클러스터링 과제

과제1:
K-Means를 구현해주세요!

알고리즘 6-1 ***k*-평균**

입력: 훈련집합 $\mathbb{X} = \{\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_n\}$, 군집의 개수 k
출력: 군집집합 $C = \{c_1, c_2, \cdots, c_k\}$

```
1  k개의 군집 중심 Z = {z_1, z_2, ..., z_k}를 초기화한다.
2  while (true)
3     for (i=1 to n)
4           x_i를 가장 가까운 군집 중심에 배정한다.
5     if (라인 3~4에서 이루어진 배정이 이전 루프에서의 배정과 같으면) break
6     for (j=1 to k)
7           z_j에 배정된 샘플의 평균으로 z_j를 대치한다.
8  for (j=1 to k)
9     z_j에 배정된 샘플을 c_j에 대입한다.
```

10기 분들은
K-Methoid 혹은 다중 시작 k-means 구현을
과제로 받겠습니다!

R,Python 상관없습니다

- 힌트-

```r
# 과제 k-means 구현하기
# k-means 구현 과제


rm(list=ls())

data <- iris[,1:4] # 라벨 제거한 iris 데이터를 쓸게요!
str(data)
head(data)

k_means<-function(data,k) # data랑 군집 몇개로 나눌지 k를 받아서
{
  # 처음에는 먼저 Forgy 방식(랜덤으로 k개 선택)으로 centroid(중심점) 설정해줘야 겠죠!

  # centroid가 뭔지 표시나 저장해주는 수단이 필요할거에요!
  # 행렬이나 데이터프레임을 만들어서 centroid만 따로 저장해주거나 /
  # 변수를 하나 더 만들어서 true/false로 centroid가 뭔지 표시해줄수도 있겠죠! 방법은 다양하니 각자 생각해보기!

  while이나 for문(반복문) # centroid가 바뀌지 않거나 일정한 반복횟수 이상동안 하도록 하면 되겠죠!
    {
      # 1. 각각의 데이터에대해 k개의 centroid 각각과의 거리를 구해서 (유클리드) (dist함수 쓰면 편하겠죠)
      # 2. 가장 가까운 거리의 centroid의 군집으로 할당시키고
      # 3. 다시 각각의 k개의 군집마다 새로운 centroid가 구해지겠죠!
  }
  return(data$label) # 그래서 최종적으로 label을 출력하도록 !
}
```

과제 2: 앙상블 과제에 직접 구현한 k-means를 이용한 변수 반영.

- K-means를 구현하고 본인의 알고리즘에 앙상블 과제 데이터를 넣어서 변수를 넣어주세요!!