k-평균

구글 코랩에서 실행하기

KMeans 클래스

In [2]:
```
!wget https://bit.ly/fruits_300_data -O fruits_300.npy
```

```
--2023-10-29 12:20:34--  https://bit.ly/fruits_300_data
Resolving bit.ly (bit.ly)... 67.199.248.10, 67.199.248.11
Connecting to bit.ly (bit.ly)|67.199.248.10|:443... connected.
HTTP request sent, awaiting response... 301 Moved Permanently
Location: https://github.com/rickiepark/hg-mldl/raw/master/fruits_300.npy [following]
--2023-10-29 12:20:34--  https://github.com/rickiepark/hg-mldl/raw/master/fruits_300.npy
Resolving github.com (github.com)... 140.82.113.4
Connecting to github.com (github.com)|140.82.113.4|:443... connected.
HTTP request sent, awaiting response... 302 Found
Location: https://raw.githubusercontent.com/rickiepark/hg-mldl/master/fruits_300.npy [following]
--2023-10-29 12:20:35--  https://raw.githubusercontent.com/rickiepark/hg-mldl/master/fruits_300.npy
Resolving raw.githubusercontent.com (raw.githubusercontent.com)... 185.199.108.133, 185.199.109.133, 185.199.111.133, ...
Connecting to raw.githubusercontent.com (raw.githubusercontent.com)|185.199.108.133|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 3000128 (2.9M) [application/octet-stream]
Saving to: 'fruits_300.npy'

fruits_300.npy      100%[===================>]   2.86M  --.-KB/s    in 0.1s

2023-10-29 12:20:35 (29.2 MB/s) - 'fruits_300.npy' saved [3000128/3000128]
```

In [3]:
```
import numpy as np

fruits = np.load('fruits_300.npy')
# fruits_300.npy 파일에 있는 모든 데이터를 fruits에 할당
fruits_2d = fruits.reshape(-1, 100*100) # 2차원 배열로 변경
```

In [4]:
```
from sklearn.cluster import KMeans

km = KMeans(n_clusters=3, random_state=42) # 클러스터 개수 3을 설정
km.fit(fruits_2d) # 비지도 학습이므로 타겟 데이터를 사용하지 않음
```

```
/usr/local/lib/python3.10/dist-packages/sklearn/cluster/_kmeans.py:870: FutureWarning: The default value of `n_init` will change from 10 to 'auto' in 1.4. Set the value of `n_init` explicitly to suppress the warning
  warnings.warn(
```

Out[4]:
```
▾ KMeans
KMeans(n_clusters=3, random_state=42)
```

In [6]:
```
print(km.labels_)
# 군집된 결과는 km.labels_에 저장
# n_cluster를 3으로 설정했으므로, labels_ 배열의 값은 0, 1, 2 중 하나로 정해짐
# km.labels_ 출력
```

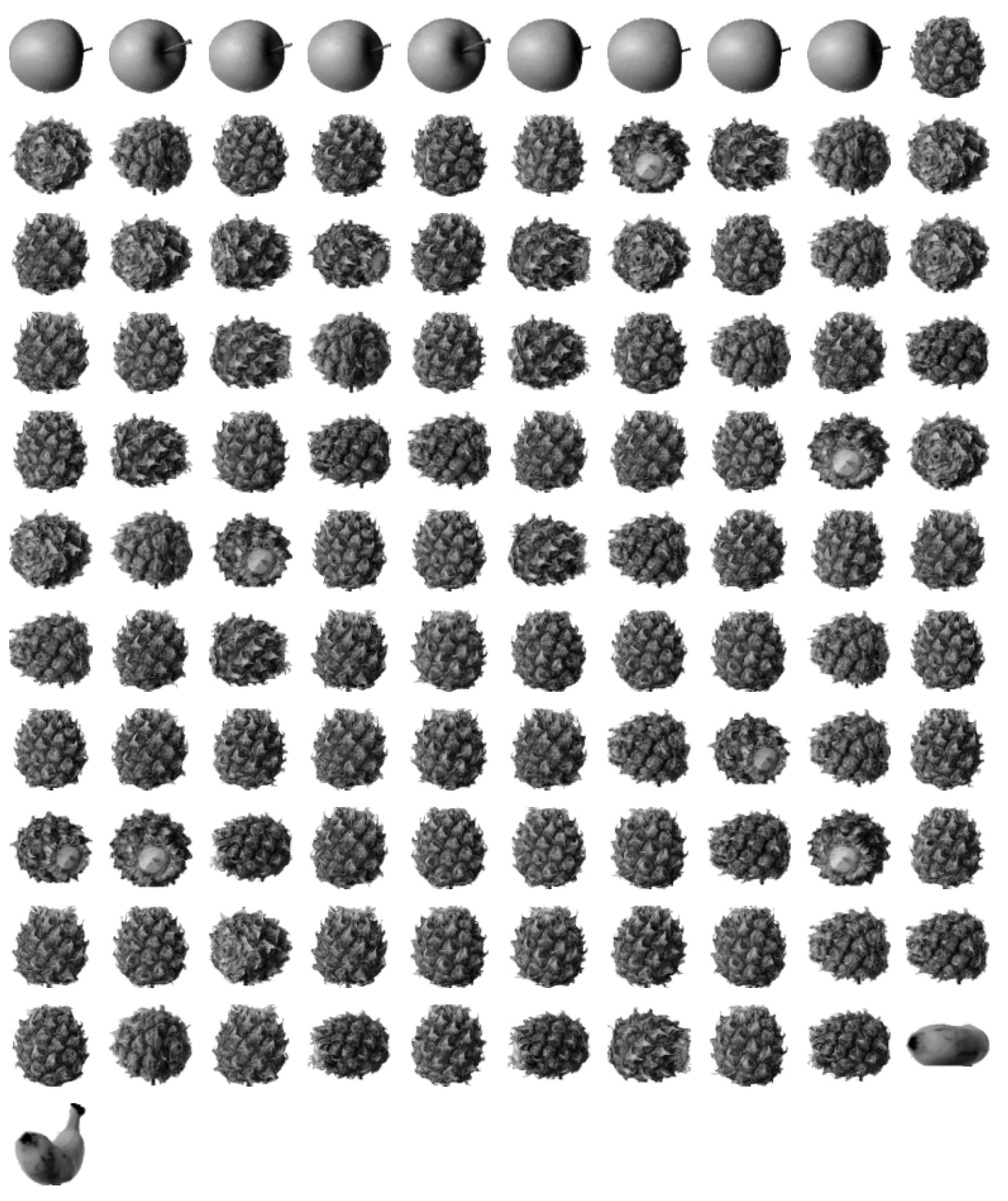
```
[2 2 2 2 2 0 2 2 2 2 2 2 2 2 2 2 2 2 0 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
 2 2 2 2 2 0 2 0 2 2 2 2 2 2 2 2 0 2 2 2 2 2 2 2 2 2 0 0 2 2 2 2 2 2 2 2 0 2
 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 0 2 2 2 2 2 2 2 2 2 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
 1 1 1 1 1 1 1 1 1 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
 1 1 1 1]
```

In [7]:
```
print(np.unique(km.labels_, return_counts=True)) # 레이블 0, 1, 2로 모은 샘플의 개수 확인
```

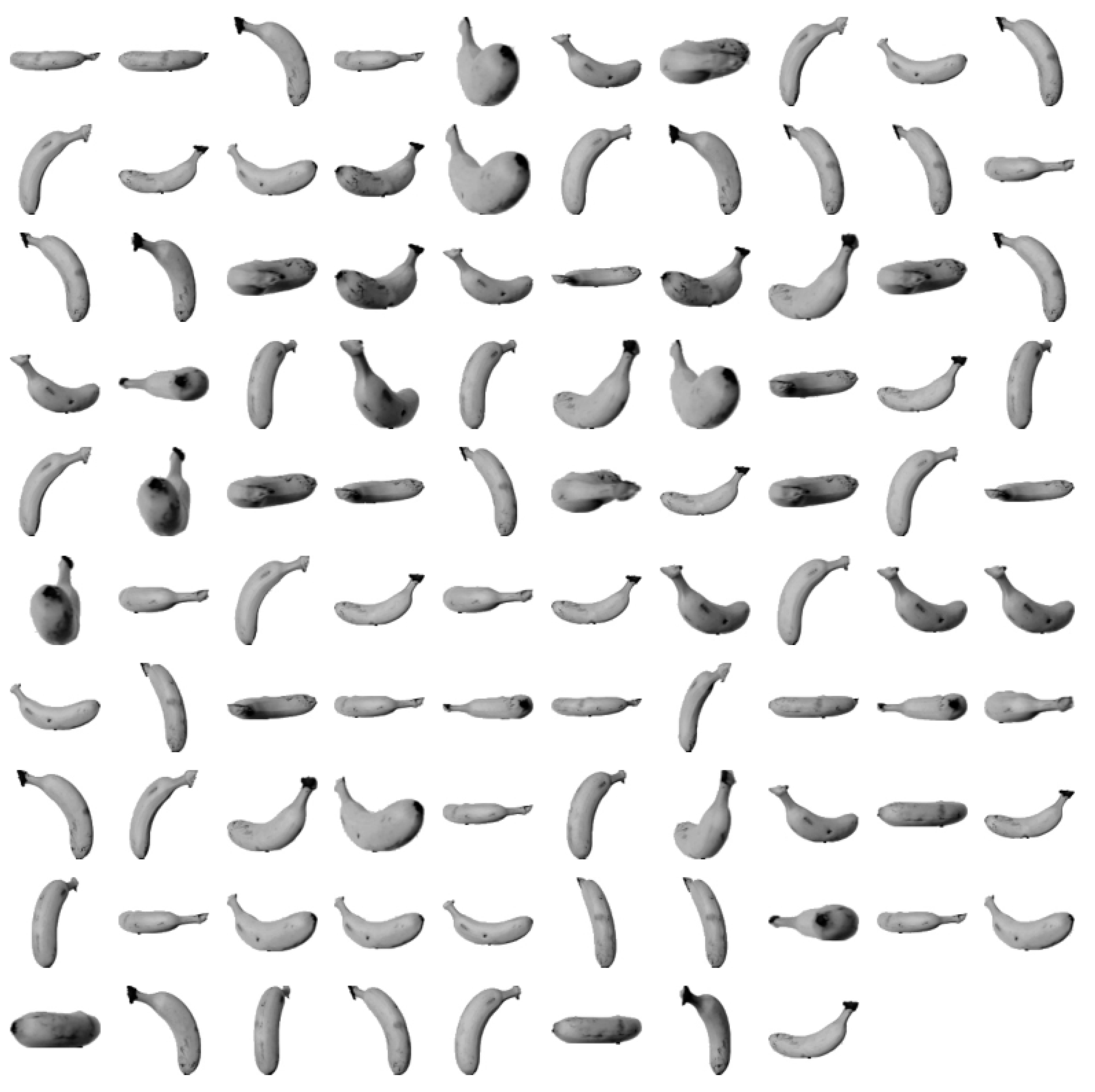
```
(array([0, 1, 2], dtype=int32), array([111,  98,  91]))
```

In [8]:
```
import matplotlib.pyplot as plt

def draw_fruits(arr, ratio=1):
    n = len(arr)    # 샘플의 개수

    rows = int(np.ceil(n/10)) # 한 줄에 10개씩 이미지를 그림, 샘플 개수를 10으로 나누어 전체 행 개수를 계산

    cols = n if rows < 2 else 10 # 행이 1개이면 열 개수는 샘플 개수, 그렇지 않으면 10개로 설정
    fig, axs = plt.subplots(rows, cols,
                            figsize=(cols*ratio, rows*ratio), squeeze=False)
    for i in range(rows):
        for j in range(cols):
            if i*10 + j < n:
                axs[i, j].imshow(arr[i*10 + j], cmap='gray_r')
            axs[i, j].axis('off')
            # n개 까지만 그림
    plt.show()
```

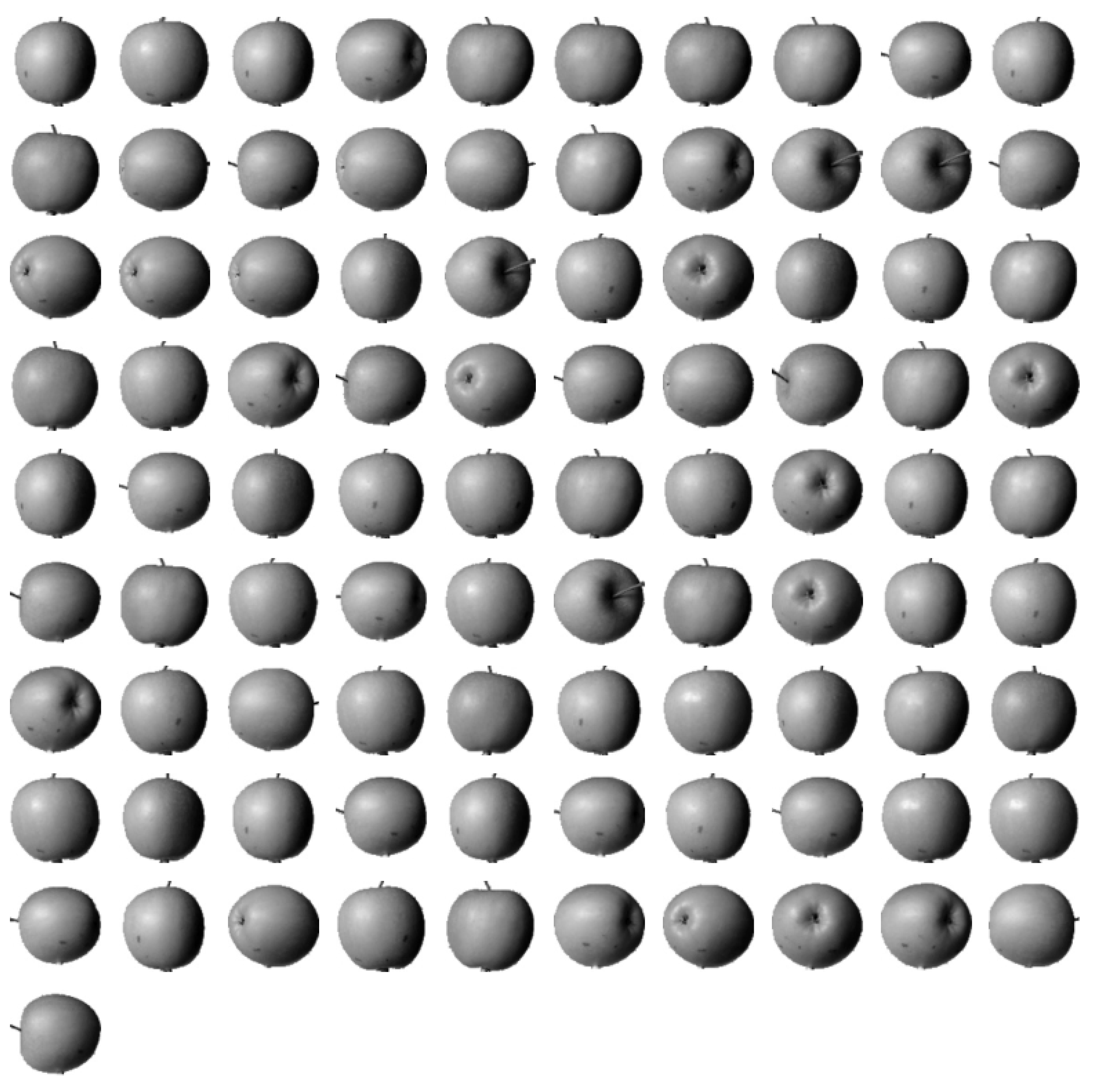
In [9]:
```
draw_fruits(fruits[km.labels_==0]) # km.labels==0을 통해 km.labels_ 배열에서 값이 0인 위치는 True로 되며 True인 위치의 원소만을 모두 출력
```



In [10]:
```
draw_fruits(fruits[km.labels_==1])
```

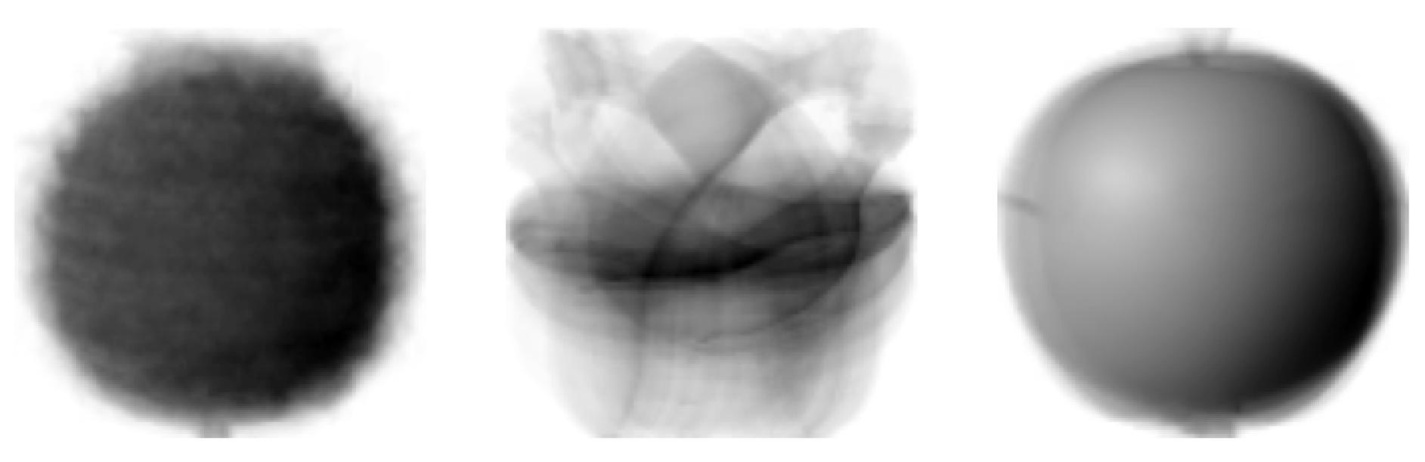


In [11]:
```
draw_fruits(fruits[km.labels_==2])
```



클러스터 중심

In [12]:
```
draw_fruits(km.cluster_centers_.reshape(-1, 100, 100), ratio=3)
# kmeans 클래스가 최종적으로 찾은 클러스터 중심은 cluster_centers_ 속성에 저장
# 이 배열은 fruits_2d 샘플의 클러스터 중심이기 때문에 100 x 100 크기의 2차원 배열로 변경
```



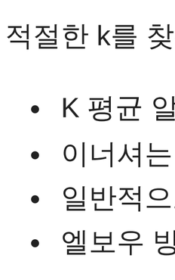
In [13]:
```
print(km.transform(fruits_2d[100:101]))
# 인덱스가 100인 샘플에 transform()메서드를 적용 -> transform()메서드는 특성값을 변환하는 도구로 사용할 수 있다는 의미
```

```
[[3393.8136117  8837.37750892 5267.70439881]]
```

In [14]:
```
print(km.predict(fruits_2d[100:101])) # predict를 통해 가장 가까운 클러스터 중심을 예측 클래스로 출력
```

```
[0]
```

In [15]:
```
draw_fruits(fruits[100:101])
```



In [16]:
```
print(km.n_iter_) # 알고리즘이 반복한 횟수
```

```
4
```

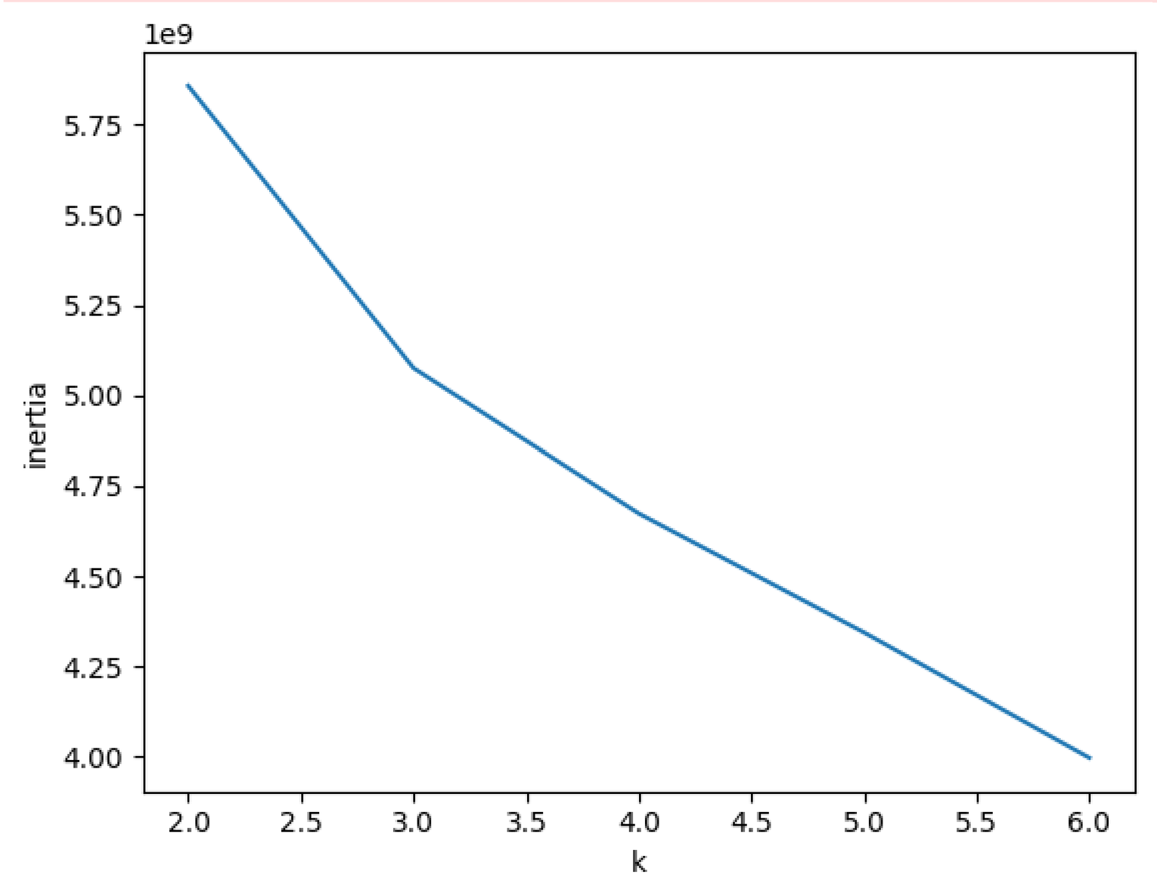
최적의 k 찾기

적절한 k를 찾는 대표적인 방법 : 엘보우 방법

- K 평균 알고리즘은 클러스터 중심과 클러스터에 속한 샘플 사이의 거리를 잴 수 있으며, 이 거리의 제곱합 = 이너셔
- 이너셔는 클러스터에 속한 앰플이 얼마나 가깝게 모여 있는지를 나타내는 값으로 생각 가능
- 일반적으로 클러스터 개수가 늘어나면 클러스터 개개의 크기는 줄어들기 때문에 이너셔도 줄어듦
- 엘보우 방법은 클러스터 개수를 늘려가면서 이너셔의 변화를 관찰하여 최적의 클러스터 개수를 찾는 방법

In [17]:
```
inertia = []
for k in range(2, 7):
    km = KMeans(n_clusters=k, random_state=42)
    km.fit(fruits_2d)
    inertia.append(km.inertia_)
    # K의 개수를 2~6까지 바꿔가며 kmeans 클래스를 5번 훈련
    # 모델을 훈련 한 후 inertia_ 속성에 저장된 이너셔 값을 inertia 리스트에 추가
plt.plot(range(2, 7), inertia)
plt.xlabel('k')
plt.ylabel('inertia')
plt.show()
```

```
/usr/local/lib/python3.10/dist-packages/sklearn/cluster/_kmeans.py:870: FutureWarning: The default value of `n_init` will change from 10 to 'auto' in 1.4. Set the value of `n_init` explicitly to suppress the warning
  warnings.warn(
/usr/local/lib/python3.10/dist-packages/sklearn/cluster/_kmeans.py:870: FutureWarning: The default value of `n_init` will change from 10 to 'auto' in 1.4. Set the value of `n_init` explicitly to suppress the warning
  warnings.warn(
/usr/local/lib/python3.10/dist-packages/sklearn/cluster/_kmeans.py:870: FutureWarning: The default value of `n_init` will change from 10 to 'auto' in 1.4. Set the value of `n_init` explicitly to suppress the warning
  warnings.warn(
/usr/local/lib/python3.10/dist-packages/sklearn/cluster/_kmeans.py:870: FutureWarning: The default value of `n_init` will change from 10 to 'auto' in 1.4. Set the value of `n_init` explicitly to suppress the warning
  warnings.warn(
/usr/local/lib/python3.10/dist-packages/sklearn/cluster/_kmeans.py:870: FutureWarning: The default value of `n_init` will change from 10 to 'auto' in 1.4. Set the value of `n_init` explicitly to suppress the warning
  warnings.warn(
```



In [ ]: