][0], Y[g][0],
                    c='red',
                    linestyle=':',
                    alpha=0.3)
####
# s_dist_diff[:K] 만큼 for문으로 [:K]까지 s_dist_diff[4]에 해당하는 draft값('yes', 'no')의 합산을 해준다.
for y in range(len(s_dist_diff))[:K]:
    yes_sum = 0
    no_sum = 0
    if s_dist_diff[y][3] == 'yes':
        yes_sum += 1
    else:
        no_sum += 1

# 위 code에서 구한 'yes'의 수와 'no'의 수 합산값을 비교하여 더 큰 draft값을 ax.test로출력
for _ in range(K):
    if yes_sum > no_sum:
        ax.text(x=test_data[0], y=test_data[1],
                va='bottom',
                ha='left',
                s='yes',
                c='purple',
                fontsize=15)
    else:
        ax.text(x=test_data[0], y=test_data[1],
                va='bottom',
                ha='left',
                s='no',
                c='purple',
                fontsize=15)

# test_data 표현
ax.scatter(test_data[0], test_data[1],
           c='purple',
           s=30,
           marker='*')


ax2 = ax.twinx()

# 비어있는 scatter 생성
for i in range(2):
    ax2.scatter([], [],
                s=20,
                c=colors[i],
                marker=marker_li[i],
                label=draft_name[i])
ax2.legend(loc='upper right',
           bbox_to_anchor=(1, 1),
           fontsize=10,
```

```
           ncol=1)

# 빈 scatter에 해당하는 y축 ticks, tickslabel 제거
ax2.tick_params(axis='y',
                right=False, labelright=False)

plt.show()
```