Basics of k-means clustering

CLUSTER ANALYSIS IN PYTHON

Shaumik Daityari
Business Analyst

Why k-means clustering?

- A critical drawback of hierarchical clustering: runtime
- K means runs significantly faster on large datasets

계층 클러스터링의 중요한 결점: 런타임
k는 대규모 데이터셋에서 훨씬 빠르게 실행됨을 의미

Step 1: Generate cluster centers

scipy에서 k - means clustering을 수행하려면 두 단계가 있음
1. generate cluster centers을 생성한 다음 군집 레이블을 할당
(1) 첫번째 단계는 kmeans 방법으로 수행

```
kmeans(obs, k_or_guess, iter, thresh, check_finite)
```

- `obs` : standardized observations whiten 방법을 통해 표준화된 관찰 목록
- `k_or_guess` : number of clusters cluster의 수
- `iter` : number of iterations (default: 20) 수행할 알고리즘의 반복 횟수 (default : 20)
- `thres` : threshold (default: 1e-05) 임계값으로 마지막 k - means 반복 이후의 왜곡 변화가 임계값 이하일 경우 알고리즘이 종료됨. defualt값은 0.00001
- `check_finite` : whether to check if observations contain only finite numbers (default: True)

마지막 인수는 무한값 또는 NaN값이 존재하는지 확인하는 부울값 (default값 : True) NaN 또는 무한 값을 가진 데이터 point가 분류에 고려되지 않으므로 결과가 정확하고 치우치지 않음

Returns two objects: cluster centers, distortion

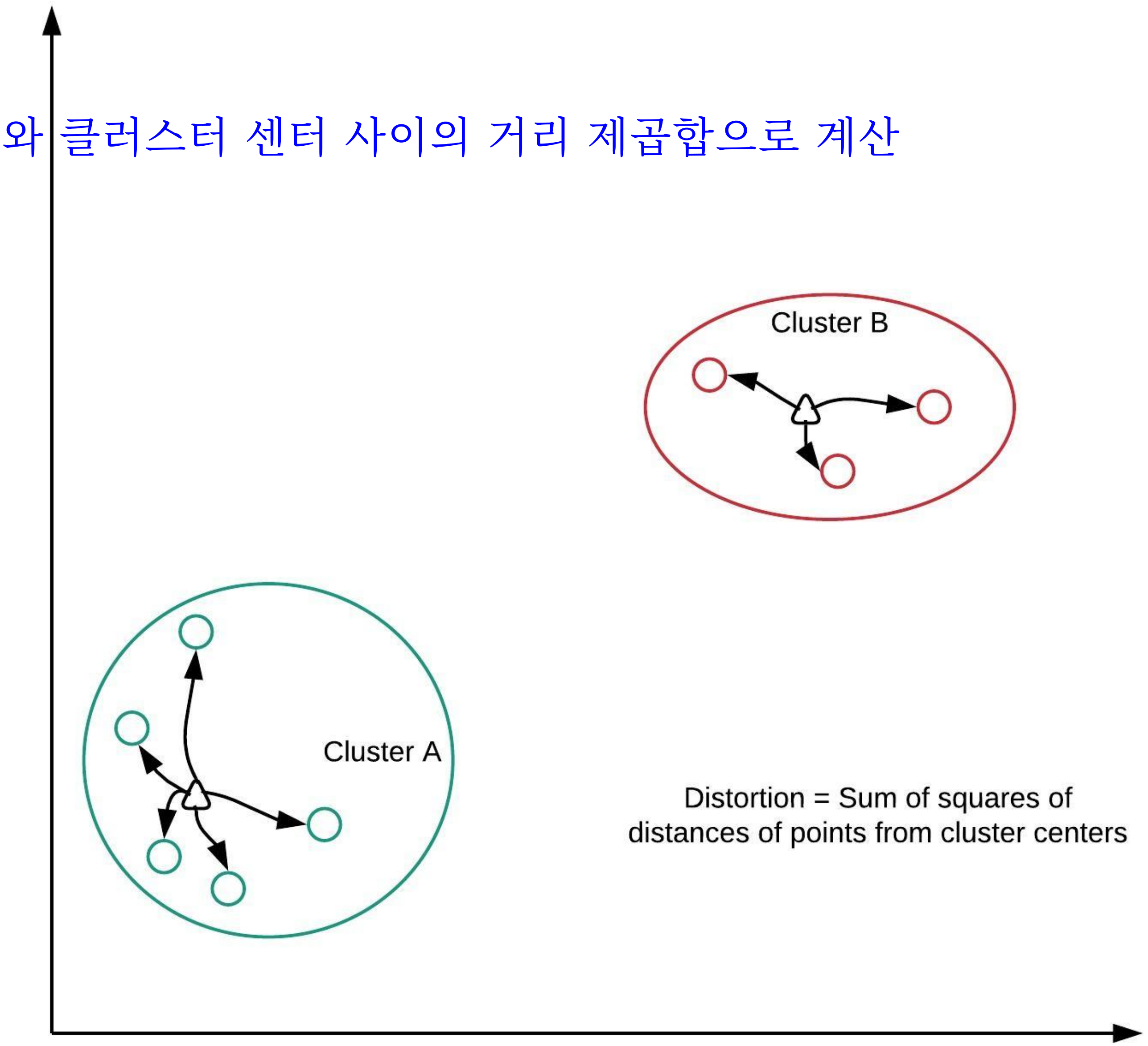
k - means 함수는 cluster center과 왜곡이라는 두 개의 인수를 return
cluster center는 code book이라고도 함

k - means cluster에서는 작업 수가 상당히 적기 때문에 계층적 군집화에 비해 k - menas이 매우 빠르게 실행

How is distortion calculated?

왜곡은 이 그림에 나타나듯이 데이터 포인트와 클러스터 센터 사이의 거리 제곱합으로 계산

$$J = \sum_{n=1}^{N}\sum_{k=1}^{K} r_{nk}\|\mathbf{x}_n - \mu_k\|^2$$



Distortion = Sum of squares of distances of points from cluster centers

Step 2: Generate cluster labels

```
vq(obs, code_book, check_finite=True)
```

vq를 사용하여 cluster label을 생성

3개의 인수 필요

- `obs` : standardized observations
  whiten method를 통해 표준화된 관찰의 목록
- `code_book` : cluster centers
  kmeans method의 첫번째 출력인 코드북
- `check_finite` : whether to check if observations contain only finite numbers (default: True)
  세 번째 옵션의 인수는 check_finite 이는 무한 또는 NaN 값이 존재하는지 여부를 확인하기 위해 데이터에 대해 체크를 실행할 필요가 있는지를 나타내는 부울값.

Returns two objects: a list of cluster labels, a list of distortions

이 함수는 "code book index"라고도 하는 클러스터 레이블과 왜곡을 return

A note on distortions

- `kmeans` returns a single value of distortions
- `vq` returns a list of distortions.

kmeans는 전체 데이터를 기준으로 단일 왜곡 값을 반환하는
반면 vq는 각 데이터 지점에 대해 하나씩 왜곡 목록을 반환

vq 방법의 왜곡 목록의 평균은 동일한 관측치 목록이 통과될 경우 kmeans 방법의 왜곡 값과 거의 같아야 합니다.

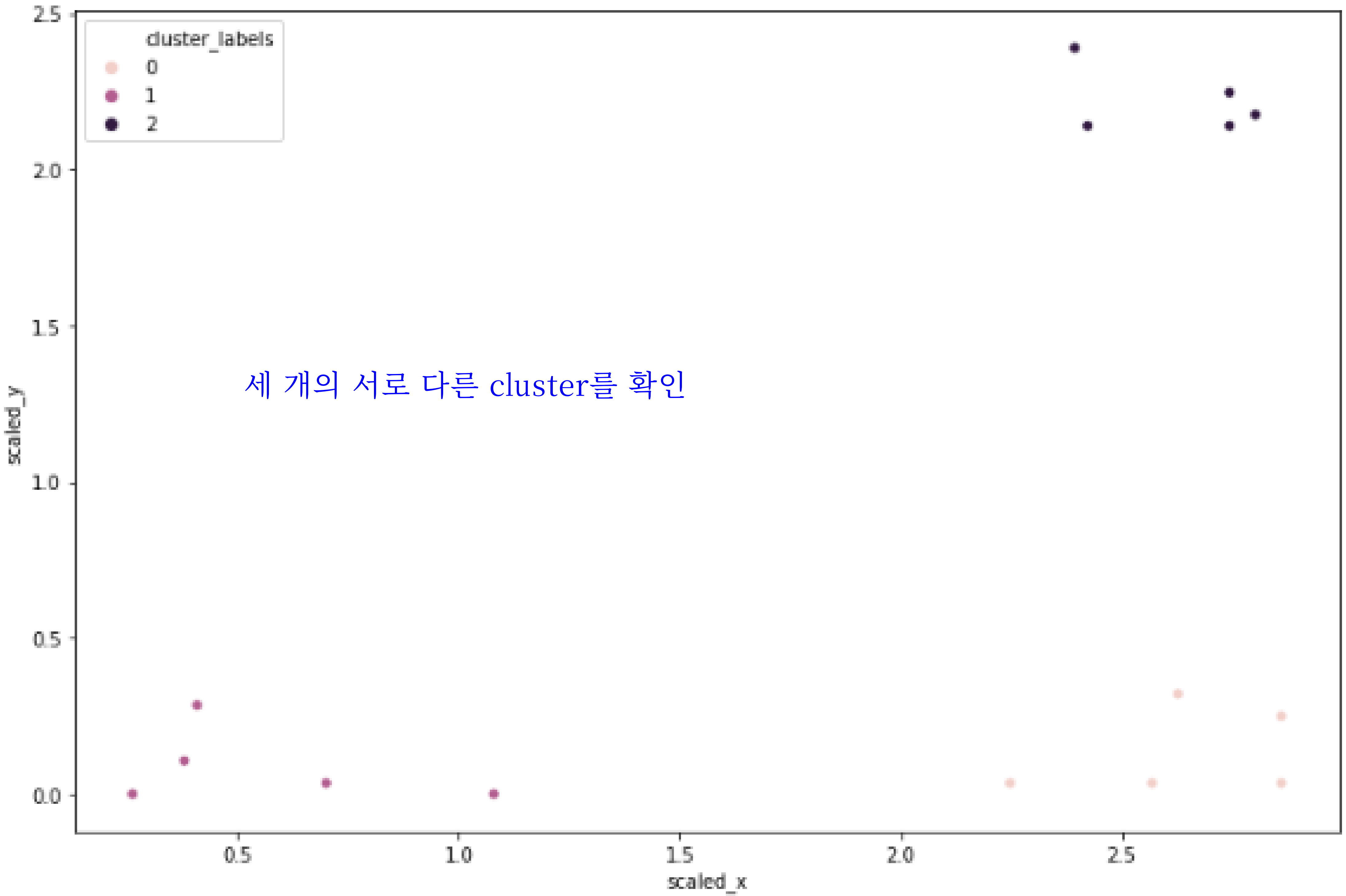
Running k-means

```
# Import kmeans and vq functions
from scipy.cluster.vq import kmeans, vq
```

```
# Generate cluster centers and labels
cluster_centers, _ = kmeans(df[['scaled_x', 'scaled_y']], 3)
df['cluster_labels'], _ = vq(df[['scaled_x', 'scaled_y']], cluster_centers)
```

kmeans를 사용하여 cluster center를 가져오고
vq를 사용하여 cluster label을 얻음

```
# Plot clusters
sns.scatterplot(x='scaled_x', y='scaled_y', hue='cluster_labels', data=df)
plt.show()
```



세 개의 서로 다른 cluster를 확인

Next up: exercises!

How many clusters?

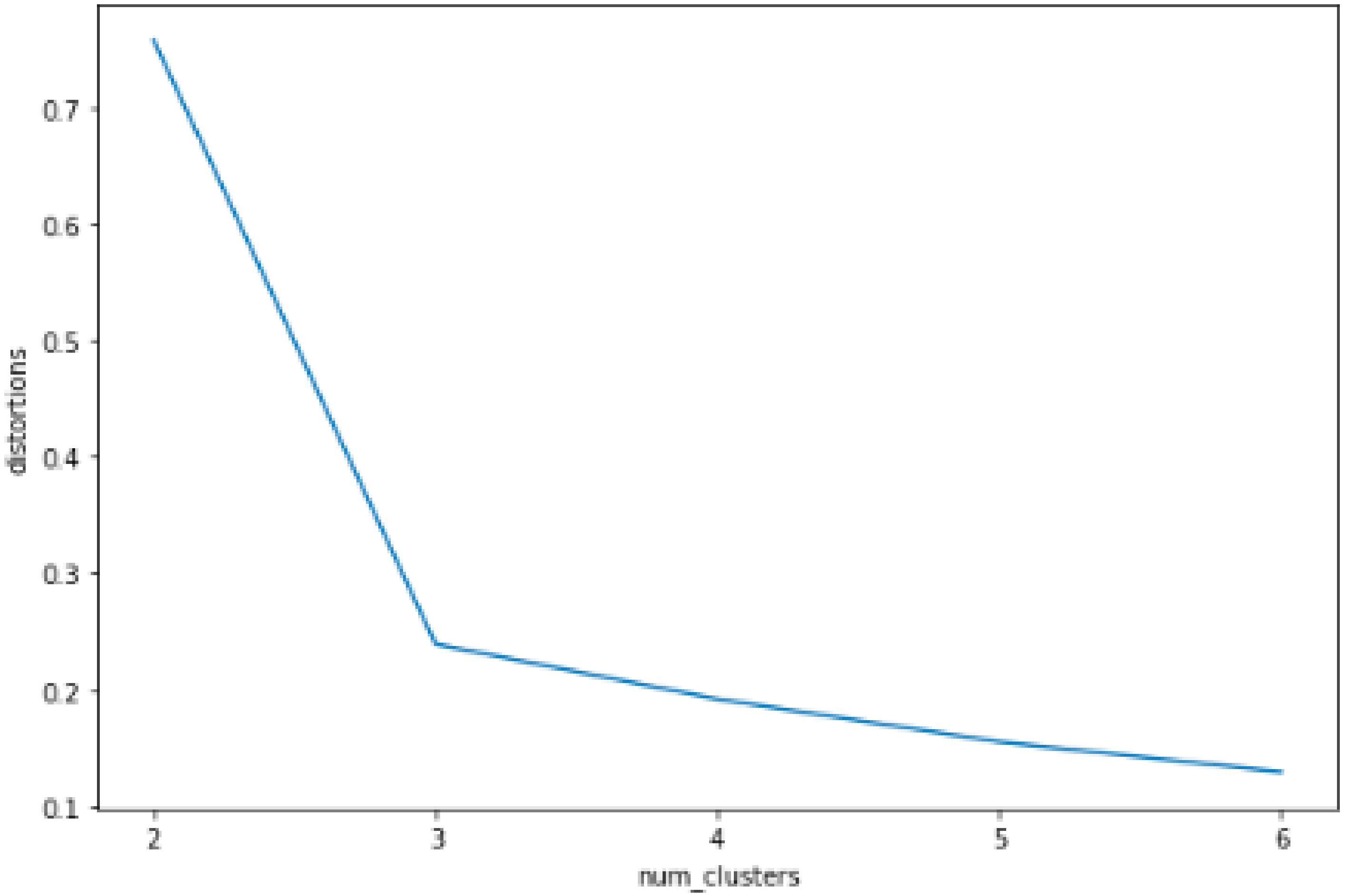
CLUSTER ANALYSIS IN PYTHON

이 비디오는 k - means clustering에서 cluster수를 결정하는 방법에 대해 설

Shaumik Daityari
Business Analyst

How to find the right k?

- No *absolute* method to find right number of clusters (k) in k-means clustering
- Elbow method



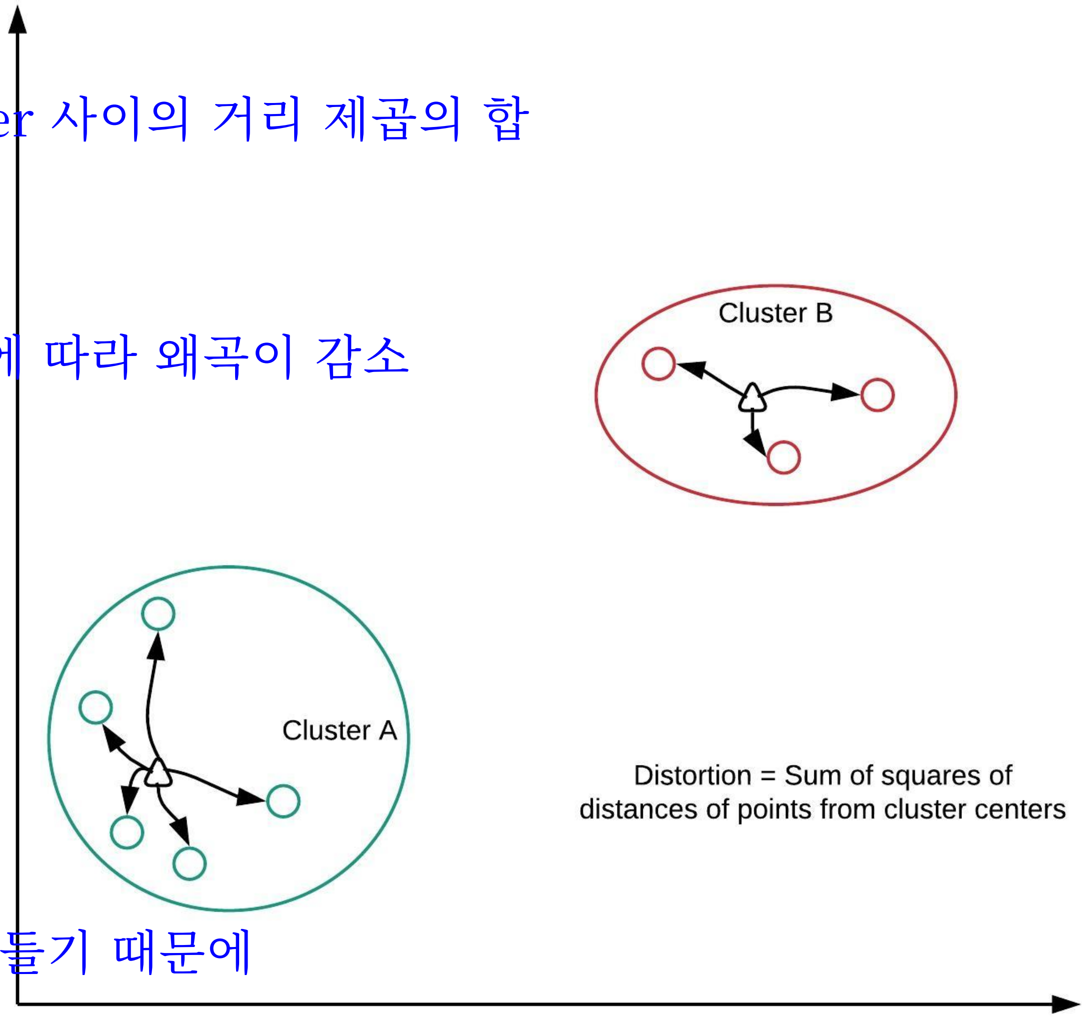
k - means clustering에 대한 한 가지 비판은 데이터 세트에 cluster 수가 얼마나 되는지 알 수 있는 올바른 방법이 없다는 것

특정 지시 방법이 있으며, 이 장에서는 이러한 방법 중 하나에 대해 설명
- > elbow plot을 구성하여 dataset에 적합한 cluster수를 결정

Distortions revisited

- Distortion: sum of squared distances of points from cluster centers 왜곡은 각 data point와 cluster center 사이의 거리 제곱의 합
- Decreases with an increasing number of clusters 이상적으로는 왜곡이 군집 수와 역관계이므로 군집수가 증가함에 따라 왜곡이 감소
- Becomes zero when the number of clusters equals the number of points cluster수가 0이 되면 0이 됨 point수와 같음
- Elbow plot: line plot between cluster centers and distortion

데이터를 더 작은 조각으로 분할하면 cluster가 서로 가까워지고 왜곡이 줄어들기 때문에 이러한 추세는 직관적



Distortion = Sum of squares of distances of points from cluster centers

Elbow method

- Elbow plot: plot of the number of clusters and distortion
- Elbow plot helps indicate number of clusters present in data

- 먼저 데이터에서 다양한 수의 cluster로 k - means clustering을 실행하고
  x축에 cluster 수와 y축에 왜곡이 있는 elbow plot을 구성 (cluster 수는 1~data point 수까지)
- 이상적인 plot은 cluster의 수를 늘릴 때 왜곡이 상대적으로 덜 감소하는 것

Elbow method in Python

```
# Declaring variables for use
distortions = []

num_clusters = range(2, 7)
```

클러스터 수의 범위를 결정

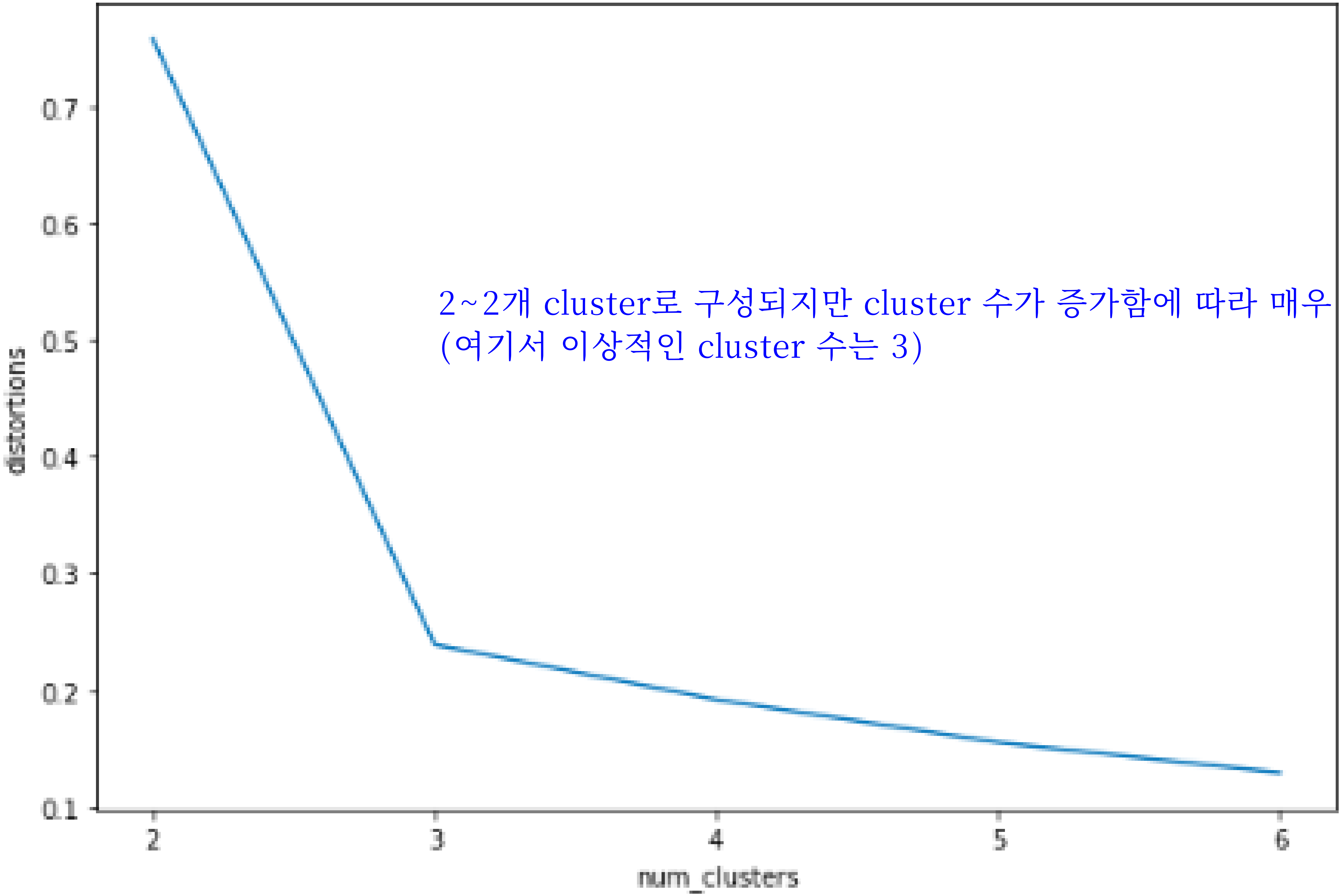
```
# Populating distortions for various clusters
for i in num_clusters:
    centroids, distortion = kmeans(df[['scaled_x', 'scaled_y']], i)
    distortions.append(distortion)
```

각 클러스터에 대한 k - means 방법을 실행하고 나중에 사용하기 위해 목록에서 해당 왜곡을 수집

```
# Plotting elbow plot data
elbow_plot_data = pd.DataFrame({'num_clusters': num_clusters,
                                'distortions': distortions})

sns.lineplot(x='num_clusters', y='distortions',
             data = elbow_plot_data)
plt.show()
```

각 수에 대한 왜곡이 있는 df만듦



2~2개 cluster로 구성되지만 cluster 수가 증가함에 따라 매우 점진적으로 감소
(여기서 이상적인 cluster 수는 3)

Final thoughts on using the elbow method

- Only gives an indication of optimal _k_ (numbers of clusters)
- Does not always pinpoint how many _k_ (numbers of clusters)
- Other methods: average silhouette and gap statistic

- elbow 방법은 이상적인 cluster 수만 표시함

경우에 따라 최저그이 k를 찾는 것만으로는 충분하지 않을 수 있음

ex) elbow 방법은 데이터가 고르게 분포될 때 실패함

- 최적의 수를 찾는 다른 방법이 있음

=> 평균 실루엣 및 간격 통계 방법과 같은 cluster

Next up: exercises

CLUSTER ANALYSIS IN PYTHON

Limitations of k-means clustering

CLUSTER ANALYSIS IN PYTHON

Shaumik Daityari
Business Analyst

클러스터링 방법의 제한과 k - 평균 클러스터링을 사용할 때 주의해야 하는 방법에 대해 중점적으로 살펴봄

Limitations of k-means clustering

- How to find the right _K_ (number of clusters)?
- Impact of seeds
- Biased towards equal sized clusters

앞에서 k - means clustering이 계층형 clustering의 가장 큰 단점인 runtime을 극복하는 것을 봄
단, 사용시 고려해야할 제한이 있음

- 첫번째 문제, 올바른 cluster 수 k를 찾는 절차. elbow method는 올바른 k를 결정하는 방법 중 하나지만 항상 효과 있는 것은 아님
- k - means clustering의 다음 제한은 seed가 clustering에 미치는 영향
- 마지막 한계 : 같은 크기의 cluster가 형성됨

Impact of seeds

seed가 cluster에 미치는 영향

Initialize a random seed

```
from numpy import random
random.seed(12)
```

초기 cluster center을 정의한느 프로세스는 랜덤이므로
초기화가 최종 cluster에 영향을 미칠 수 있음.
=> 따라서 동일한 데이터 집합에서 k-means clustering을 여러번
실행할 때 일관된 결과를 얻으려면 난수 생성을 위한 초기화 매개변수를 설정

seed는 numpy 단위
단일 정수 또는 1D array를 인수로 전달할 수 있음
알고리즘을 실행하기 전에 두 개의 다른 seed를 전달했을 때의 k-means clustering결과를 보면
테스트를 위해 랜덤으로 생성된 200개의 point 목록을 추출하여 5개의 cluster를 사용함.
두 경우 cluster 크기가 다르다는 것을 알 수 있음

Seed: `np.array(1000, 2000)`

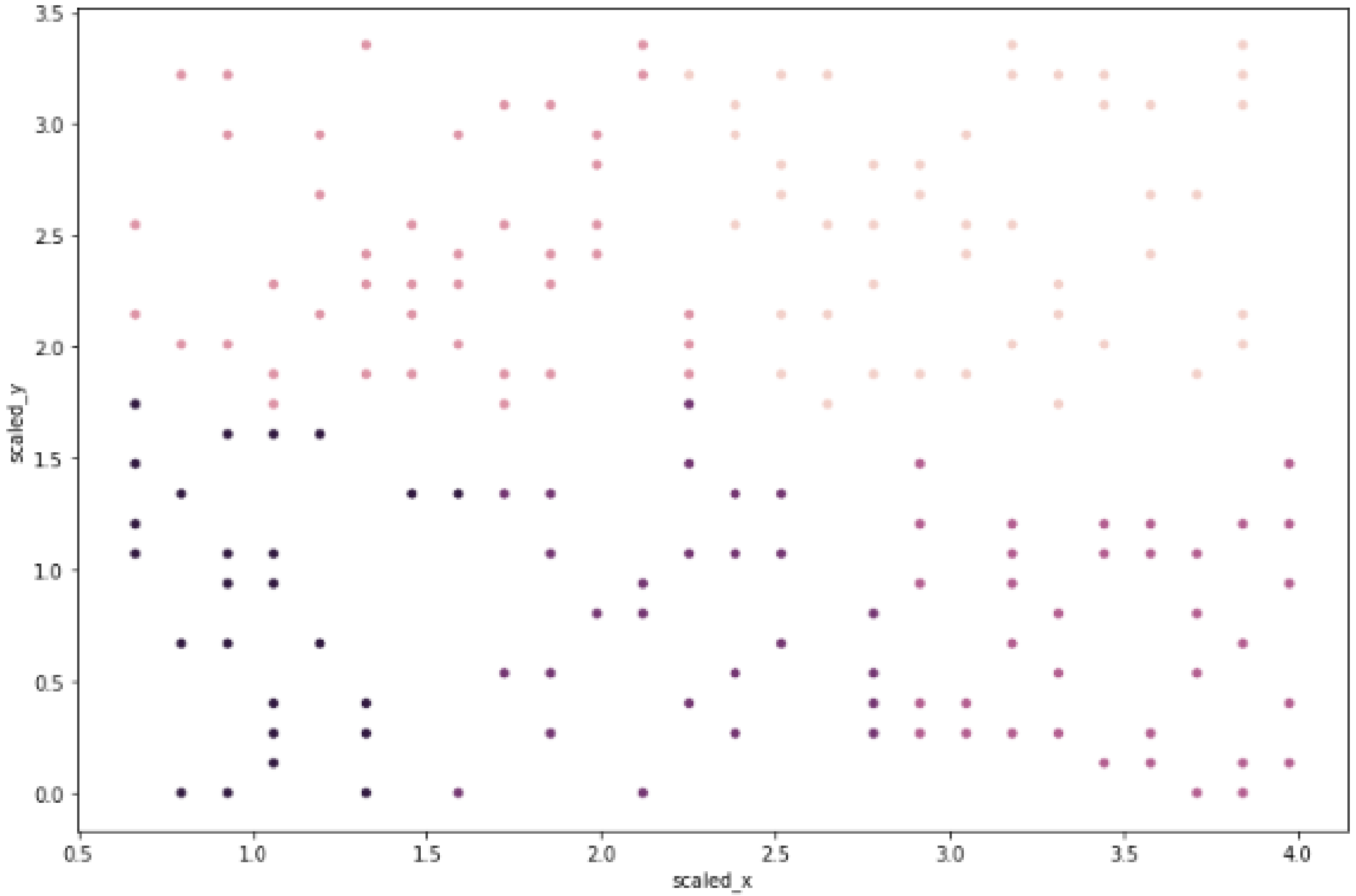
Cluster sizes: 29, 29, 43, 47, 52

Seed: `np.array(1,2,3)`

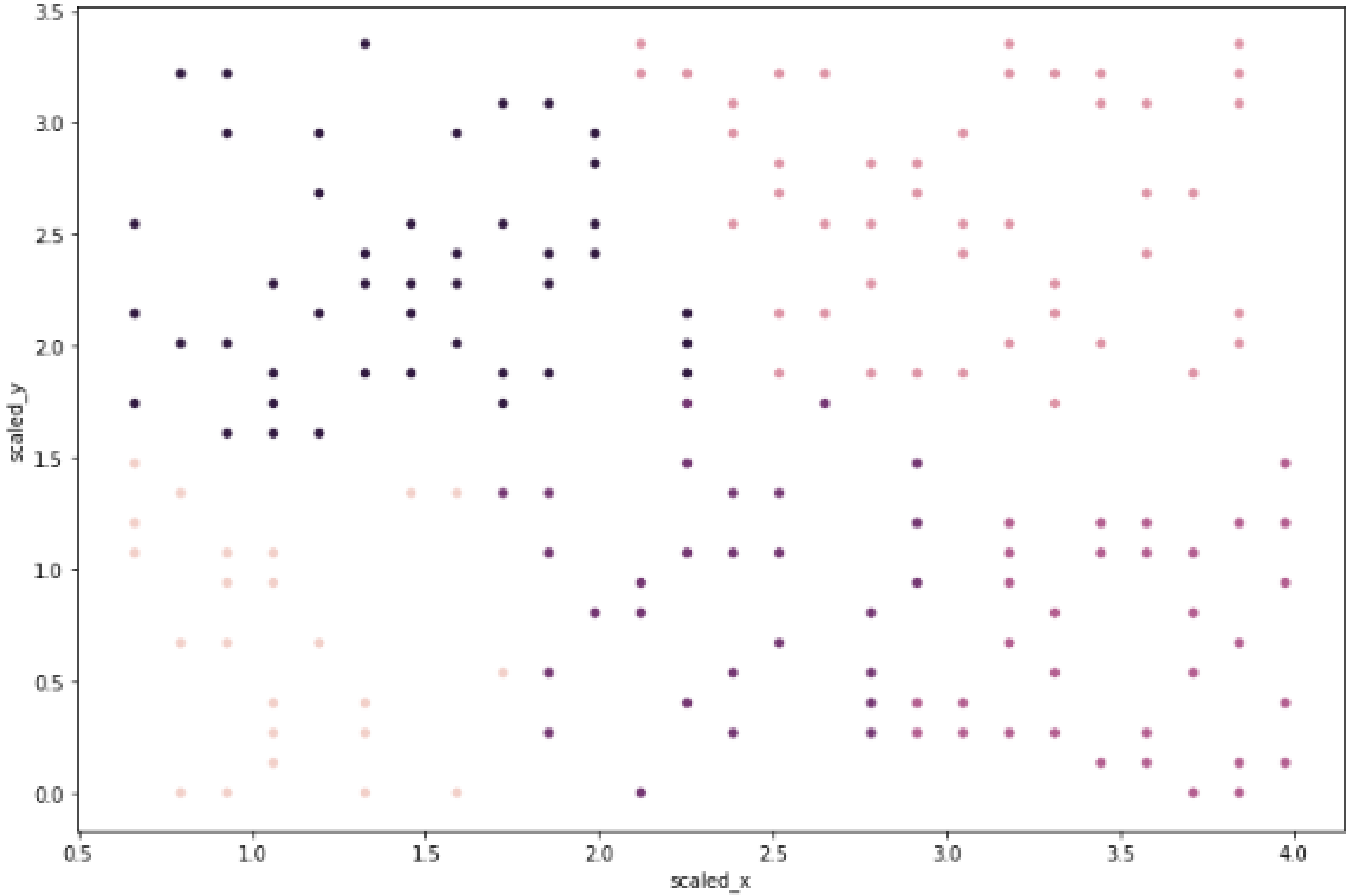
Cluster sizes: 26, 31, 40, 50, 53

Impact of seeds: plots

Seed: `np.array(1000, 2000)`



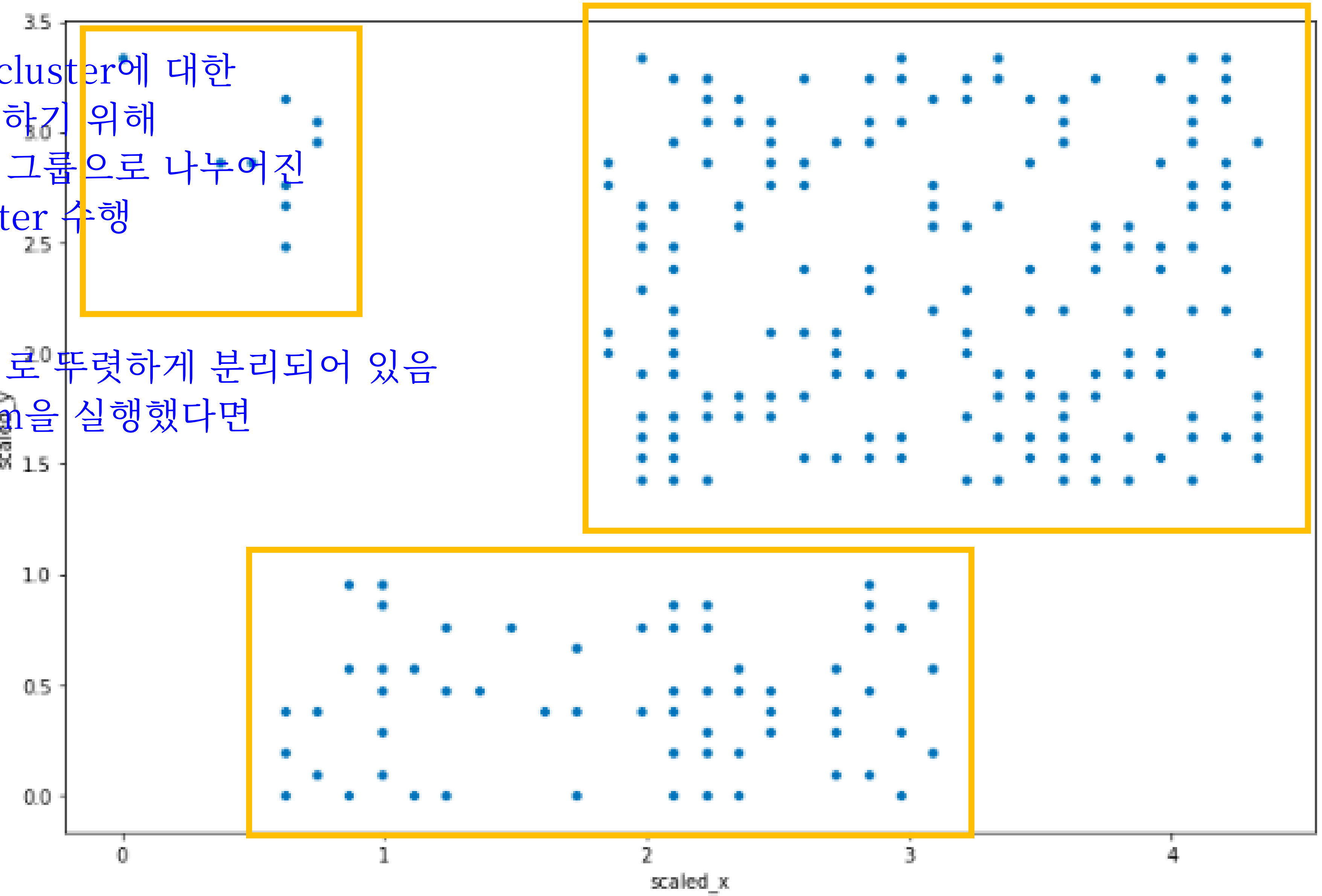
Seed: `np.array(1,2,3)`



Uniform clusters in k means

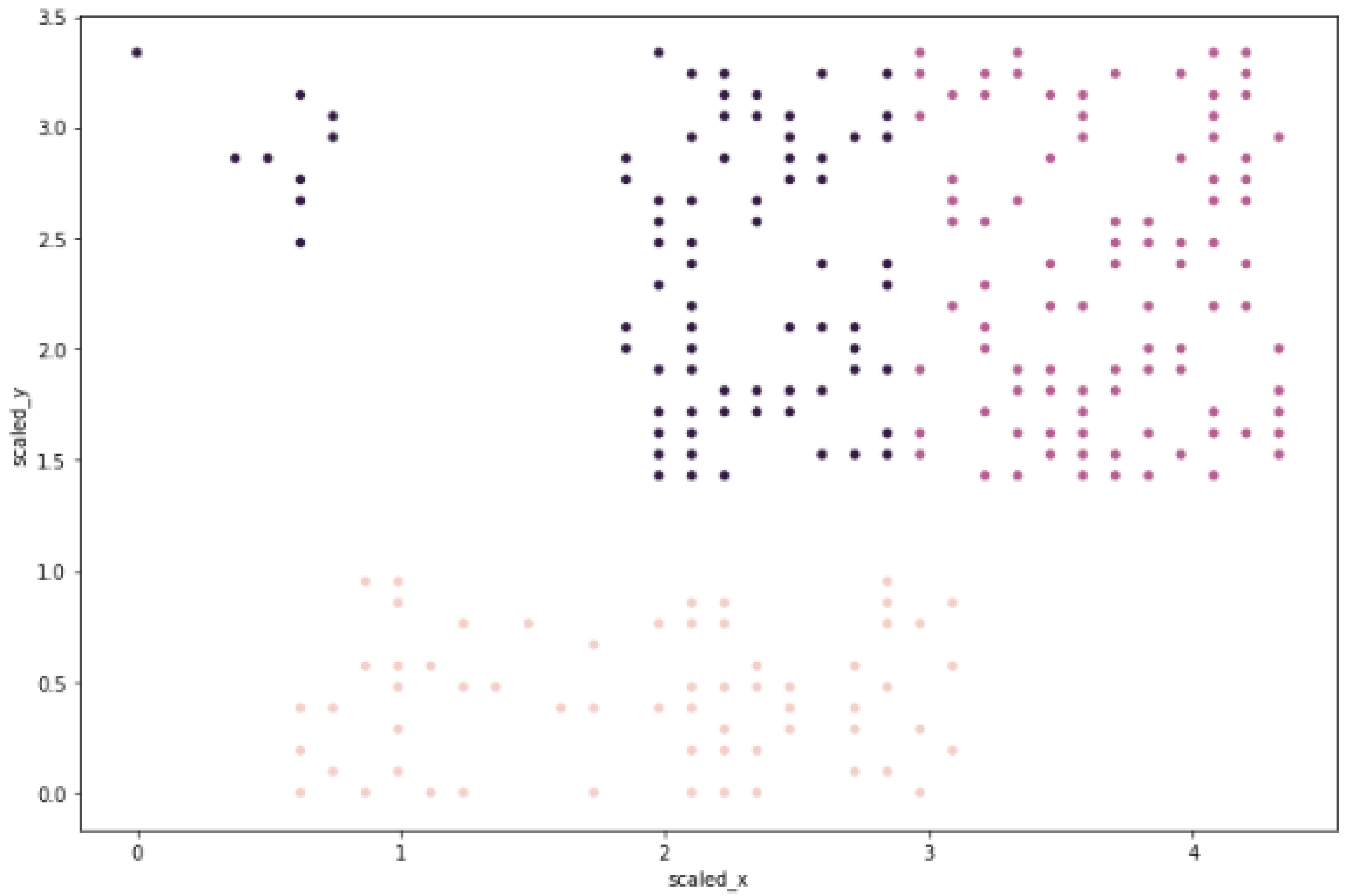
분산을 최소화하기 위해 균일한 cluster에 대한
k - means 군집화의 편향을 설명하기 위해
200, 70 및 10의 균일하지 않은 그룹으로 나누어진
총 280개 point set에 대해 cluster 수행

그래픽으로 보면 3개의 클러스터로 뚜렷하게 분리되어 있음
= > 따라서 clustering algorithm을 실행했다면
이 세 cluster를 선택해야함

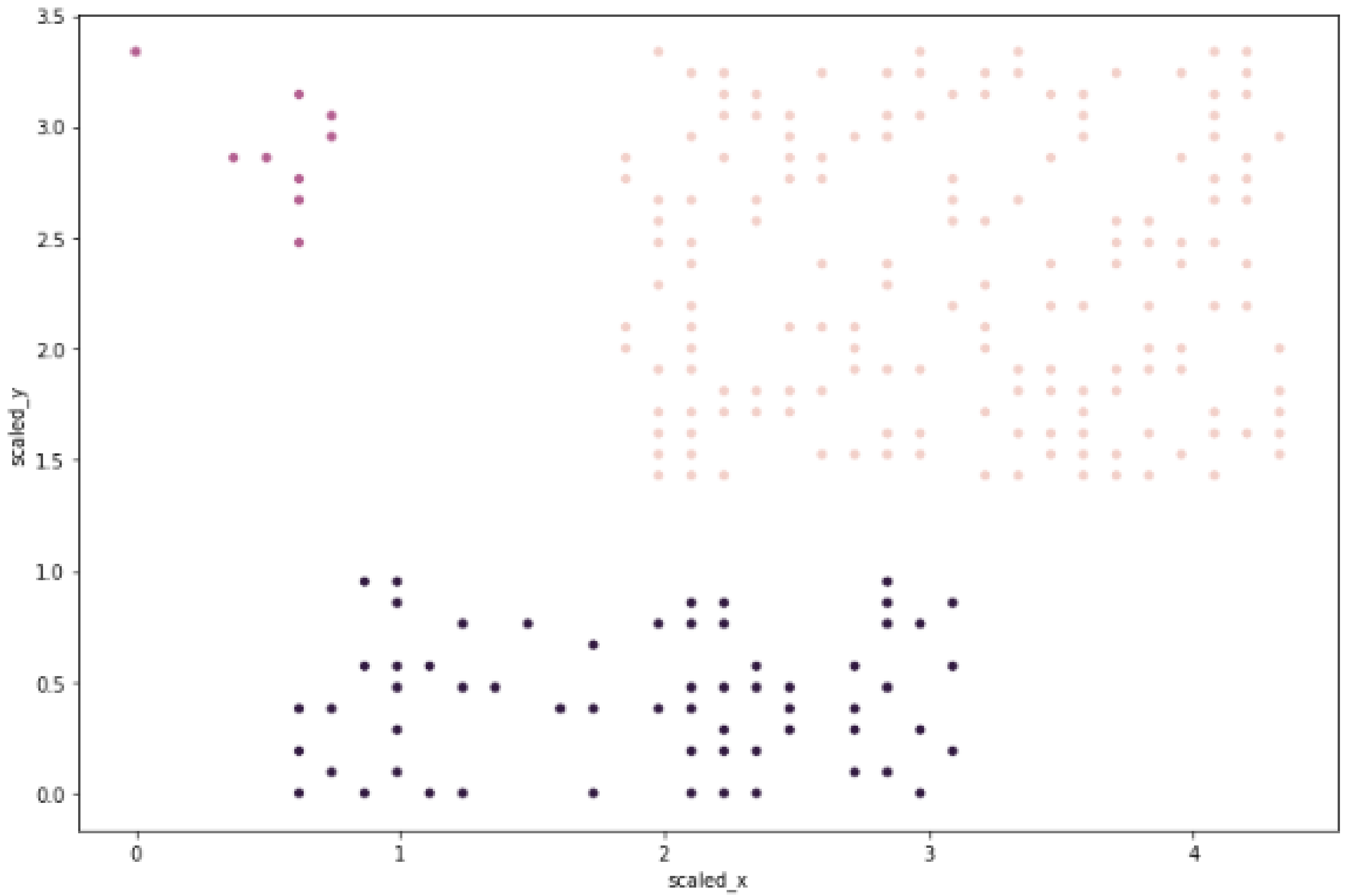


Uniform clusters in k-means: a comparison

K-means clustering with 3 clusters



Hierarchical clustering with 3 clusters



k - means cluster 분석 결과를 보면 seed를 변경해도 직관적이지 않은 군직이 나타남
이유 : k - means cluster의 개념은 왜곡을 최소화하는 것이기 때문
= > 따라서 cluster는 비슷한 면적을 가지지만 데이터 포인트 수는 비슷하지 않음
그러나 완전한 방법을 사용하여 동일한 데이터셋에서 계층적 clustering을 수행한 결과 형성된 cluster가 직관적이고 이전 slide의 가정과 일치

Final thoughts

- Each technique has its pros and cons
- Consider your data size and patterns before deciding on algorithm
- Clustering is exploratory phase of analysis

각 기술에는 장단점이 있는데 각 기술을 적용하기 전에 각 기술의 기본 전제조건을 파악해야함
이상적으로는 알고리즘을 확저하기 전에 데이터 크기, 데이터 패턴, 리소스 및 사용 가능한 시간에 대해 생각해야함
clustering은 여전히 분석의 탐색 단계

Next up: exercises

CLUSTER ANALYSIS IN PYTHON