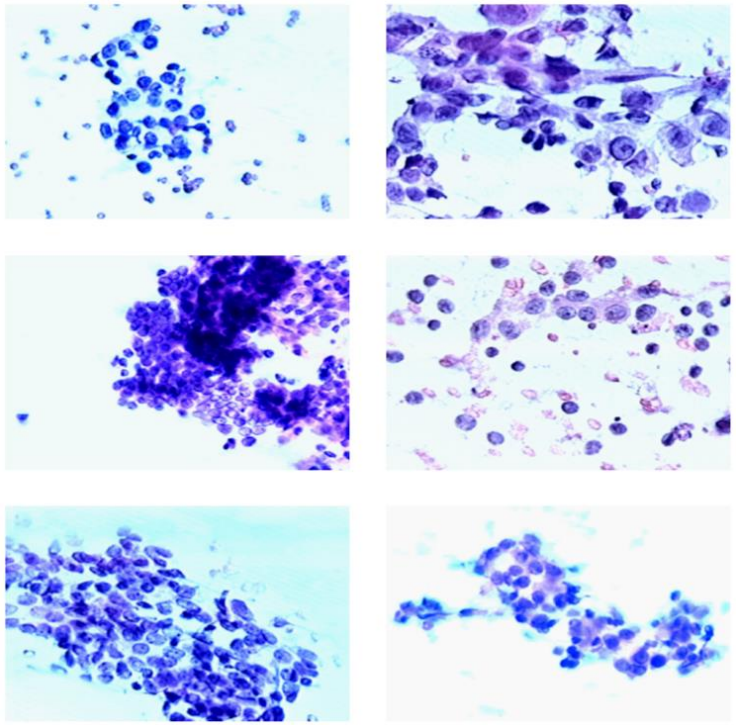
Data Mining

k-최근접 이웃 과제

2021년 3월 25일

위스콘신 유방암 진단 데이터셋 (Wisconsin Breast Cancer Diagnostic dataset)

양성 (benign) **악성** (malignant)



위스콘신 유방암 진단 데이터셋 (WBCD)

- 위스콘신 대학(University of Wisconsin)의 연구원들이 기부
- 유방암 조직 검사 **569개** 샘플
- 총 **32개 컬럼**으로 구성됨 (ID, 진단 결과, 30 실측값)

- **진단 결과** " M" : 악성 (malignant), " B" : 양성 (benign)
- **30개 실측값**
  - 유방 종양의 미세침 흡인물 이미지에서 측정한 세포핵의 특징
  - 세포핵의 10개 특징에 대한 평균, 표준 오차, 최악의 값(즉, 최댓값)

https://archive.ics.uci.edu/ml/datasets/Breast+Cancer+Wisconsin+(Diagnostic)

위스콘신 유방암 진단 데이터셋 (Wisconsin Breast Cancer Diagnostic dataset)

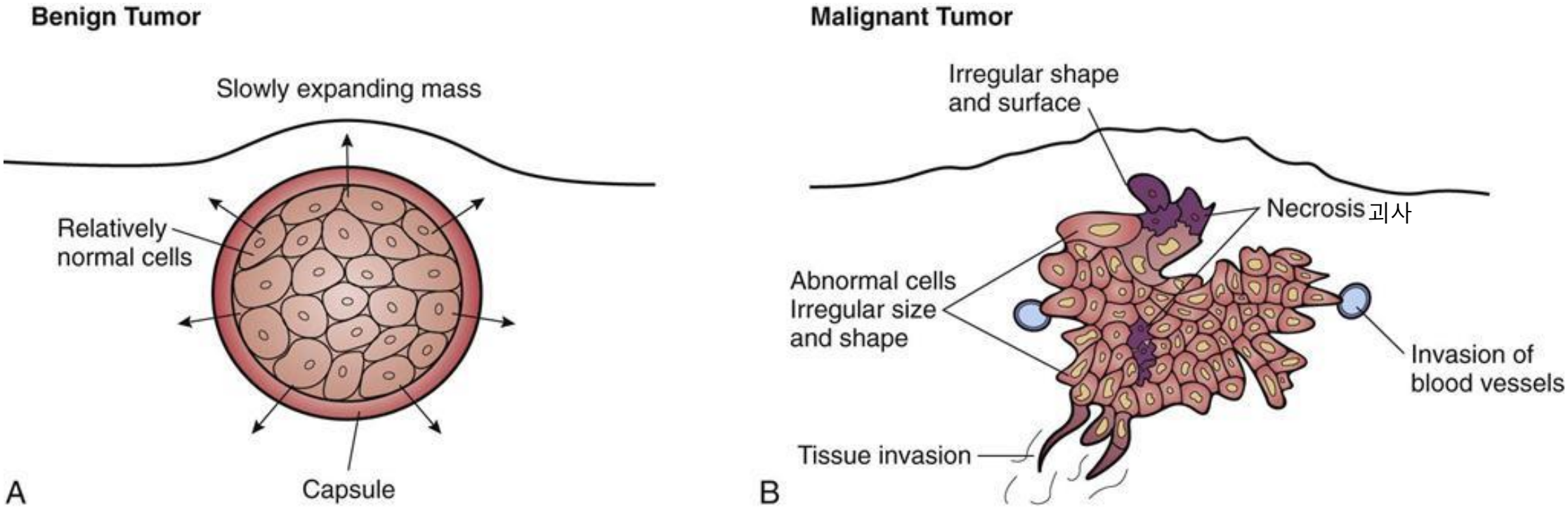
총 32개 컬럼

속성	설명	타입
id	아이디	int
diagnosis	M, B	char
_mean	평균 (3-12 컬럼)	float
_se	표준 오차 (13-22 컬럼)	float
_worst	최댓값 (23-32 컬럼)	float

10개 특징 별 (평균, 표준 오차, 최댓값)

속성	설명	타입
Radius	반지름	float
Texture	질감 (Gray-Scale 값의 표준 편차)	float
Perimeter	둘레	float
Area	넓이	float
Smoothness	매끄러움 (반지름의 변화율)	float
Compactness	조밀성 ($Perimeter^2/Area-1$)	float
Concavity	오목함	float
Concave points	오목한 점의 수	float
Symmetry	대칭성	float
Fractal dimension	프랙탈 차원	float

악성 종양 vs. 양성 종양



https://www.macadamian.com/learn/a-practical-application-of-machine-learning-in-medicine/

데이터셋 다운로드

데이터 다운로드

```
import requests

data = requests.get("https://archive.ics.uci.edu/ml/machine-learning-
databases/breast-cancer-wisconsin/wdbc.data")
dataset_path = os.path.join('data', 'wdbc.data')

with open(dataset_path, "w") as f:
  f.write(data.text)
```

- URL에서 데이터를 다운로드해서 'wdbc.data' 파일에 저장

csv 파일 읽기

패키지 임포트

```
import matplotlib.pyplot as plt
import os
from typing import Dict
import csv
from collections import defaultdict
```

csv 파일 읽기 및 한 행 씩 파싱

```
with open(dataset_path) as f:
    reader = csv.reader(f)
    cancer_data = [parse_cancer_row(row) for row in reader if row]
```

- csv 파일 읽기
- 각 row를 파싱해서 데이터들을 LabledPoint 리스트로 구성

데이터 파싱 (Q)

데이터 파싱

```
def parse_cancer_row(row: List[str]) -> LabeledPoint:
    # your code
    return LabeledPoint(measurements, label)
```

컬럼 이름

컬럼 이름

```
columns = [
    "radius_mean", "texture_mean", "perimeter_mean", "area_mean", "smoothness_mean",
    "compactness_mean", "concavity_mean", "points_mean", "symmetry_mean", "dimension_mean",
    "radius_se", "texture_se", "perimeter_se", "area_se", "smoothness_se",
    "compactness_se", "concavity_se", "points_se", "symmetry_se", "dimension_se",
    "radius_worst", "texture_worst", "perimeter_worst", "area_worst", "smoothness_worst",
    "compactness_worst", "concavity_worst", "points_worst", "symmetry_worst", "dimension_worst",
    ]
```

행렬 생성

데이터 탐색 단계의 시각화를 위해 임시로 데이터 행렬 생성

LabledPoint 리스트에서 행렬을 생성하는 함수

```
from scratch.linear_algebra import get_column, shape

def make_matrix(dataset):
    matrix = []
    for datapoint in dataset:
        matrix.append(datapoint.point)
    return matrix
```

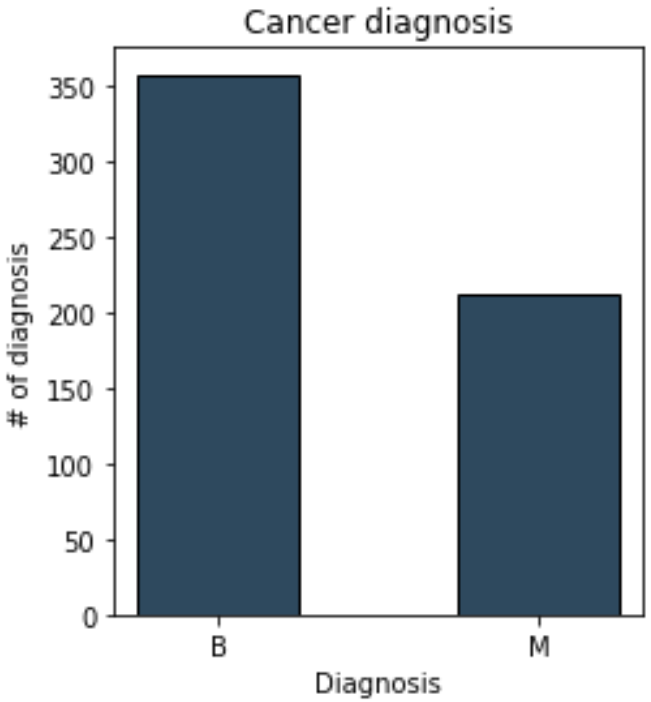
- LabledPoint의 point를 모아서 리스트로 만들면 행렬이 됨

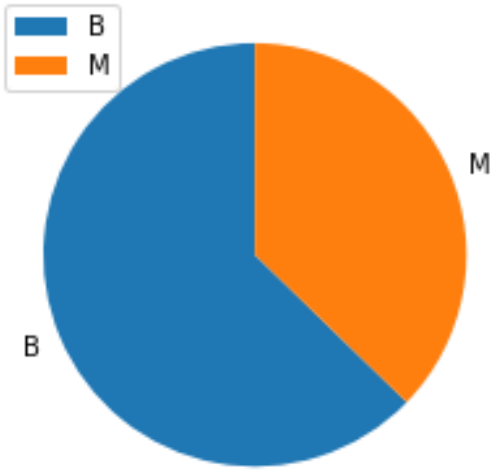
행렬 생성

```
cancer_matrix = make_matrix(cancer_data)
print(shape(cancer_matrix))
```

(569, 30)

데이터 탐색 클래스 비율 확인





레이블 개수 세기

```
label_type = defaultdict(int)
for cancer in cancer_data:
    label_type[cancer.label] += 1
```

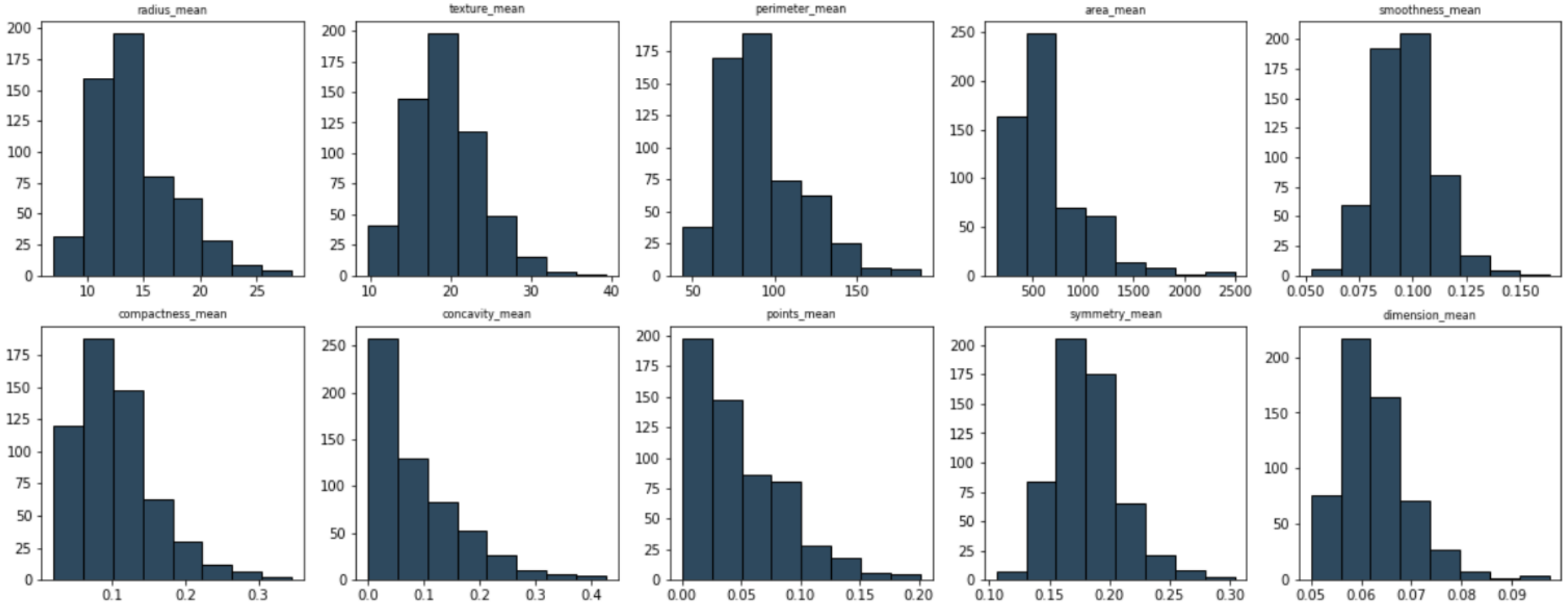
데이터 탐색 클래스 비율 확인

막대 그래프와 파이 차트 그리기

```
plt.figure(figsize=(8,4))
plt.subplot(1, 2, 1)
plt.bar(label_type.keys(),
        label_type.values(),
        0.5,
        facecolor="#2E495E",
        edgecolor=(0, 0, 0))                # Black edges for each bar

plt.xlabel("Diagnosis")
plt.ylabel("# of diagnosis")
plt.title("Cancer diagnosis")

plt.subplot(1, 2, 2)
pies = plt.pie(label_type.values(),
               labels=label_type.keys(),
               startangle=90)
plt.legend()
plt.show()
```

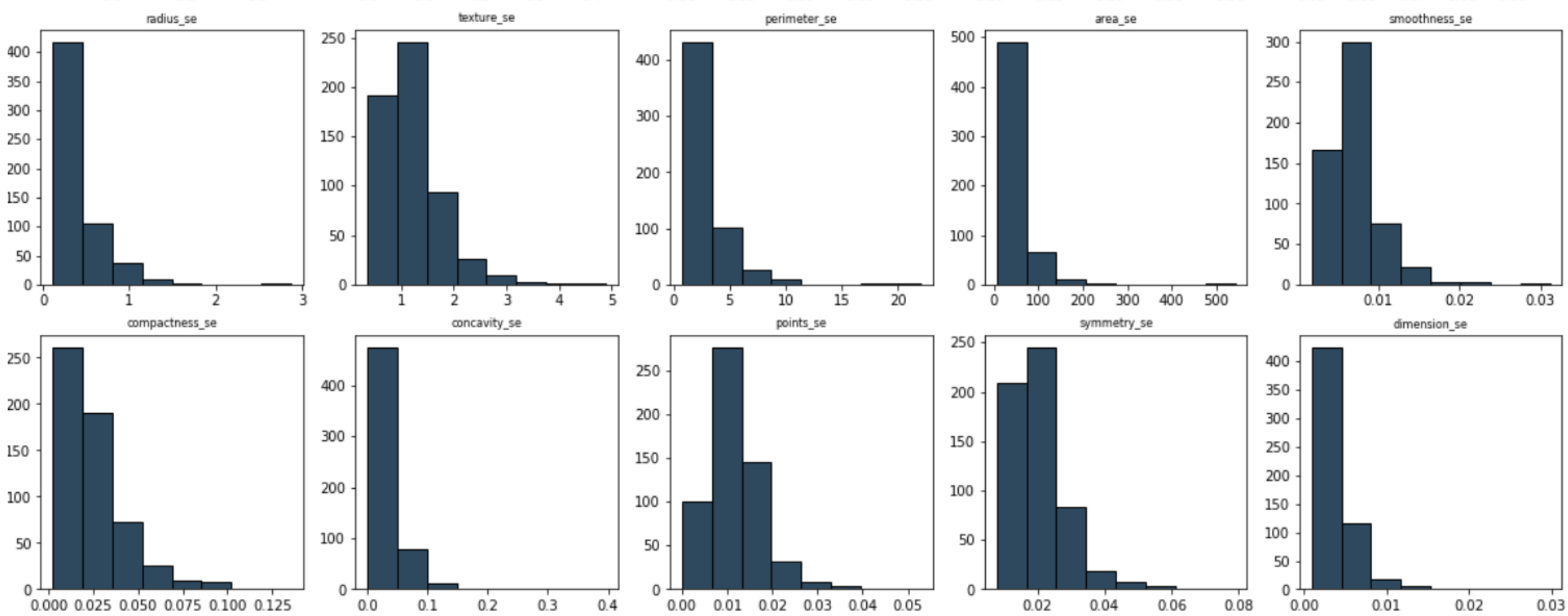
데이터 탐색 특징 별 히스토그램

평균



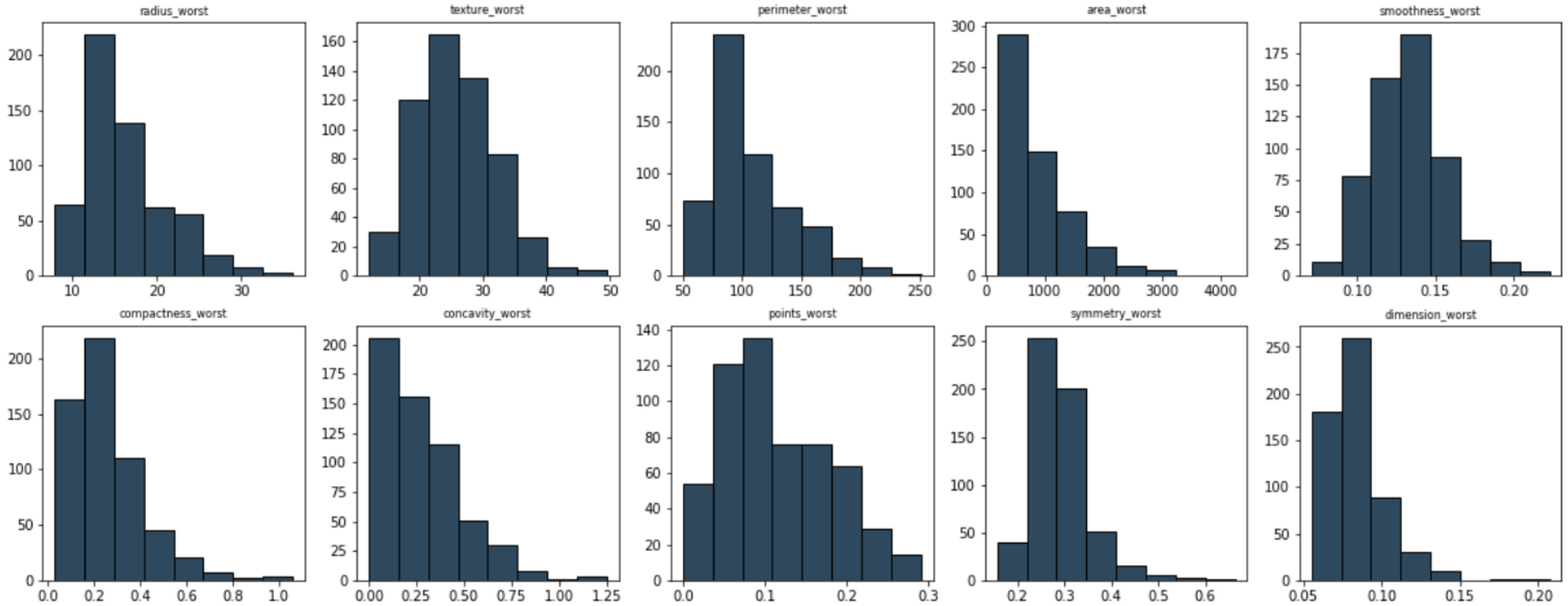
데이터 탐색 특징 별 히스토그램

표준 오차



데이터 탐색 특징 별 히스토그램

최댓값



데이터 탐색 특징 별 히스토그램

특징 별로 히스토그램 그리기

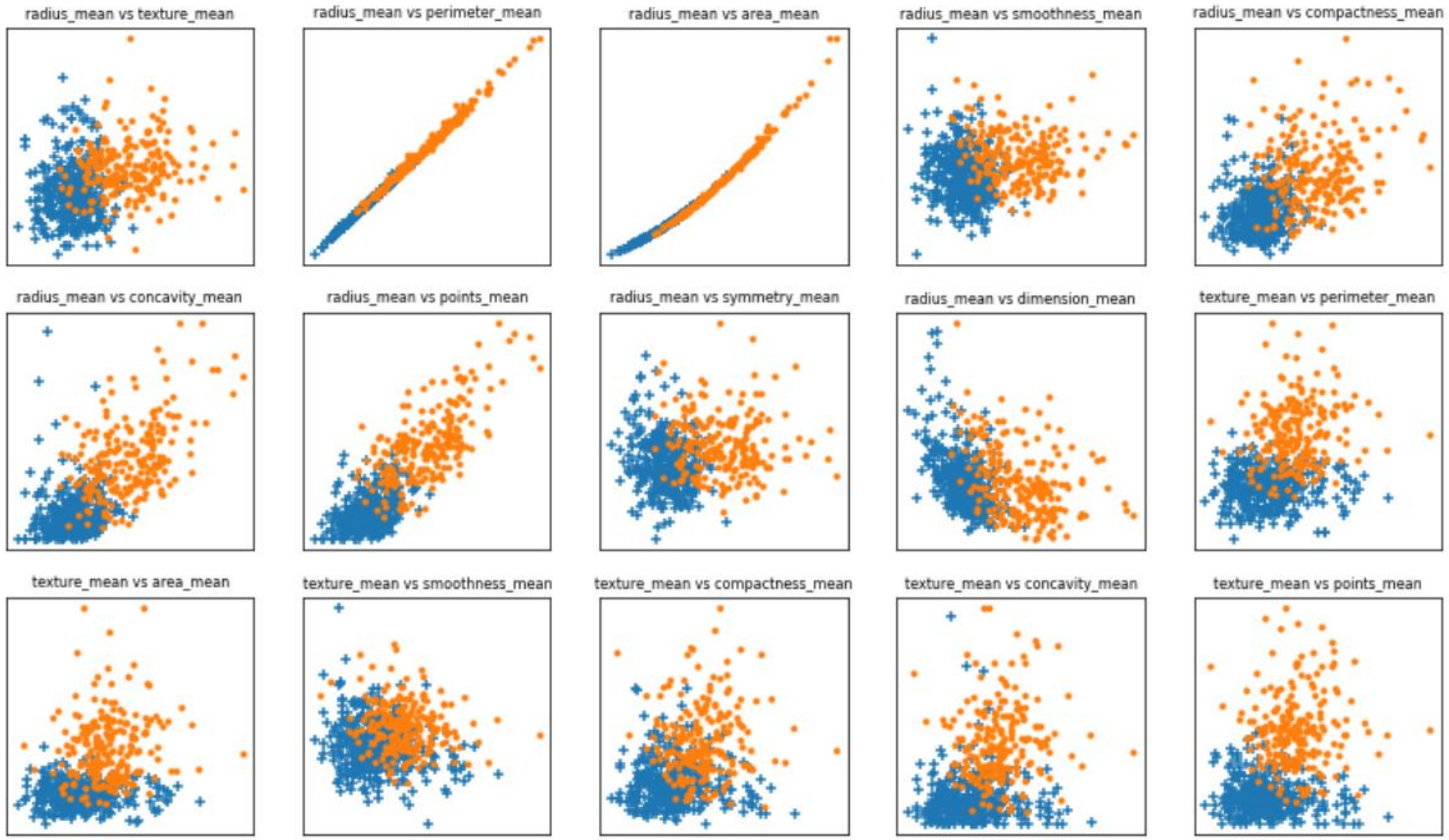
```
from matplotlib import pyplot as plt
num_rows = 6
num_cols = 5

fig, ax = plt.subplots(num_rows, num_cols, figsize=(num_cols*4, num_rows*4))
for row in range(num_rows):
    for col in range(num_cols):
        histogram(ax[row][col], num_cols * row + col)
plt.show()
```

특정 컬럼의 특징을 히스토그램으로 그리는 함수

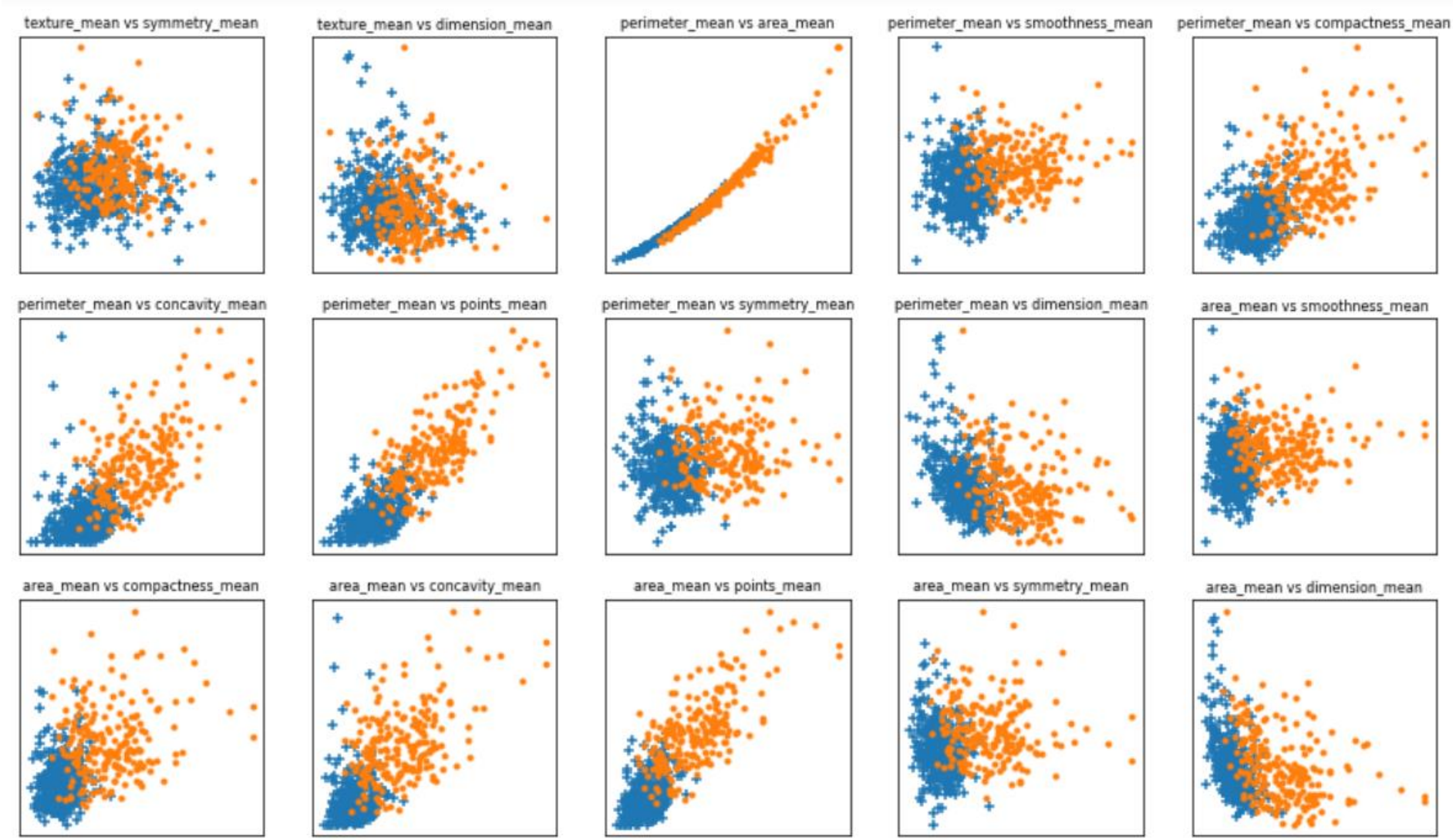
```
def histogram(ax, col : int):

    n, bins, patches = ax.hist(get_column(cancer_matrix, col),
                               8,
                               facecolor="#2E495E",
                               edgecolor=(0, 0, 0))

    ax.set_title(columns[col], fontsize=8)
```

데이터 탐색 특징 쌍 별 산포도

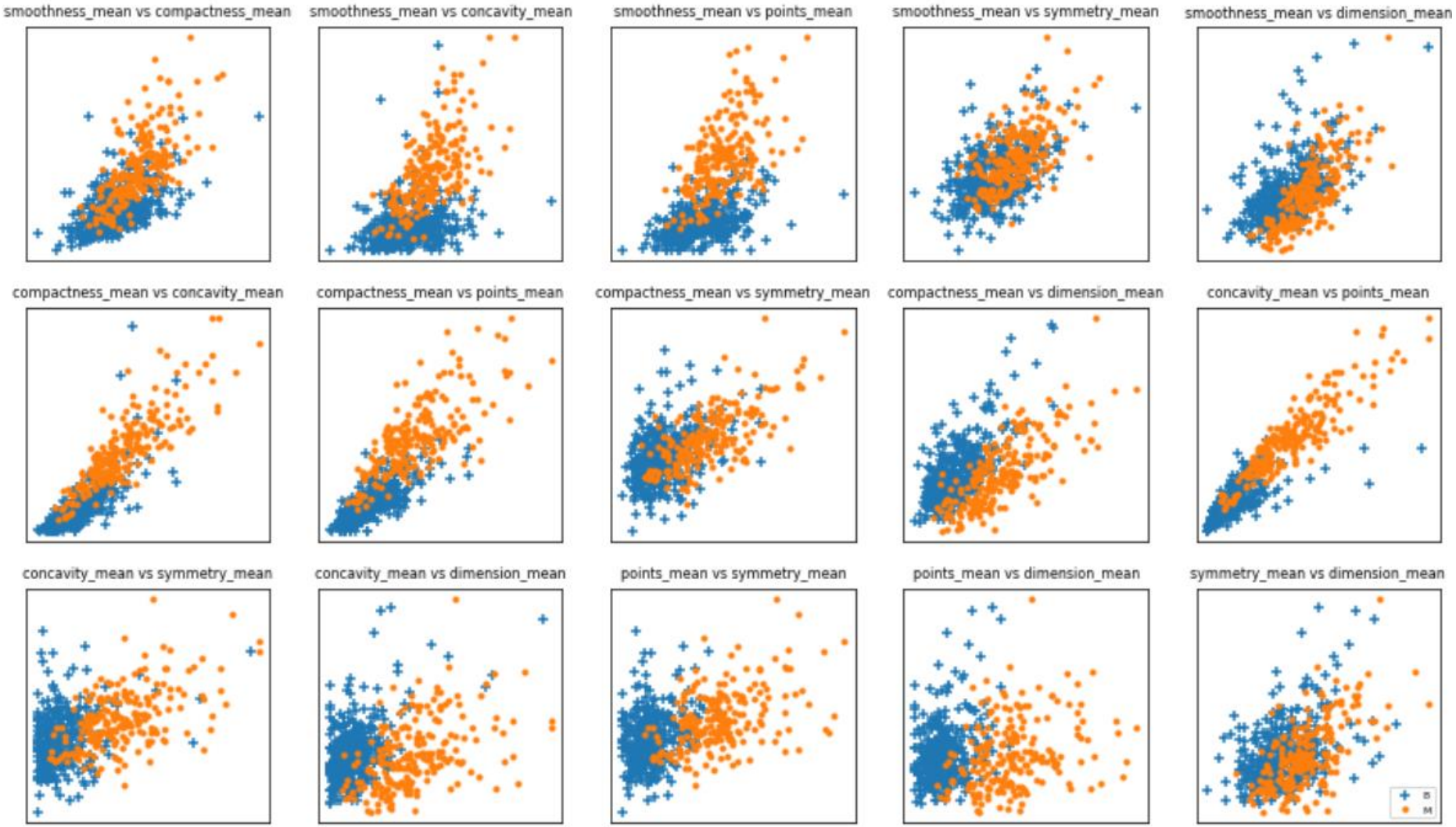
평균 관련 특징만 비교



데이터 탐색 특징 쌍 별 산포도



데이터 탐색 특징 쌍 별 산포도



데이터 탐색 특징 쌍 별 산포도

같은 레이블끼리 딕셔너리에 모으기

```
points_by_diagnosis: Dict[str, List[Vector]] = defaultdict(list)
for cancer in cancer_data:
    points_by_diagnosis[cancer.label].append(cancer.point)
```

- points_by_diagnosis 딕셔너리에 같은 레이블 별로 데이터 벡터를 리스트 형태로 모으기

평균 관련 특징의 쌍 만들기

```
start = 0
end = start + 10
pairs = [(i, j) for i in range(start, end) for j in range(i+1, end) if i < j]
marks = ['+', '.']
```

pairs = [(0, 1), (0, 2), (0, 3), (0, 4), (0, 5), (0, 6), (0, 7), (0, 8), (0, 9), (1, 2), (1, 3), (1, 4), (1, 5), (1, 6), (1, 7), (1, 8), (1, 9), (2, 3), (2, 4), (2, 5), (2, 6), (2, 7), (2, 8), (2, 9), (3, 4), (3, 5), (3, 6), (3, 7), (3, 8), (3, 9), (4, 5), (4, 6), (4, 7), (4, 8), (4, 9), (5, 6), (5, 7), (5, 8), (5, 9), (6, 7), (6, 8), (6, 9), (7, 8), (7, 9), (8, 9)]

데이터 탐색 특징 쌍 별 산포도

9x5 그리드에 그림 그리기

```
from matplotlib import pyplot as plt
num_rows = 9
num_cols = 5

fig, ax = plt.subplots(num_rows, num_cols, figsize=(num_cols*3, num_rows*3))

for row in range(num_rows):
    for col in range(num_cols):
        i, j = pairs[num_cols * row + col]
        ax[row][col].set_title(f"{columns[i]} vs {columns[j]}", fontsize=8)
        ax[row][col].set_xticks([])
        ax[row][col].set_yticks([])

        for mark, (diagnosis, points) in zip(marks, points_by_diagnosis.items()):
            xs = [point[i] for point in points]
            ys = [point[j] for point in points]
            ax[row][col].scatter(xs, ys, marker=mark, label=diagnosis)

ax[-1][-1].legend(loc='lower right', prop={'size': 6})
plt.show()
```

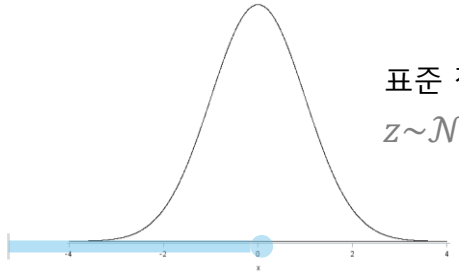
데이터셋 분리 (Q)

데이터셋 분리

```
import random
from scratch.machine_learning import split_data

# your code
```

데이터 표준화

정규분포 표준화 (standardization)



표준 정규 분포

$z \sim \mathcal{N}(0,1)$

$$z = \frac{x - \mu}{\sigma}$$

μ : 평균 σ: 표준 편차

- 데이터 구간의 값을 [0,1] 구간으로 정규화
- 값의 구간이 $[-\infty, \infty]$일 때 유용

데이터의 특징 별로 표준화

```
from scratch.working_with_data import scale, rescale

def normalization(dataset):
    return rescale(make_matrix(dataset))
```

훈련 데이터셋과 테스트 데이터셋 표준화

```
cancer_train_matrix = normalization(cancer_train)
cancer_test_matrix = normalization(cancer_test)
```

예측 (Q)

예측 함수를 정의해서 정확도와 혼동 행렬을 반환한도록 한다.

테스트 및 성능 측정

```
from typing import Tuple

def prediction(k : int) -> Tuple[float, Dict[Tuple[str, str], int]]:
    # your code
    return pct_correct, confusion_matrix
```

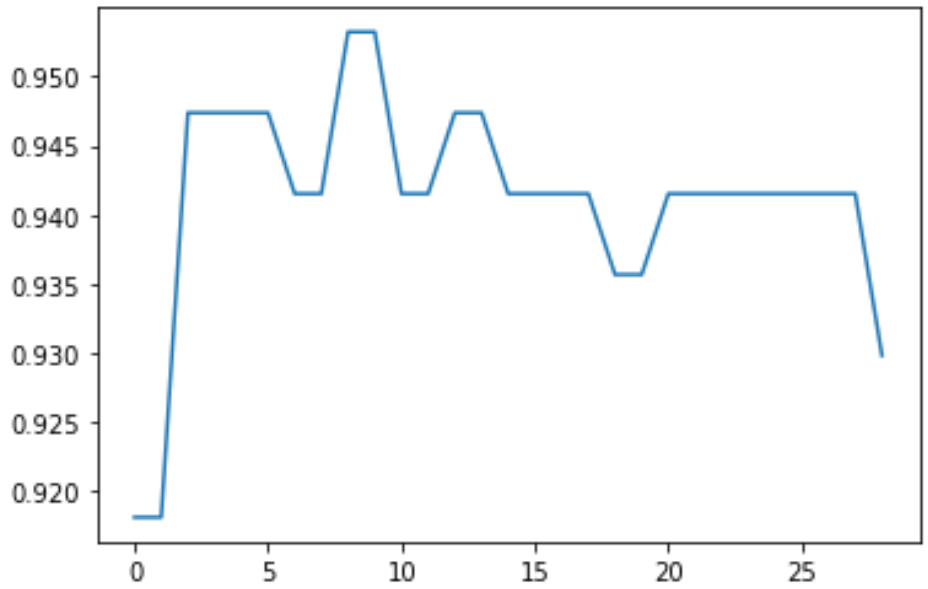
엘보 방법 (Elbow method) (Q)

k를 1에서 30까지 키워가면서 정확도를 측정해서 최대 정확도를 갖는 k를 선정한다.

엘보 방법 (Elbow method)으로 k 선정

```
k_candidate = (k for k in range(1, 30))
optimal_k = 0

acc_list : List[float] = []
for k in k_candidate:
    accuracy, confusion_matrix = prediction(k)
    acc_list.append(accuracy)
    # your code

print("")
print("Optimal k = ", optimal_k)
plt.plot(acc_list)
plt.show()
```

Optimal k = 9



k: 1 acc: 0.9181286549707602 defaultdict(<class 'int'>, {('M', 'M'): 62, ('B', 'B'): 95, ('M', 'B'): 5, ('B', 'M'): 9})
k: 3 acc: 0.9473684210526315 defaultdict(<class 'int'>, {('M', 'M'): 64, ('B', 'B'): 98, ('B', 'M'): 7, ('M', 'B'): 2})
k: 9 acc: 0.9532163742690059 defaultdict(<class 'int'>, {('M', 'M'): 66, ('B', 'B'): 97, ('B', 'M'): 5, ('M', 'B'): 3})

Thank you!