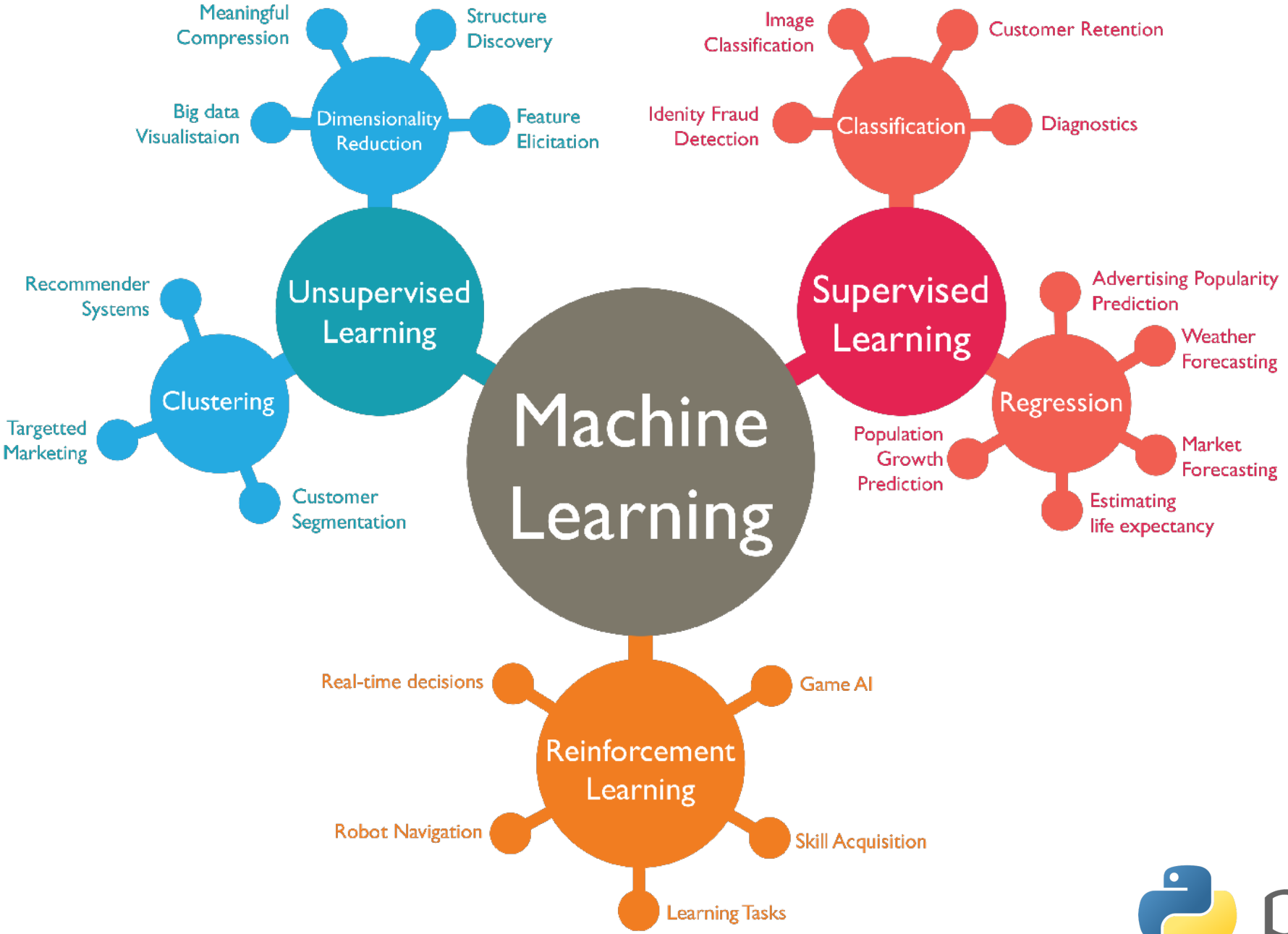
python™

Machine Learning Algorithm

- 지도학습 (supervised learning)
  데이터에 대한 label(정답)이 주어진 상태에서 컴퓨터를 학습 시키는 것

- **비지도학습 (unsupervised learning)**
  데이터에 대한 label(정답)이 주어지지 않은 상태에서 컴퓨터를 학습 시키는 것

- 강화학습(reinforcement learning)
  현재 상태에서 어떤 행동을 취하는 것이 최적인지를 컴퓨터에 학습시키는 것


Machine Learning
Unsupervised Learning
Dimensionality Reduction
Meaningful Compression
Structure Discovery
Big data Visualistaion
Feature Elicitation
Clustering
Recommender Systems
Targetted Marketing
Customer Segmentation
Supervised Learning
Classification
Image Classification
Customer Retention
Idenity Fraud Detection
Diagnostics
Regression
Advertising Popularity Prediction
Weather Forecasting
Market Forecasting
Estimating life expectancy
Population Growth Prediction
Reinforcement Learning
Real-time decisions
Game AI
Robot Navigation
Skill Acquisition
Learning Tasks

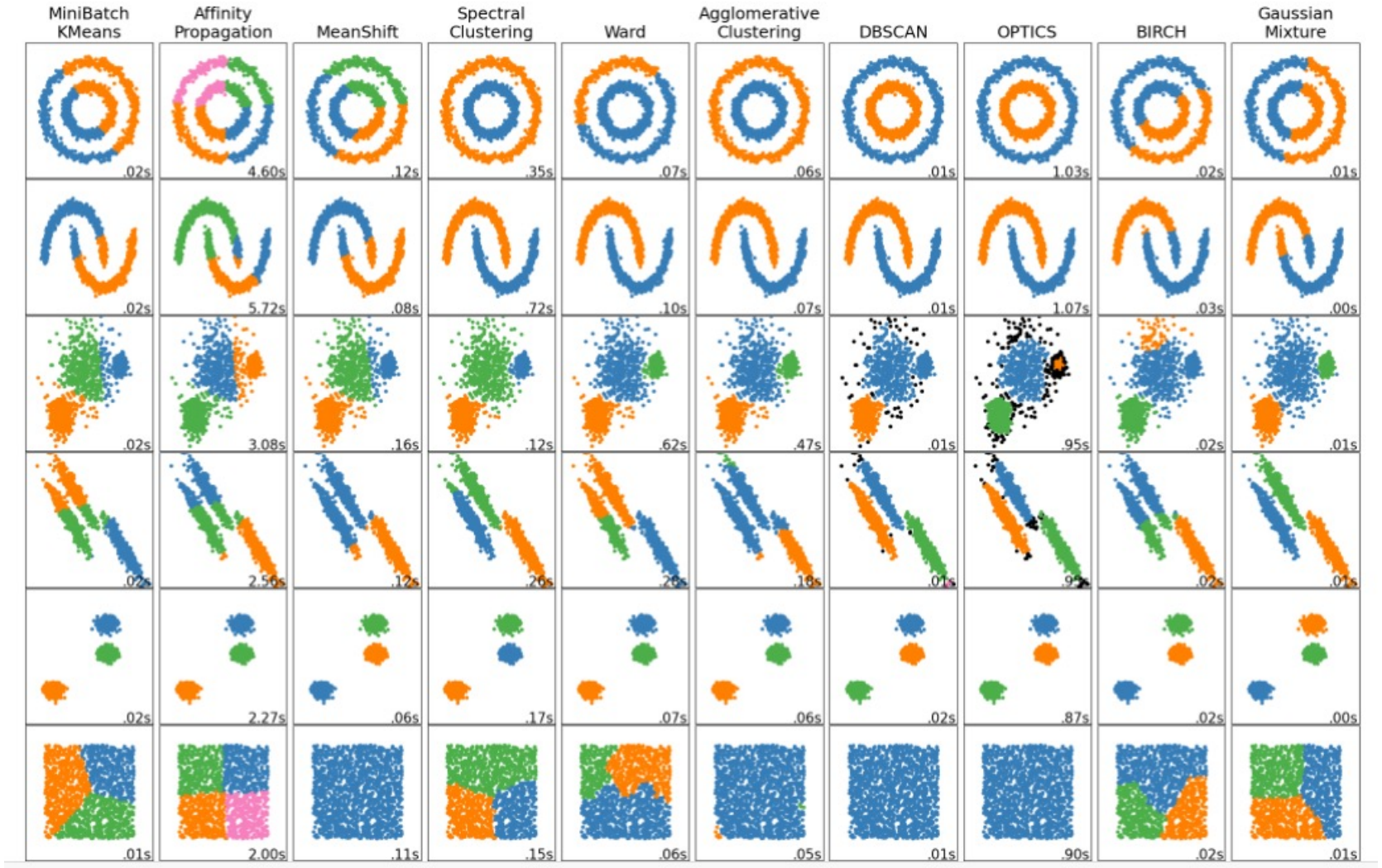
python™

비지도학습

- 데이터에 대한 label(정답)이 주어지지 않은 상태에서 컴퓨터를 학습 시키는 것
  기계가 알아서 데이터들의 특징을 가지고 유사성을 판별함
- 데이터의 숨겨진 특징이나 구조를 발견하는 데에 사용되는 알고리즘

- Clustering (군집화)
- Dimensionality Reduction(차원축소)
- Visualization(시각화)
- Association Rule Learning (연관 규칙 학습)

Clustering



←
Clustering
알고리즘의 종류

분포에 따라
적용해야하는
알고리즘이 다르다

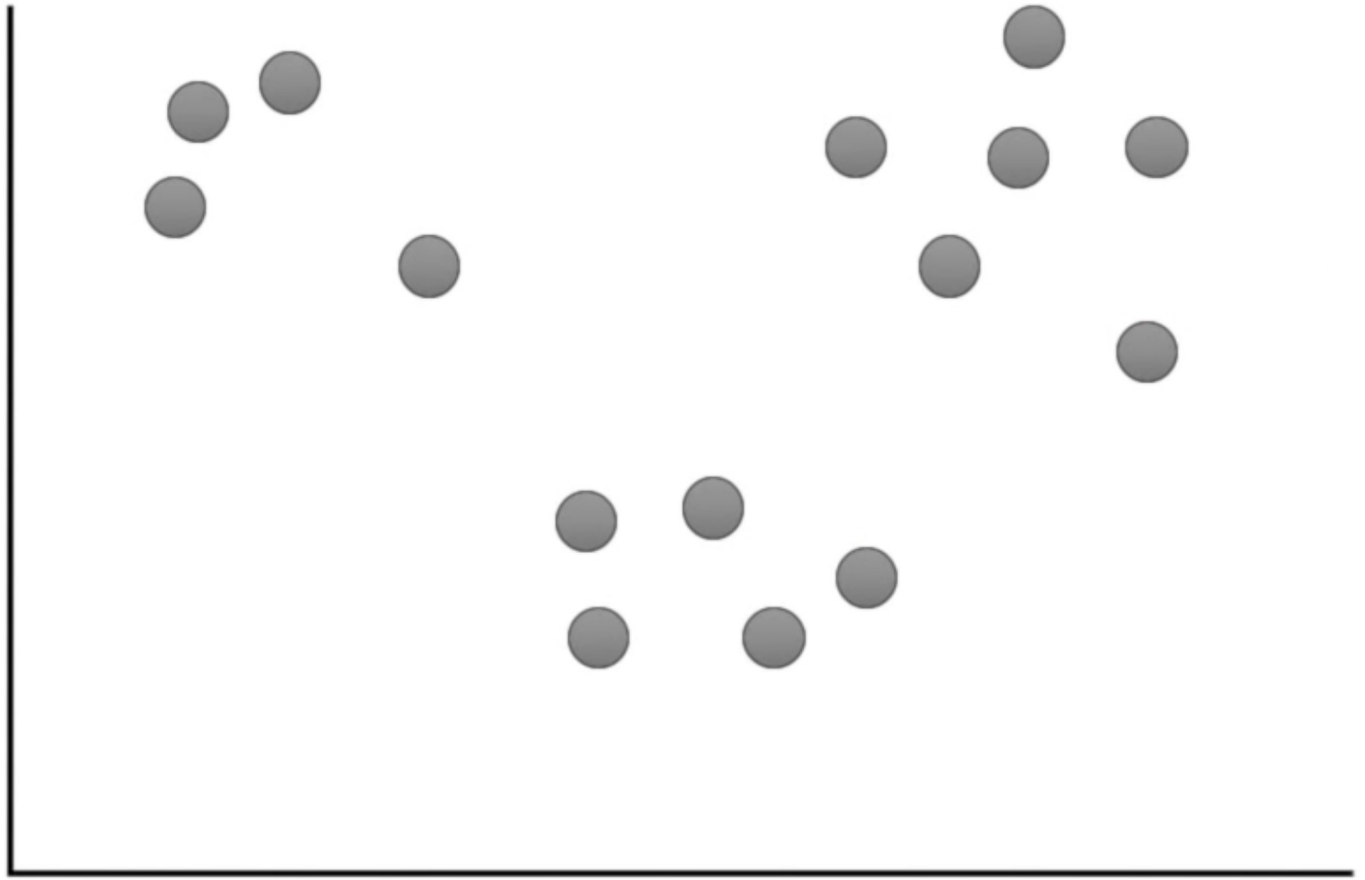
python™

K-means

- K : Cluster의 개수
- Means : Cluster의 중심과 데이터들의 평균 거리　　* cluster의 중심 : centroid

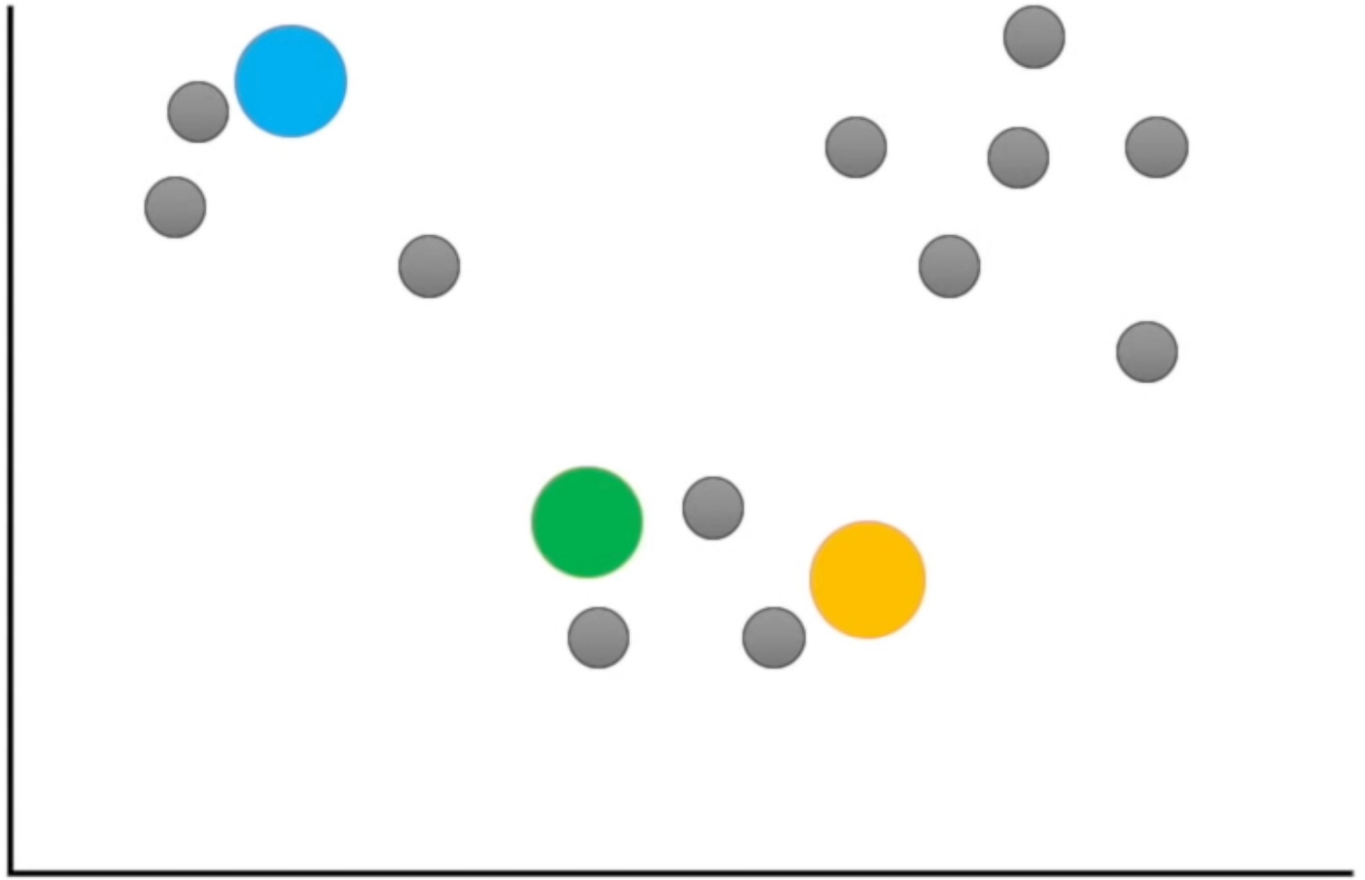
1. Data set에서 K개의 클러스터 중심(centroid)을 임의로 지정
2. 각 데이터들을 가장 가까운 centroid가 속한 그룹에 할당
3. 각 cluster에 대하여 새로운 centroid를 계산 (cluster에 속한 데이터 포인트의 평균값)
4. 2~3번 과정을 centroids가 변하지 않을 때 까지 반복

K-means

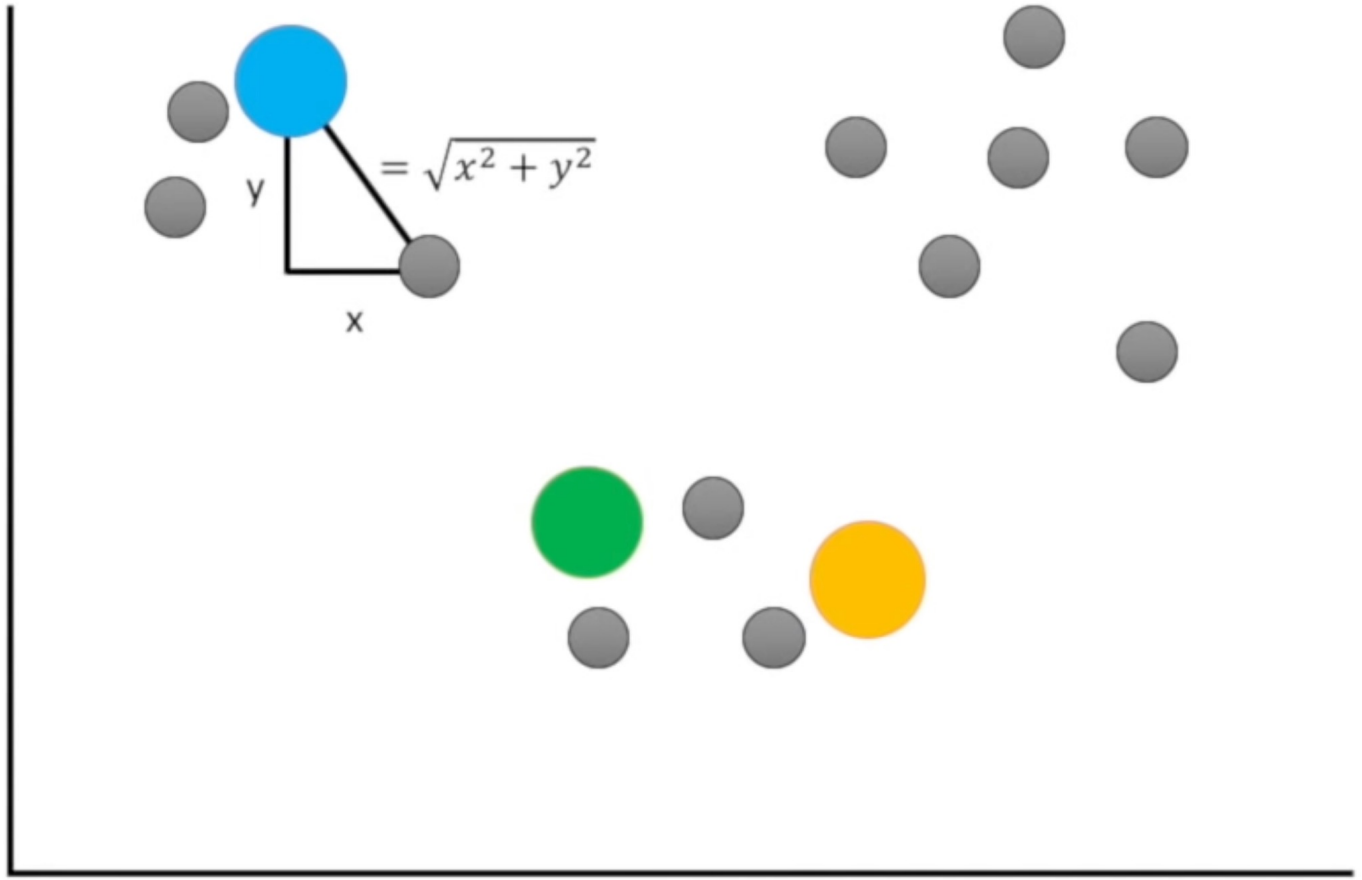


python™

K-means

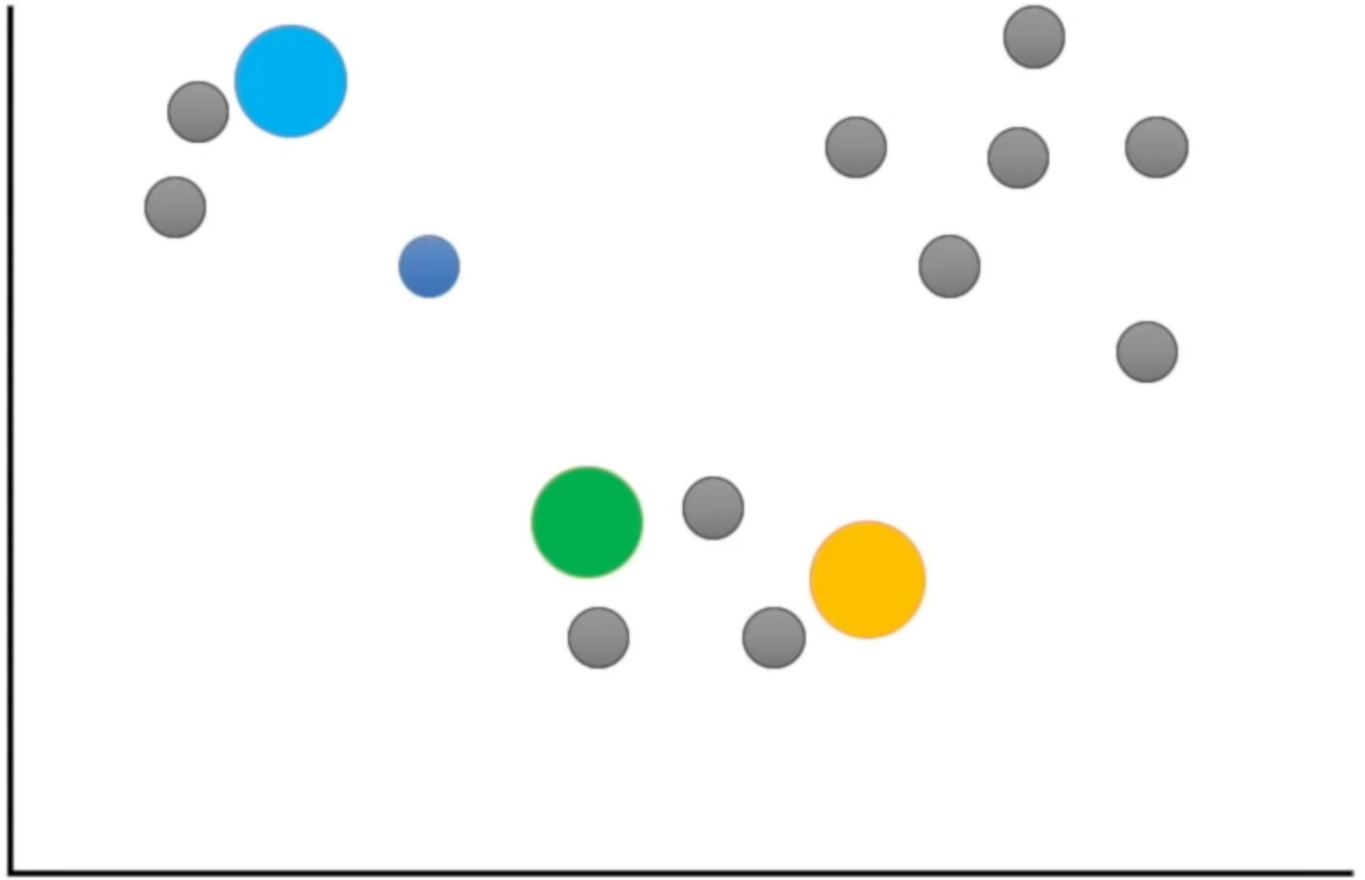


K-means



python™

K-means

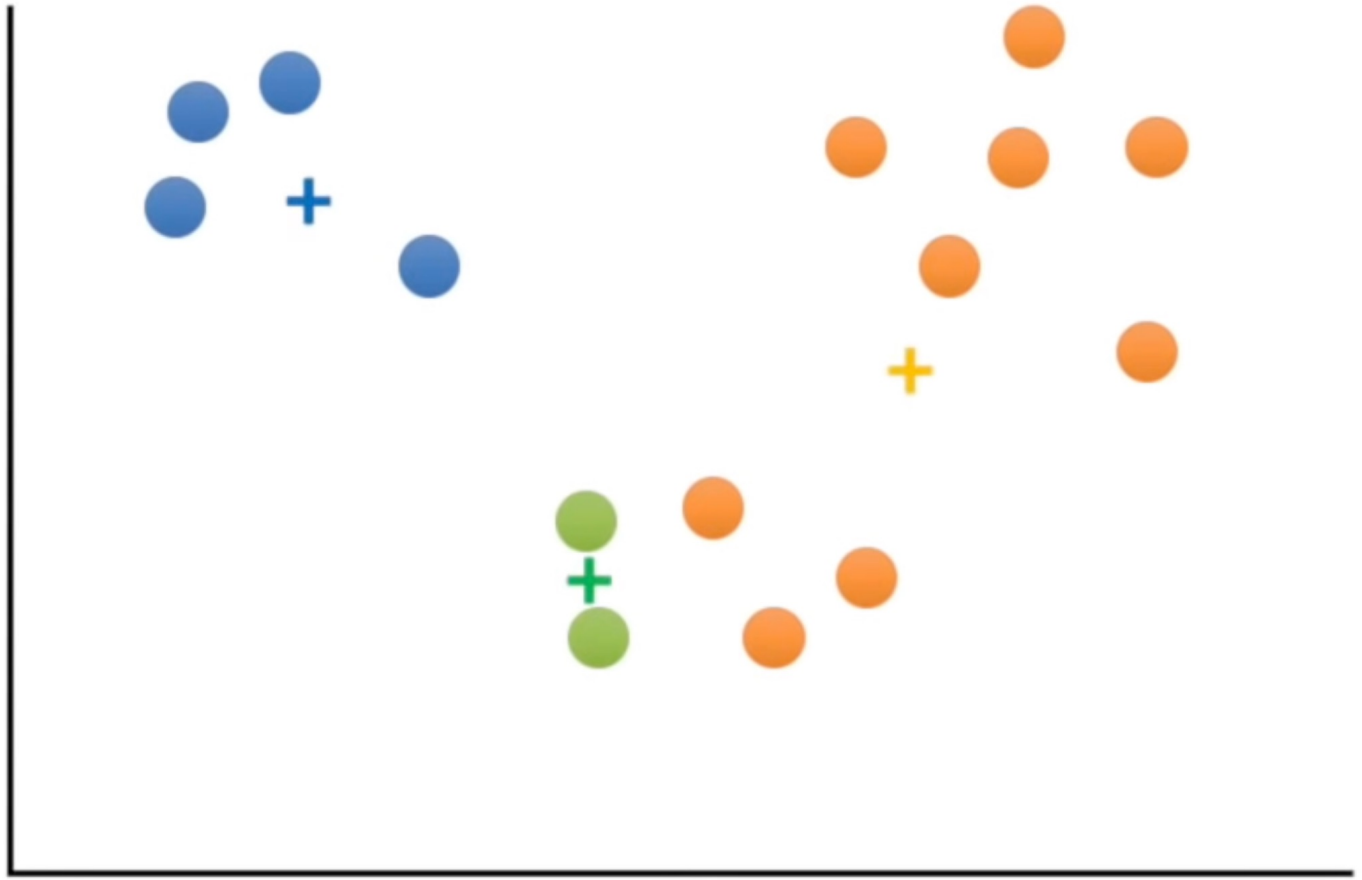


python™

K-means

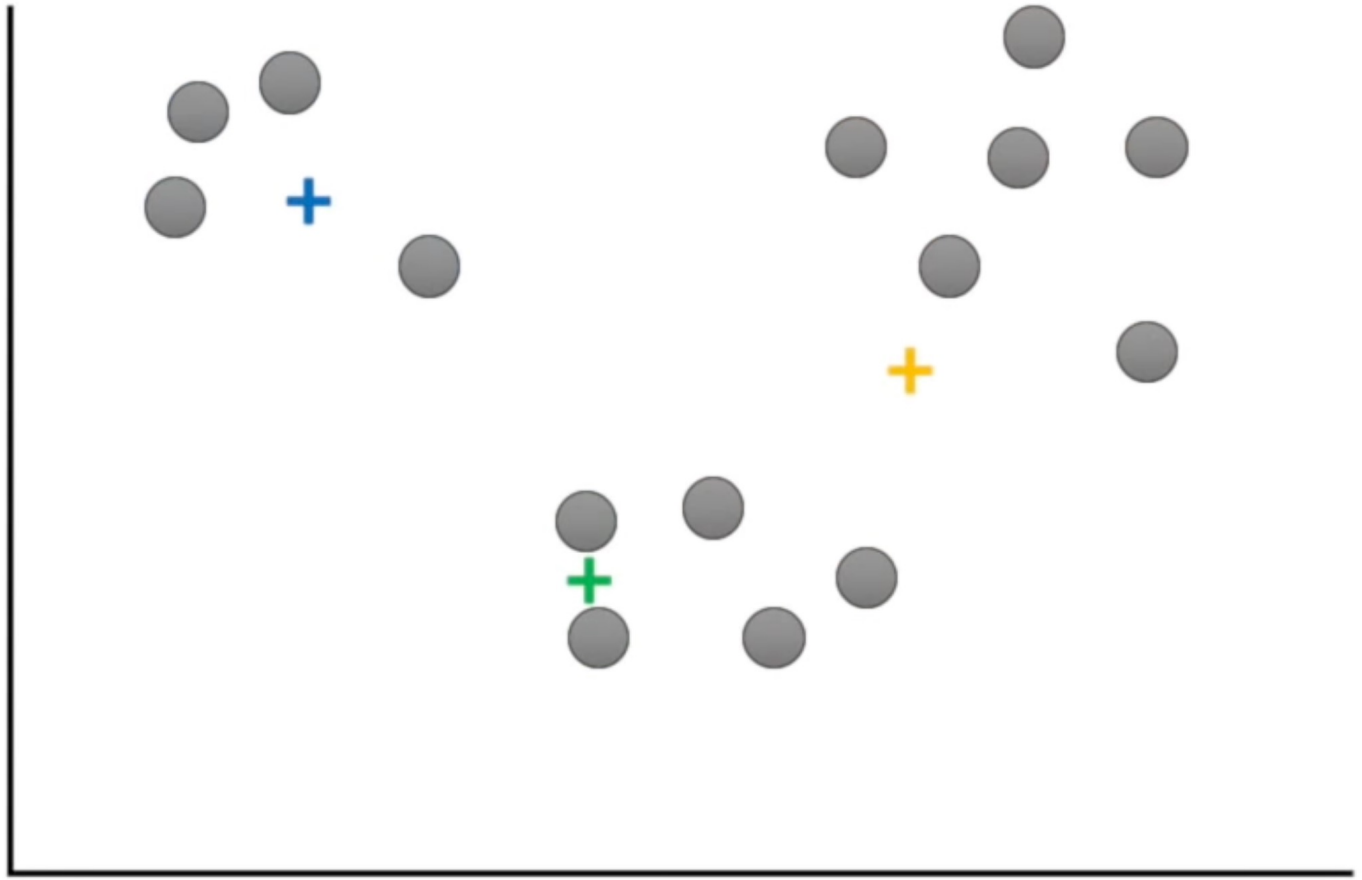
python™

K-means



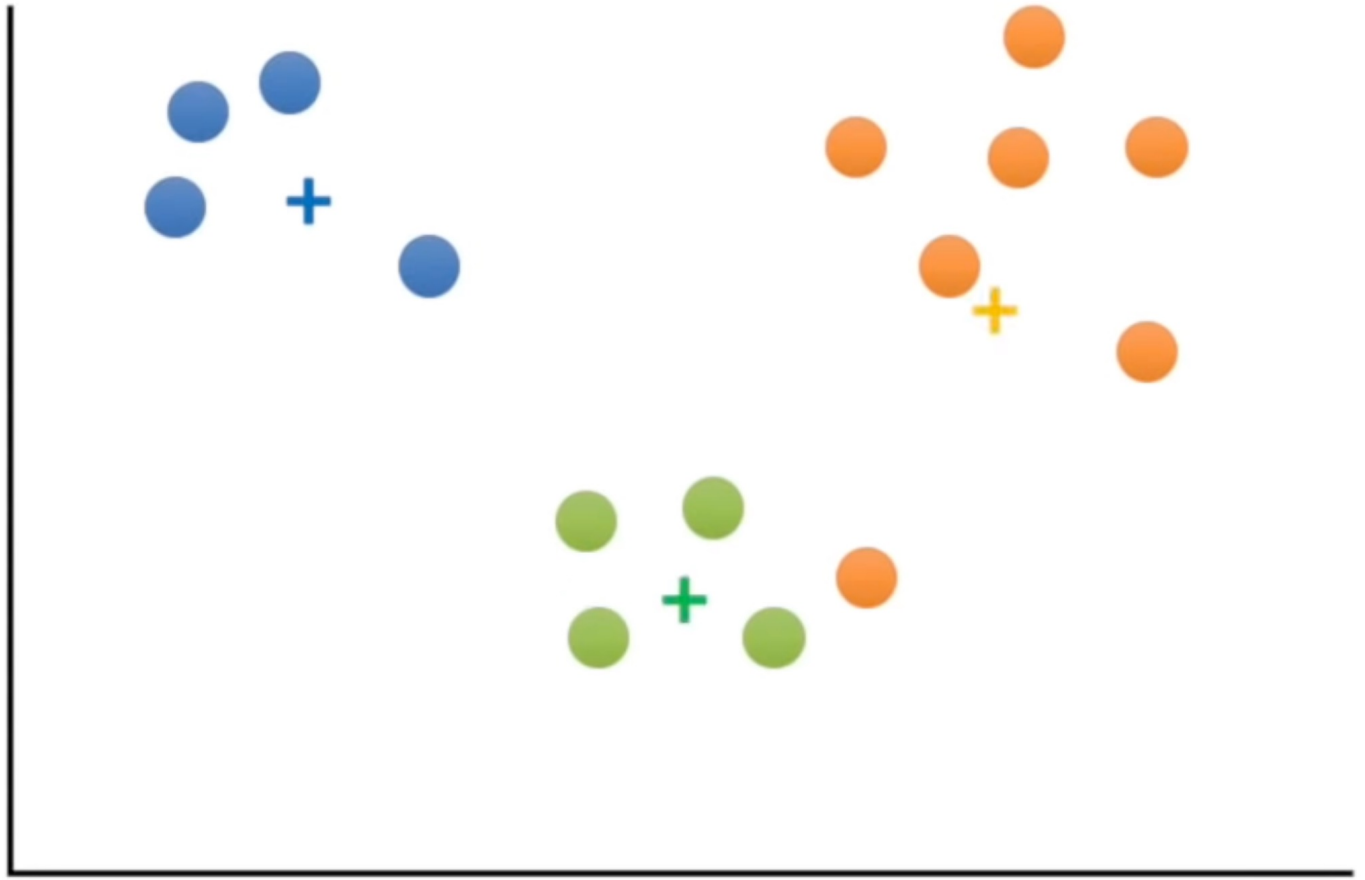
python™

K-means



python™

K-means

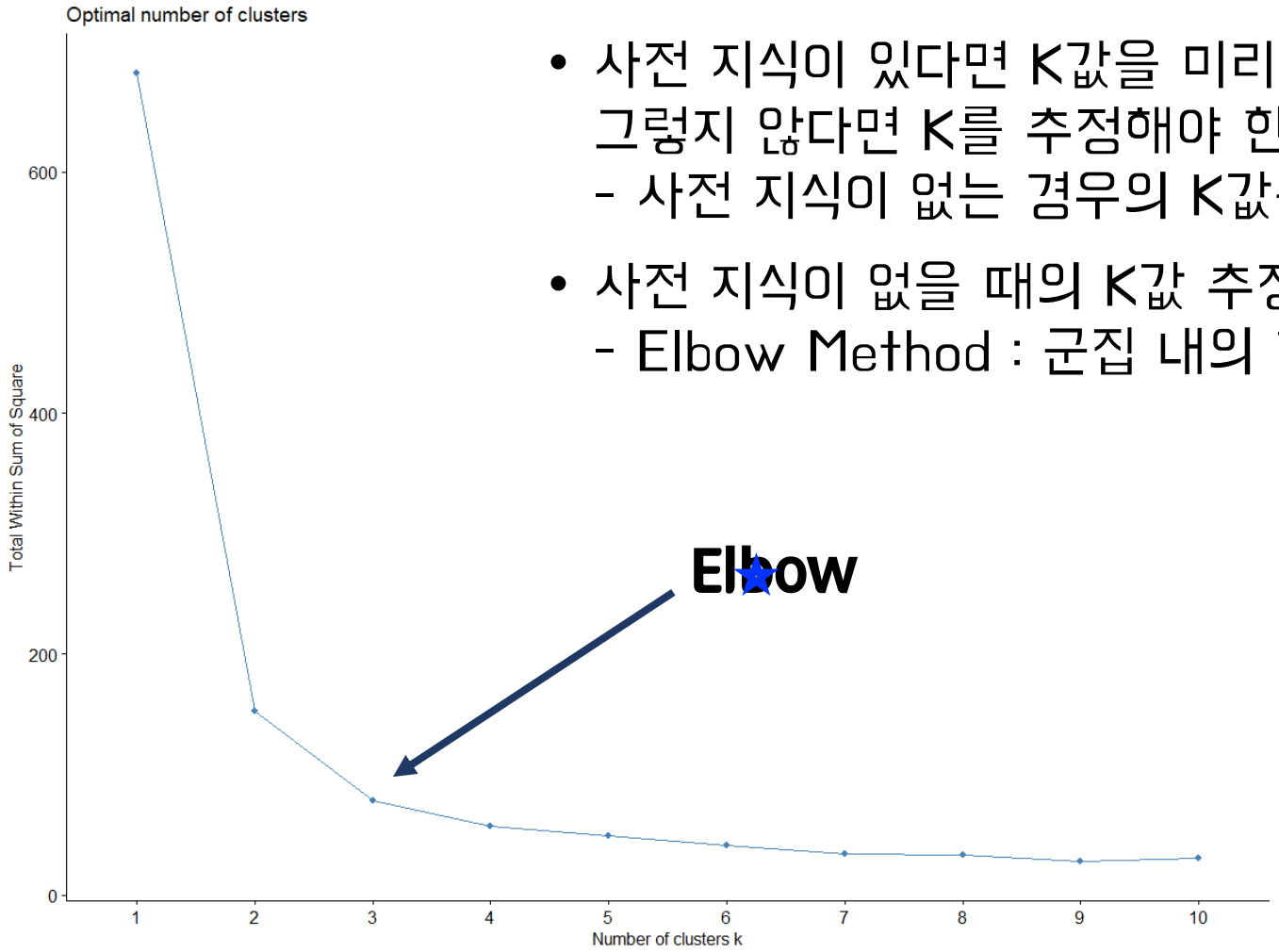


python™

K-means

python™

K-means



- 사전 지식이 있다면 K값을 미리 지정할 수 있지만,
  그렇지 않다면 K를 추정해야 한다
  - 사전 지식이 없는 경우의 K값은 2 ~ n/2 까지 가능
- 사전 지식이 없을 때의 K값 추정 : Elbow Method를 이용
  - Elbow Method : 군집 내의 제곱합이 눈에 띄게 작아지는 점

python™

K-means의 장단점

- 장점
  - 간단한 기준으로 clustering을 진행
  - 유연하고 효율적

- 단점
  - 계산이 단순하기 때문에 underfitting 발생 가능
  - 운이 없다면 최적의 cluster가 되지 않을 수 있음
  - K값을 미리 알고 있어야 함

DBSCAN

- Density-Based Spatial Clustering of Applications with Noise
  밀도 기반 데이터 클러스터링 알고리즘

- 밀도 기반 이란?
  cluster를 찾을 때, **데이터 포인터들이 밀도 있게 모여 있는 곳**을 찾는 알고리즘
  = 서로 인접한 데이터들은 같은 cluster 안에 있음

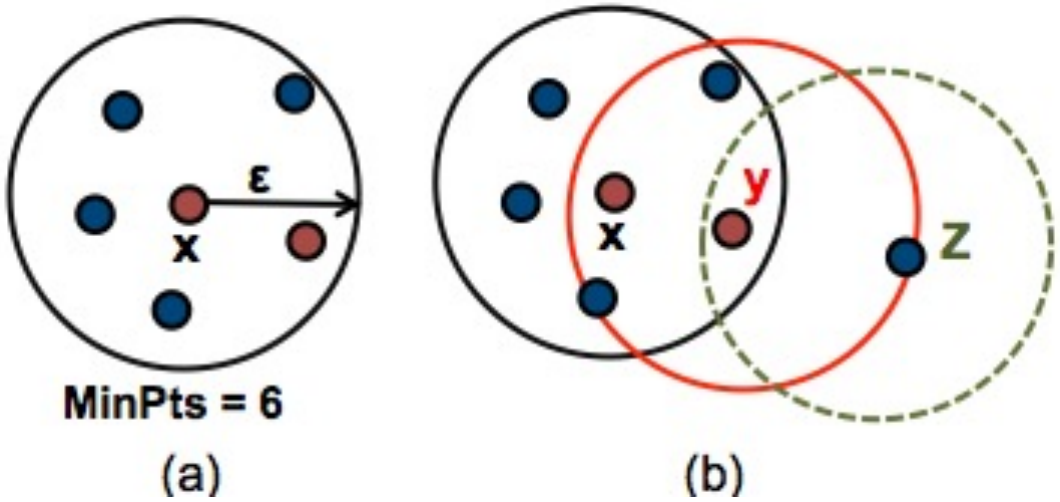
- K-means와의 차이점
  DBSCAN : 서로 인접한 데이터들이 같은 cluster 안에 있음
  K-means : cluster의 중심지와 가까운 데이터들이 같은 cluster로 할당됨

DBSCAN

- 파라미터
  - ε(epsilon) : 데이터로부터의 반경
  - minPts : cluster을 구성하는 데에 필요한 최소 데이터의 개수

- 분류 포인트
  - core point : 한 점의 ε 반경 내에 minPts 이상의 개체가 포함된 점
  - border point : 한 점의 ε 반경 내에 mitPts 보다 적은 개체를 포함하고 있지만, 적어도 하나의 core point의 반경에 속하는 점
  - noise point : core point 또는 border point가 아닌 점 ε 반경 내에 minPts보다 적은 수의 개체를 포함 하는 점

DBSCAN

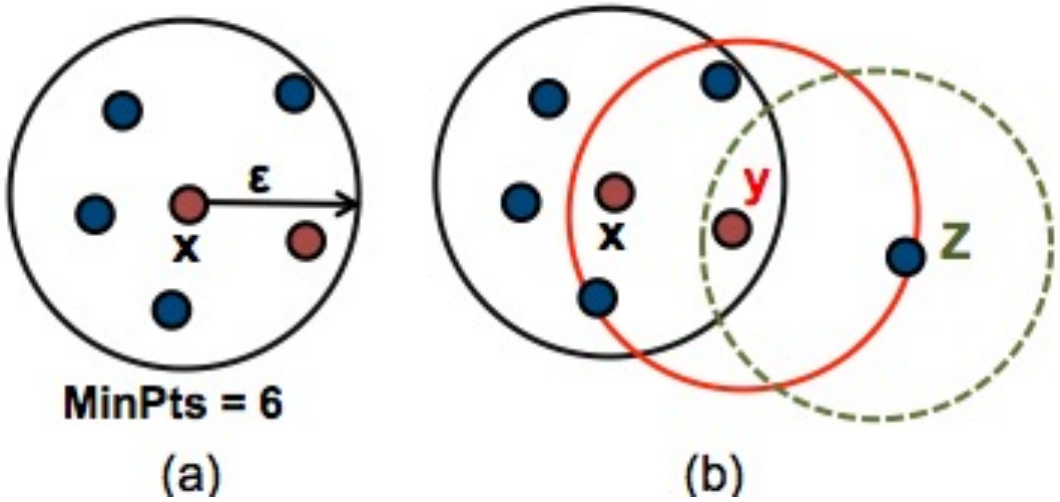


- 파라미터
  - ε
  - minPts

- 분류 포인트
  - core point
  - border point
  - noise point

python™

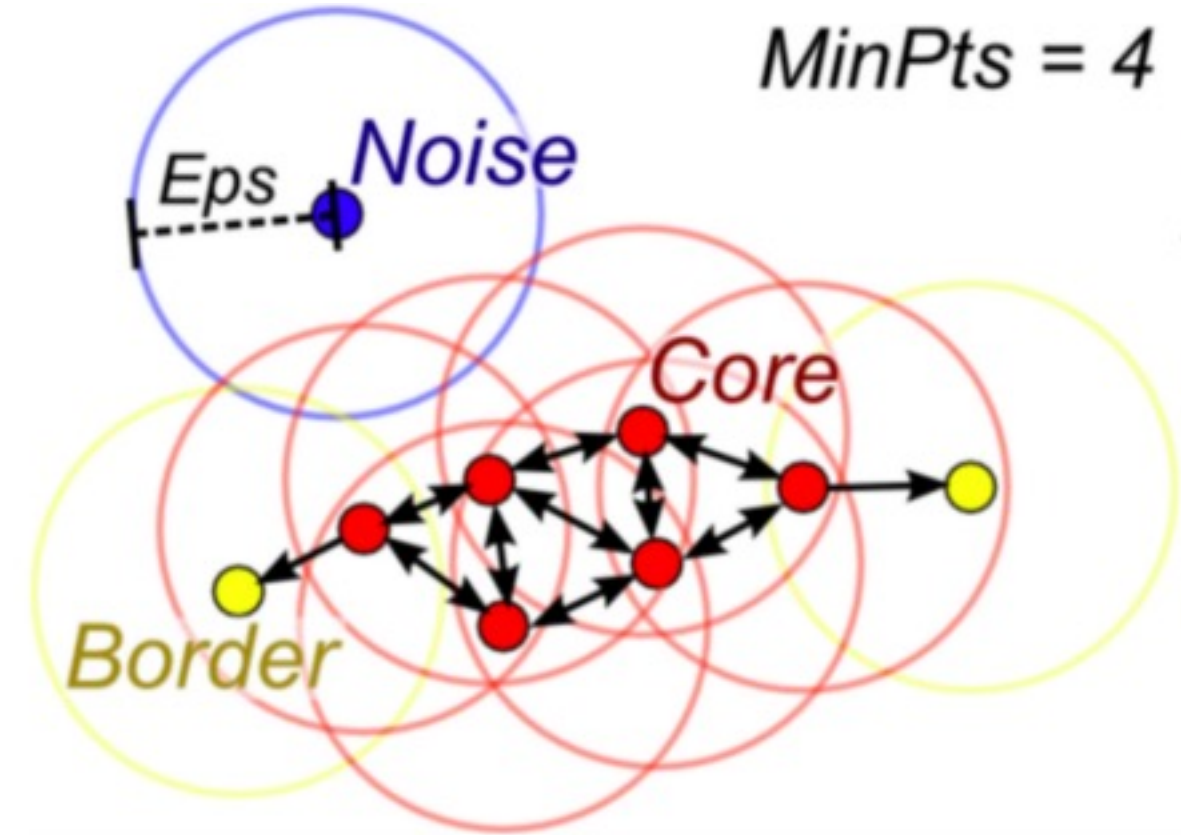
DBSCAN



- 파라미터
  - ε
  - minPts

- 분류 포인트
  - core point x
  - border point y
  - noise point z

DBSCAN



Red: Core Points

Yellow: Border points. Still part of the cluster because it's within epsilon of a core point, but not does not meet the min_points criteria

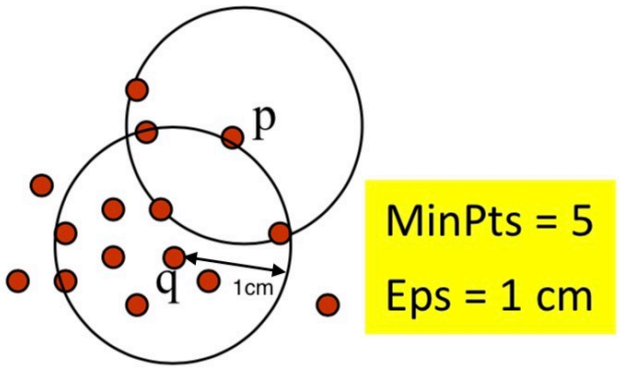
Blue: Noise point. Not assigned to a cluster

python™

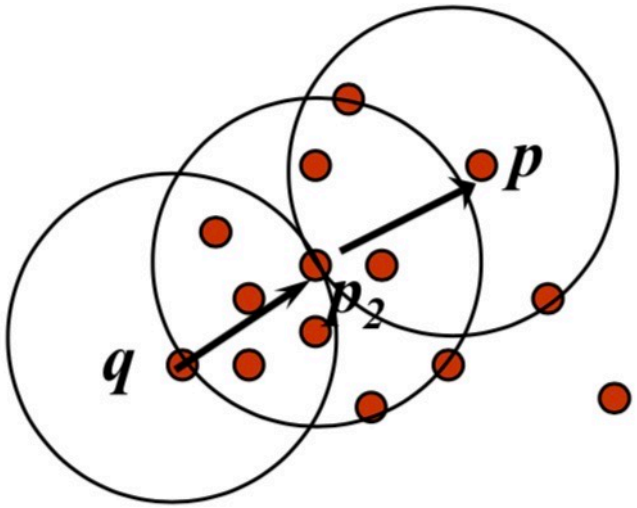
DBSCAN

- 직접 밀도-도달 가능
  점 p가 점 q의 반경 안에 들어오고 점 q가 코어 점일 때,
  “점 p가 점 q로부터 직접 밀도-도달 가능한 관계에 있다”
- 밀도-도달 가능
  점 p와 점 q 사이에 $p_1,\ p_2, \ldots, p_n\ (p_1 = q, p_n = p)$들이 있고,
  점 p_{i+1}이 점 p_i로부터 직접 밀도-도달 가능하다면,
  “점 p는 q로부터 밀도-도달 가능 관계에 있다”
- 밀도-연결
  어떤 점 O로부터 다른 두 점 p, q가 반경 내 minPts 조건 하에 밀도-도달 가능하다면,
  “점 p와 q는 O를 통해 서로 연결되어 있다”

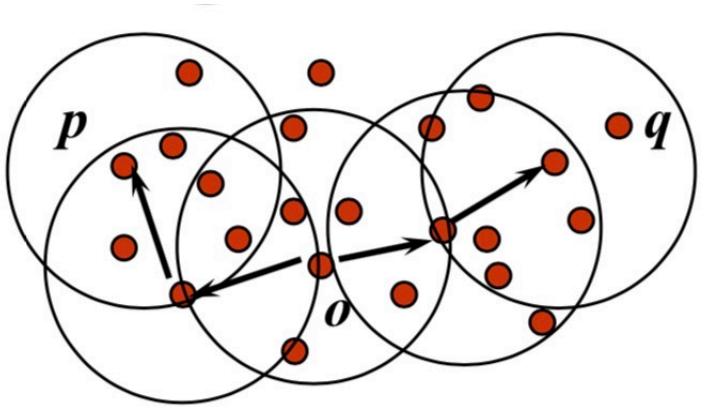
DBSCAN



직접 밀도-도달 가능



밀도-도달 가능



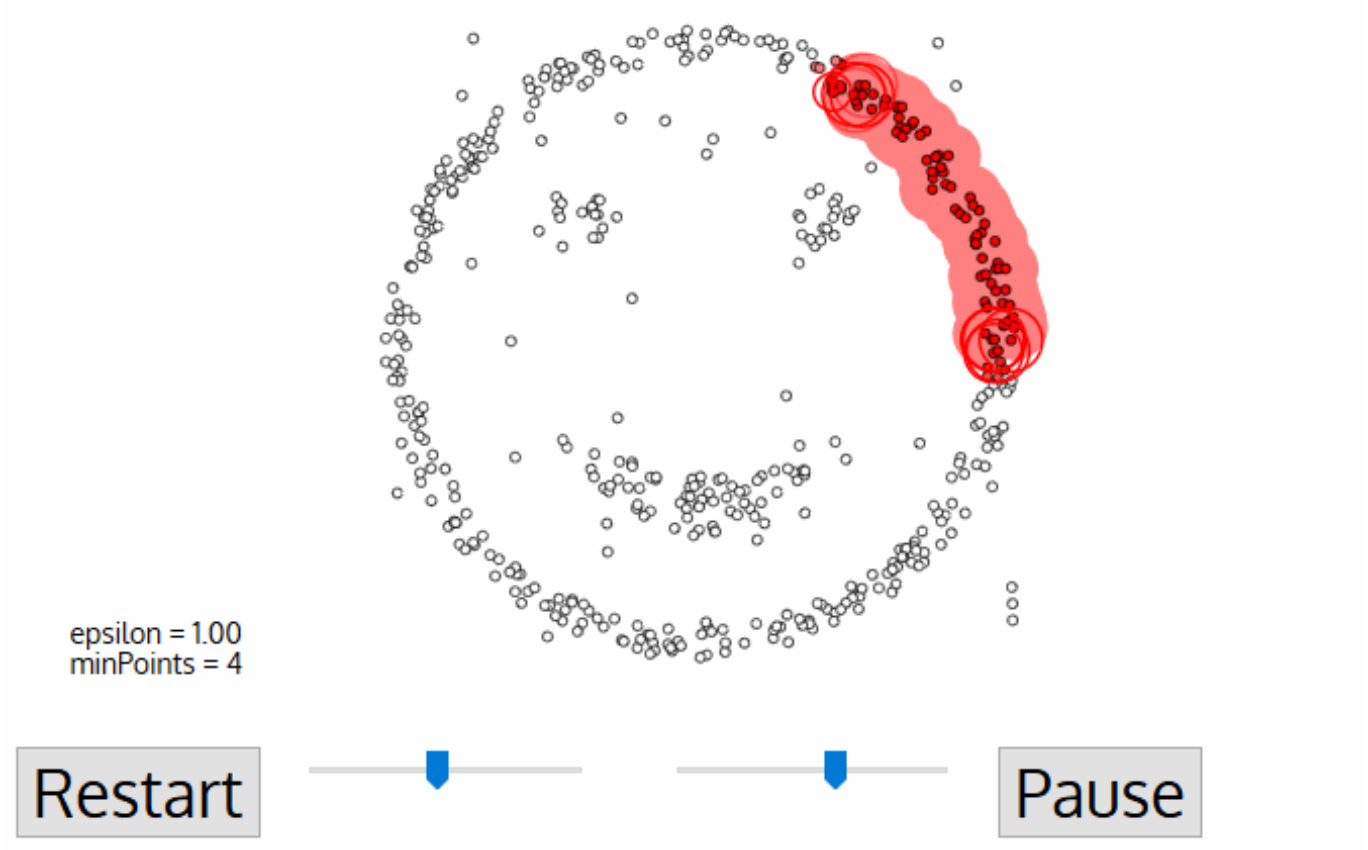
밀도-연결

python™

DBSCAN

1. ε과 minPts를 설정
2. 데이터로부터 core point를 만족하는 임의의 점 선택
3. 밀도-도달 가능한 점들을 뽑아 core point와 border point를 구분,
   이에 속하지 않는 점들은 noise point로 분류
4. ε반경 안에 있는 core point들을 서로 연결
5. 연결된 core point를 통해 하나의 cluster를 형성
6. 모든 border point들은 어느 하나의 cluster로 할당
   (여러 cluster에 걸쳐 있는 경우, 반복 과정에서 먼저 할당된 cluster로 할당)

DBSCAN

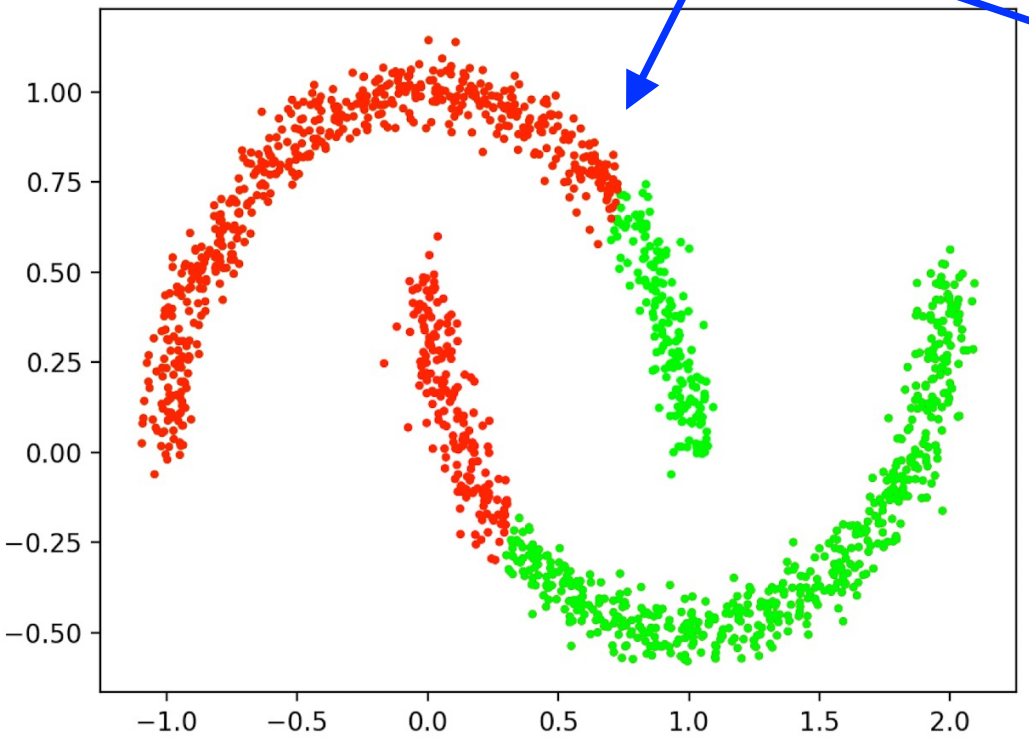


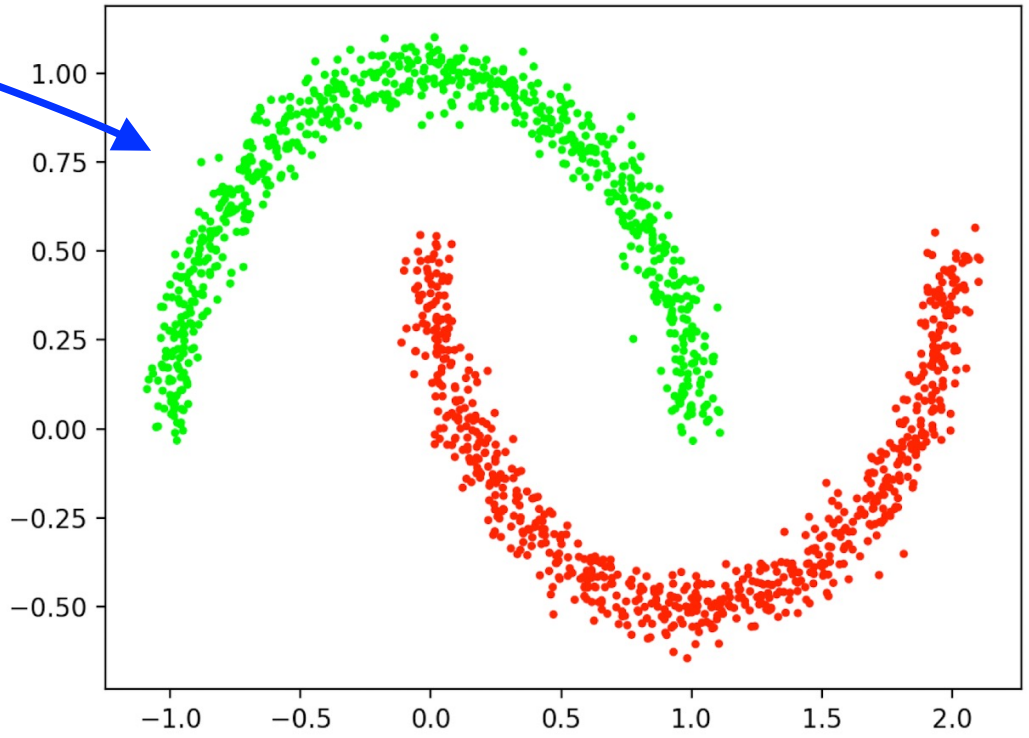
python™

DBSCAN의 장단점

- 장점
  - non_linear한 clustering도 가능
  - cluster의 개수를 지정하지 않아도 됨
  - outlier에 크게 민감하지 않음
  - 시작 점에 의존성이 크기 않음
  - 속도가 빠르다
- 단점
  - 결과 값이 일정하지 않고 다양하게 나타난다
  - 여러 개의 cluster에 속하는 점들은 애매하게 분류가 된다
  - 유클리드 거리에 의존하여, 차원이 높아질 수록 적당한 ε을 찾기가 어렵다
  - 듬성듬성한 데이터에 대해서는 cluster 결과가 좋지 못하다
  - Normalizing의 의존성이 높다

DBSCAN vs K-means





python™

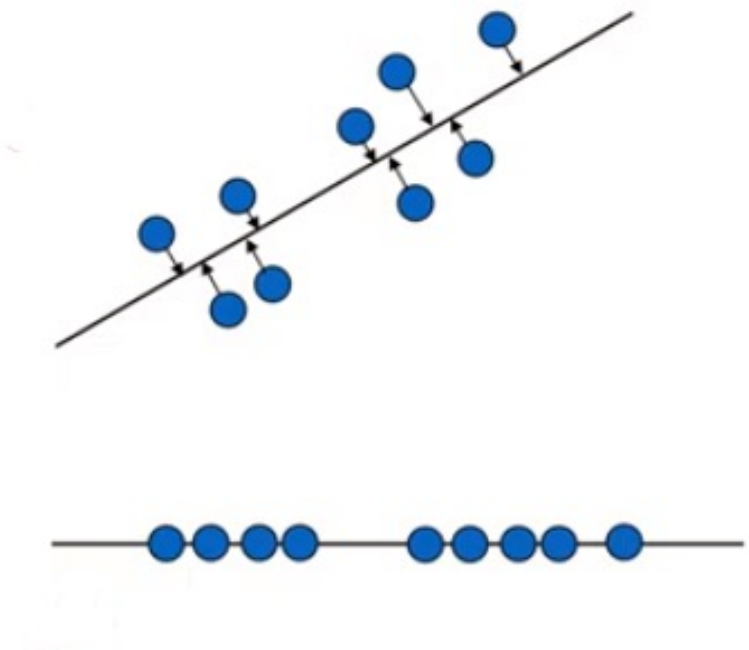
PCA

- Principle Component Analysis (주성분분석)
  가장 널리 사용되는 차원 축소 기법 중 하나
- 각 데이터들의 feature들의 기여도를 분석하여 높은 feature만 추출하는 방법
  차원을 축소하지만, 손실되는 정보는 최대한 적게 하는 것이 이 알고리즘의 목적이다
- 예를 들어, 200개의 feature가 있는 데이터가 있을 때, 이 중에서 10개의 feature가
  원본의 대부분을 설명한다면 10개의 feature만 이용하여 cluster를 실시해도 충분하다

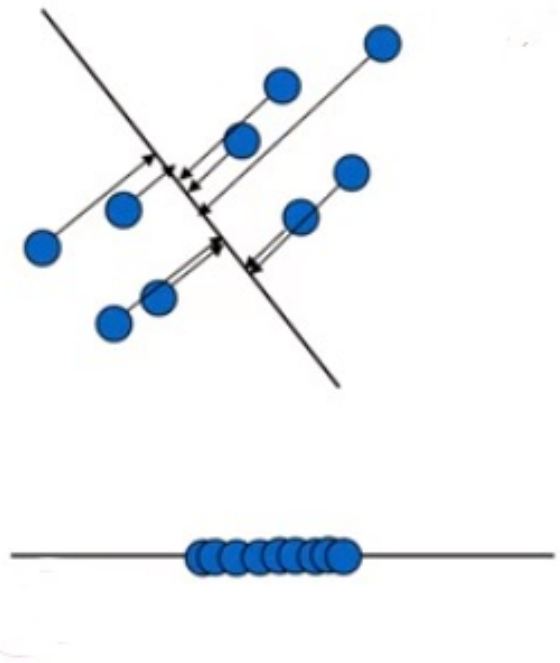
- 계산 방법이 있으나... 생략... ~~(꽤나 복잡하기 때문)~~
  - covariance matrix에서 계산한 eigenvalue를 활용한다는 정도로만 알고 있자...

PCA

Find the new axis that
maximizes the variance of data

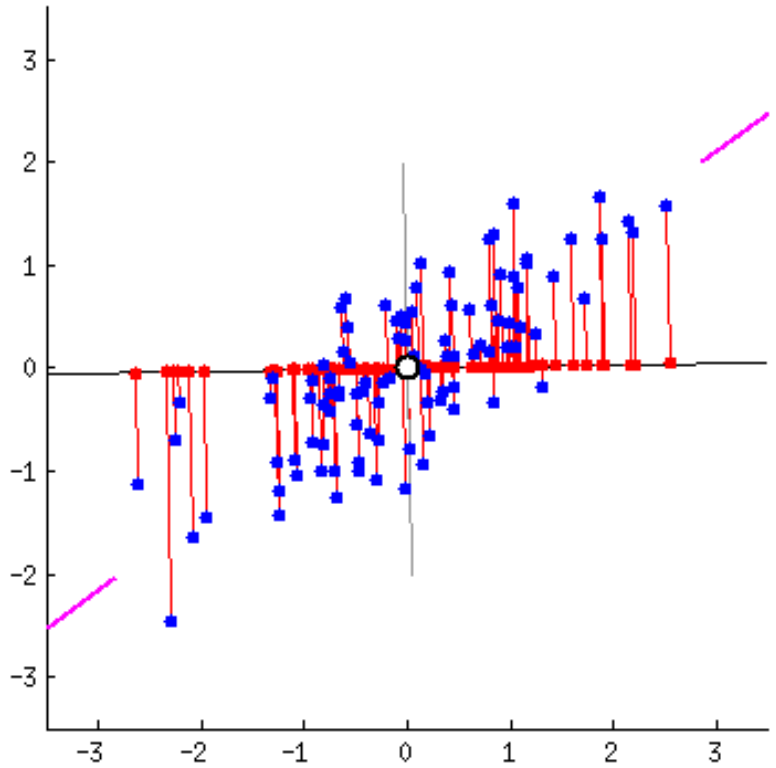


Find the new axis that
minimizes the variance of data



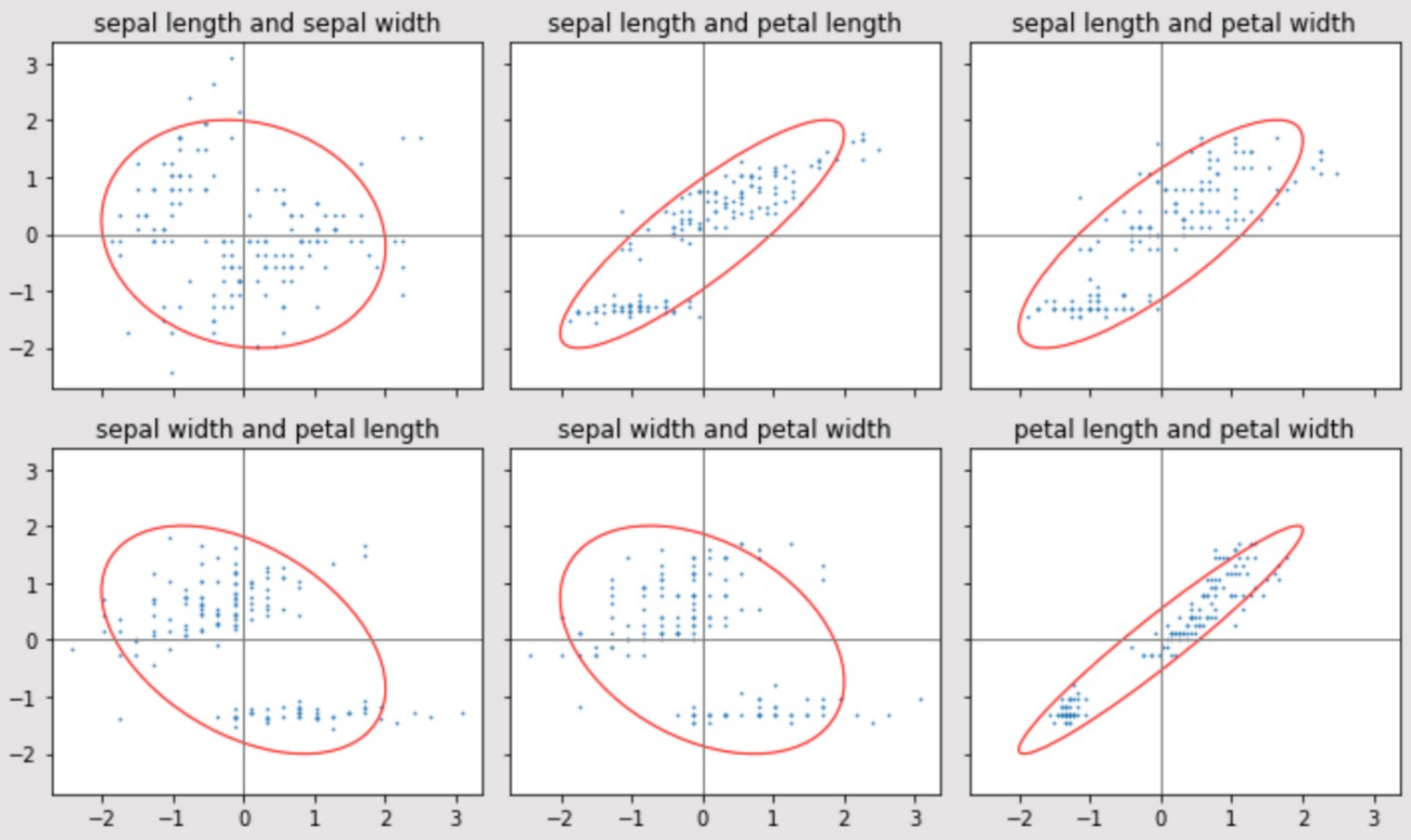
python™

PCA



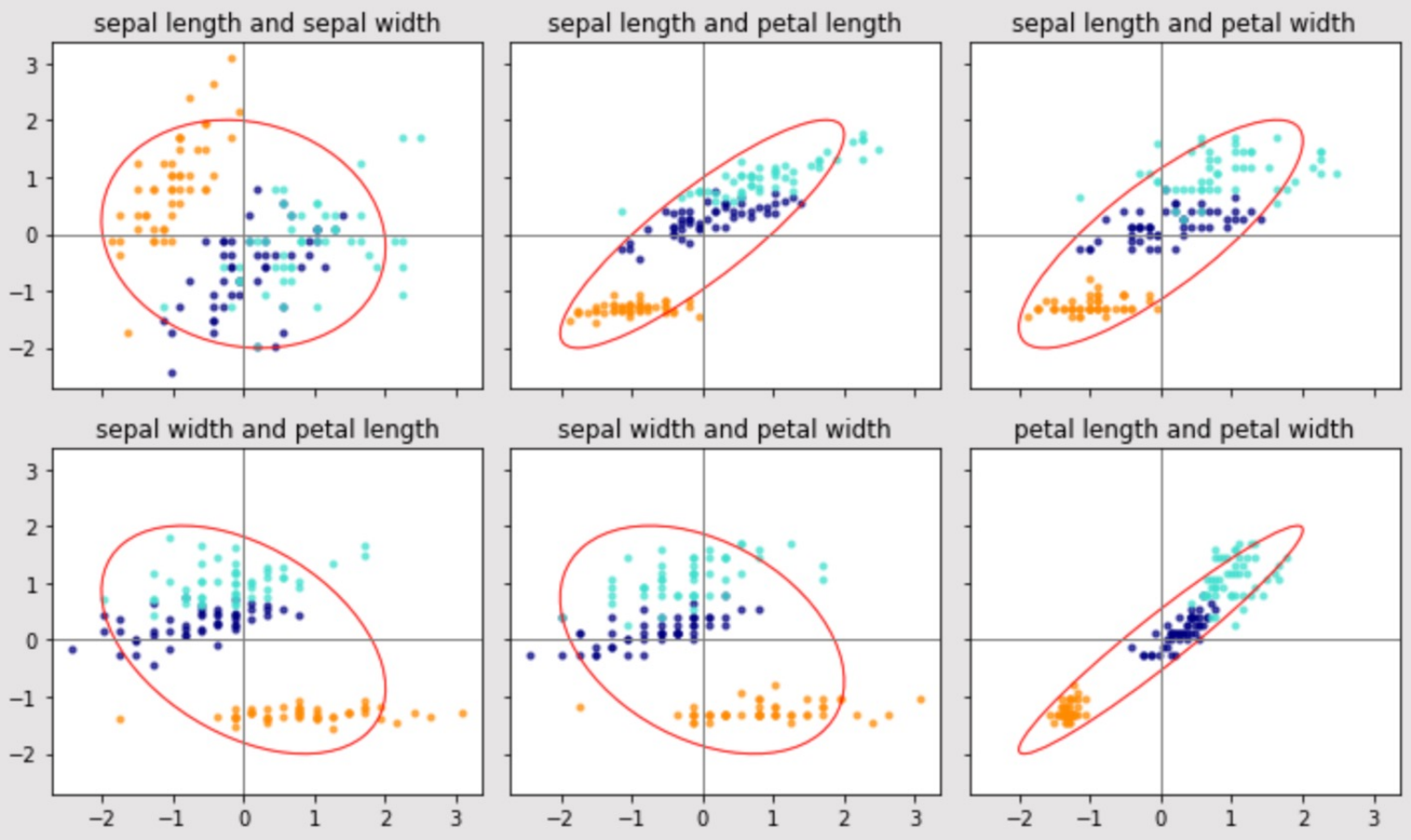
python™

PCA


sepal length and sepal width
sepal length and petal length
sepal length and petal width
sepal width and petal length
sepal width and petal width
petal length and petal width
3
2
1
0
−1
−2
−2
−1
0
1
2
3

python™

PCA


sepal length and sepal width
sepal length and petal length
sepal length and petal width
sepal width and petal length
sepal width and petal width
petal length and petal width

python™

PCA의 장단점

- 장점
  - 고차원의 데이터를 저차원으로 축소하기 때문에 효율적임

- 단점
  - 때에 따라서 중요한 정보가 손실될 때가 있다
  - 비선형 데이터에서는 적절하지 못하다

python™