R 데이터 분석 입문

Chapter 13

데이터 마이닝 기초 (2)

오 세 종

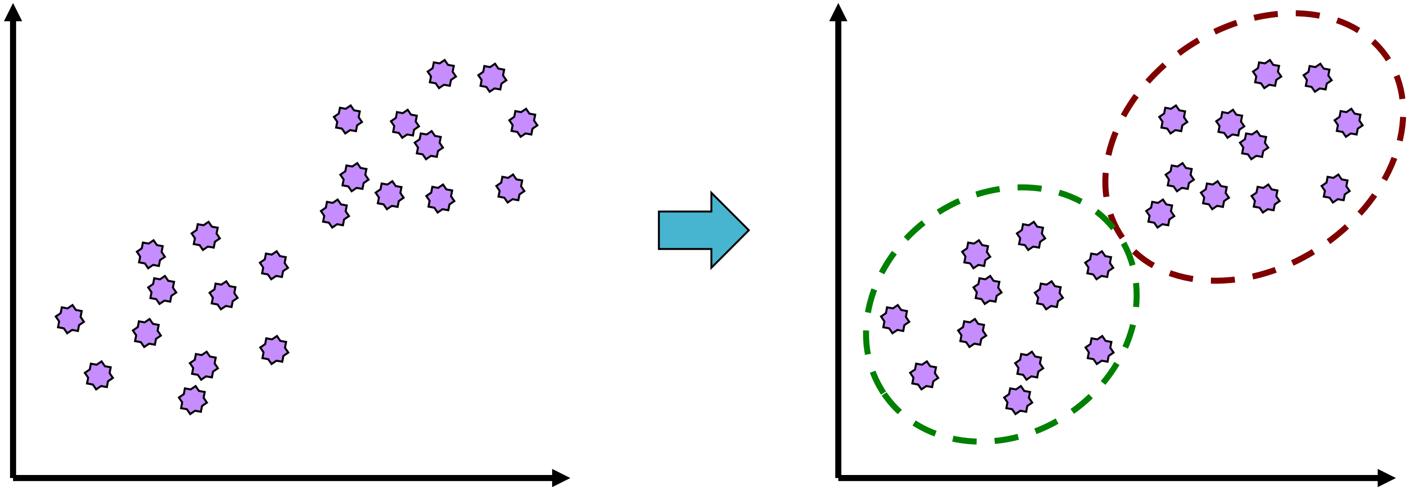
DKU DANKOOK UNIVERSITY

Contents

1. 군집화, 분류
2. k-means clustering
3. KNN classification
4. k-fold cross validation

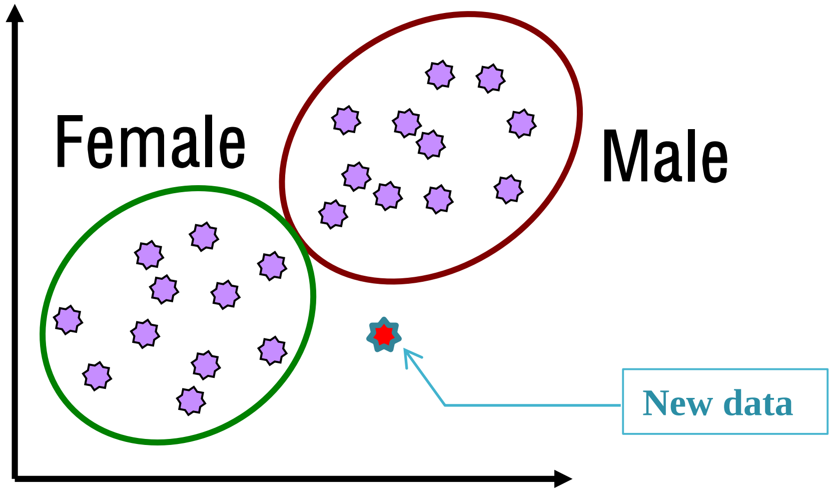
1. 군집화, 분류

- 군집화(Clustering)
  - Grouping target data into some category (class)
  - Data in same group has similar characteristics
  - Group points into clusters based on how “near” they are to one another
  - 비지도 학습 (Unsupervised learning)



1. 군집화, 분류

- 분류(Classification)
  - Classify new data into one of known category.
  - The category has “label”
  - 현실에서는 예측(prediction)문제에 적용
  - 지도 학습 (Supervised learning)



1. 군집화, 분류

- 군집화 예제
  - 차량의 특성을 가지고 grouping 을 해 보자

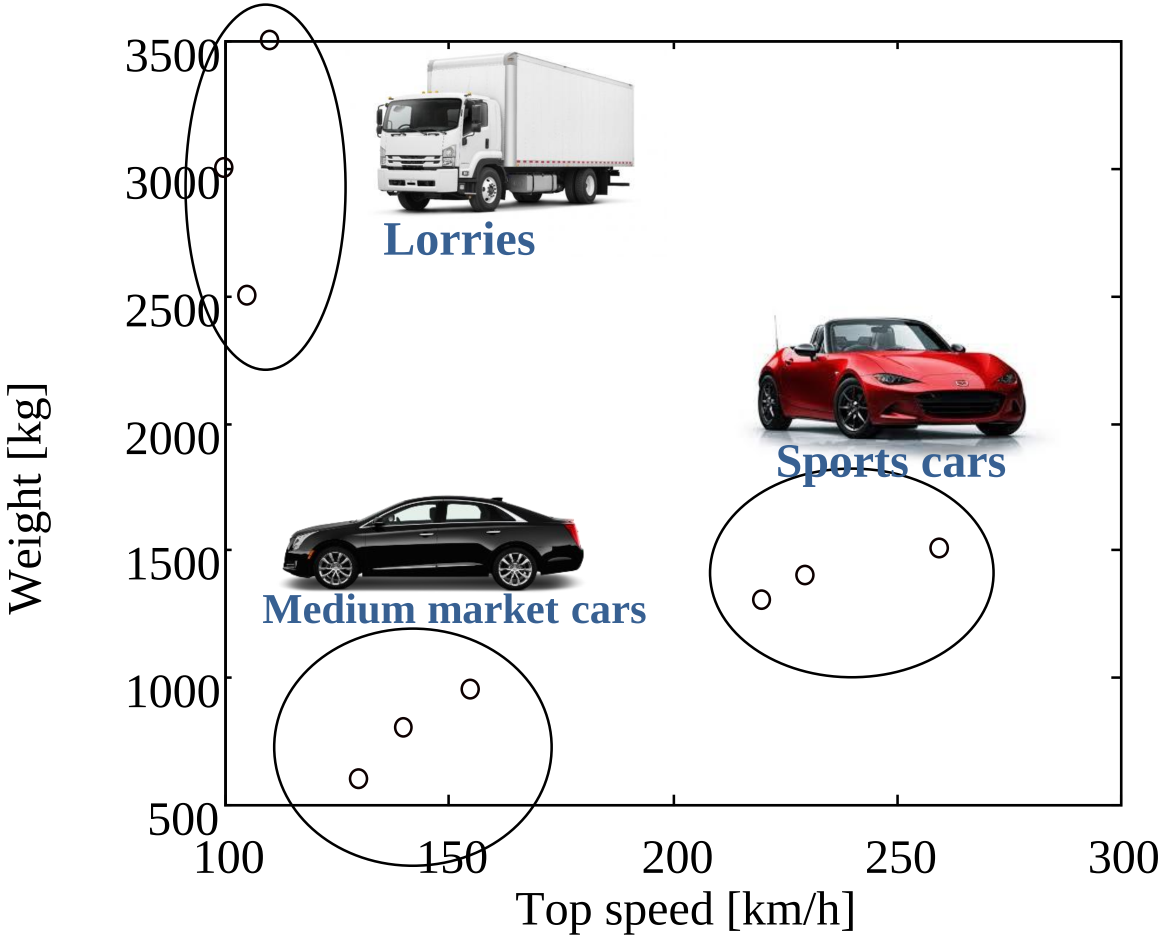
그룹이 보이는가?

Vehicle	Top speed km/h	Color	Air resistance	Weight Kg
V1	220	red	0.30	1300
V2	230	black	0.32	1400
V3	260	red	0.29	1500
V4	140	gray	0.35	800
V5	155	blue	0.33	950
V6	130	white	0.40	600
V7	100	black	0.50	3000
V8	105	red	0.60	2500
V9	110	gray	0.55	3500

https://www.slideshare.net/picasso544/clustering-tutorial

1. 군집화, 분류

- 군집화 예제



1. 군집화, 분류

- 분류 예제

No	Height	Weight	running hour	working hour	Category
1	0.41	0.36	0.27	0.65	Patient
2	0.23	0.37	0.34	0.68	patient
3	0.38	0.38	0.46	0.95	patient
4	0.45	0.31	0.37	0.75	patient
5	0.37	0.45	0.48	0.75	patient
6	0.28	0.26	0.36	0.86	patient
7	0.66	0.44	0.51	0.98	patient
8	0.55	0.43	0.43	0.91	patient
9	0.23	0.44	0.28	0.78	patient
10	0.41	0.53	0.46	0.86	patient
11	0.65	0.38	0.74	0.51	normal
12	0.89	0.53	0.67	0.46	normal
13	0.58	0.54	0.56	0.43	normal
14	0.78	0.55	0.67	0.34	normal
15	0.89	0.56	0.81	0.56	normal
16	0.65	0.57	0.81	0.43	normal
17	0.75	0.67	0.76	0.35	normal
18	0.46	0.48	0.65	0.42	normal
19	0.89	0.69	0.78	0.23	normal
20	0.78	0.81	0.88	0.26	normal

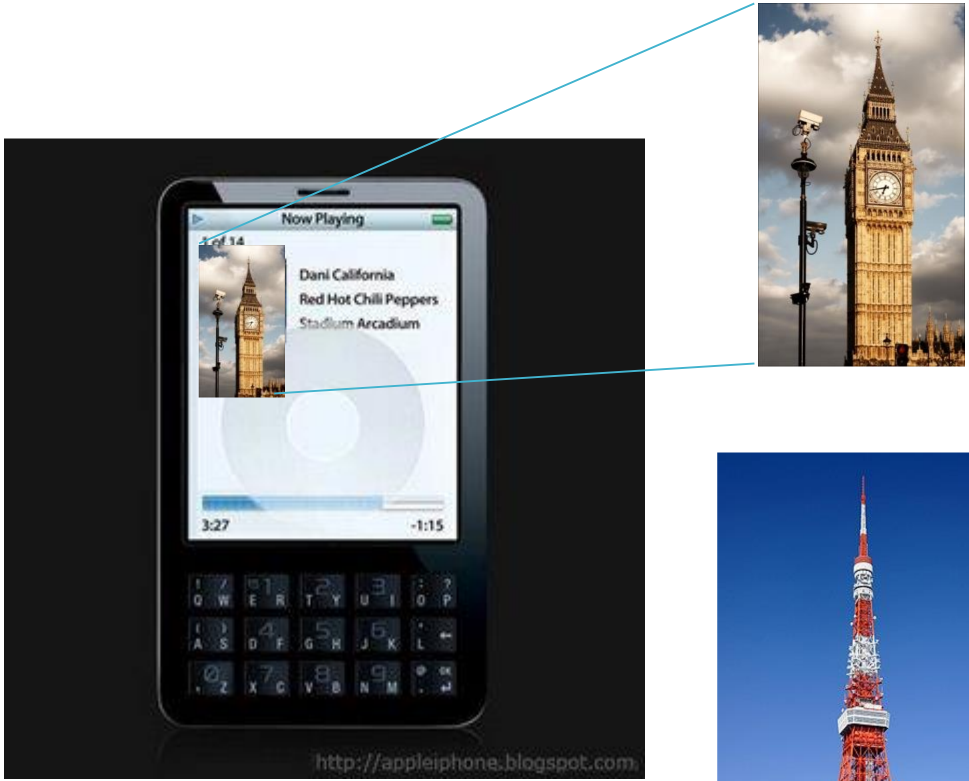
Disease A

Height	Weight	running hour	working hour
0.5	0.44	0.45	0.61

Patient or Normal ?

1. 군집화, 분류

- 분류 예제: image classification



Apple iPhone

(1) take a picture by phone camera

Tokyo tower

Empire state building

Big Ben

(2) Search similar image and shows detail information about it

1. 군집화, 분류

- 분류분석 절차

  1. Prepare target dataset that has label (class) information.
  2. Divide target dataset into training data and test data

     - assume we don’t know class labels of test data

  3. Training model using training data
  4. Predict class labels of test data using learning model
  5. Evaluate prediction performance

모델 평가 기준

$$\text{accuracy} = \frac{\#\text{ of instances that are correctly predicted}}{\#\text{ of total instances in test data}}$$

1. 군집화, 분류

- Binary vs. multiple classification

- Binary classification
  - # of class is two

Male Female

Patient Normal

Yes No

- multiple classification
  - # of class over two

Well-done medium rare

university High school

Middle school Elementary school

1. 군집화, 분류

- Binary Classification Error

Fact (실제값)

predict
(예측치)

	Fact is True	Fact is False
Predict as True	TP	FP
Predict as False	FN	TN

TP : true positive FP : false positive
FN : false negative TN : true negative

$$Accuracy = \frac{(TP + TN)}{(TP + TN + FP + FN)}$$

(정확도)

1. 군집화, 분류

- Binary Classification Error

Sensitivity = TP/(TP+FN)
(민감도)

Specificity = TN/(TN+FP)
(특이도)

- Sensitivity
  - Fraction of all Class1 (True) that we correctly predicted at Class 1
  - *How good are we at finding what we are looking for*

- Specificity
  - Fraction of all Class 2 (False) called Class 2
  - *How many of the Class 2 do we filter out of our Class 1 predictions*

어떤 평가기준이라도 값이 클수록 좋다

실습x

2. k-means clustering

- 금이간 타일과 정상 타일 군집화

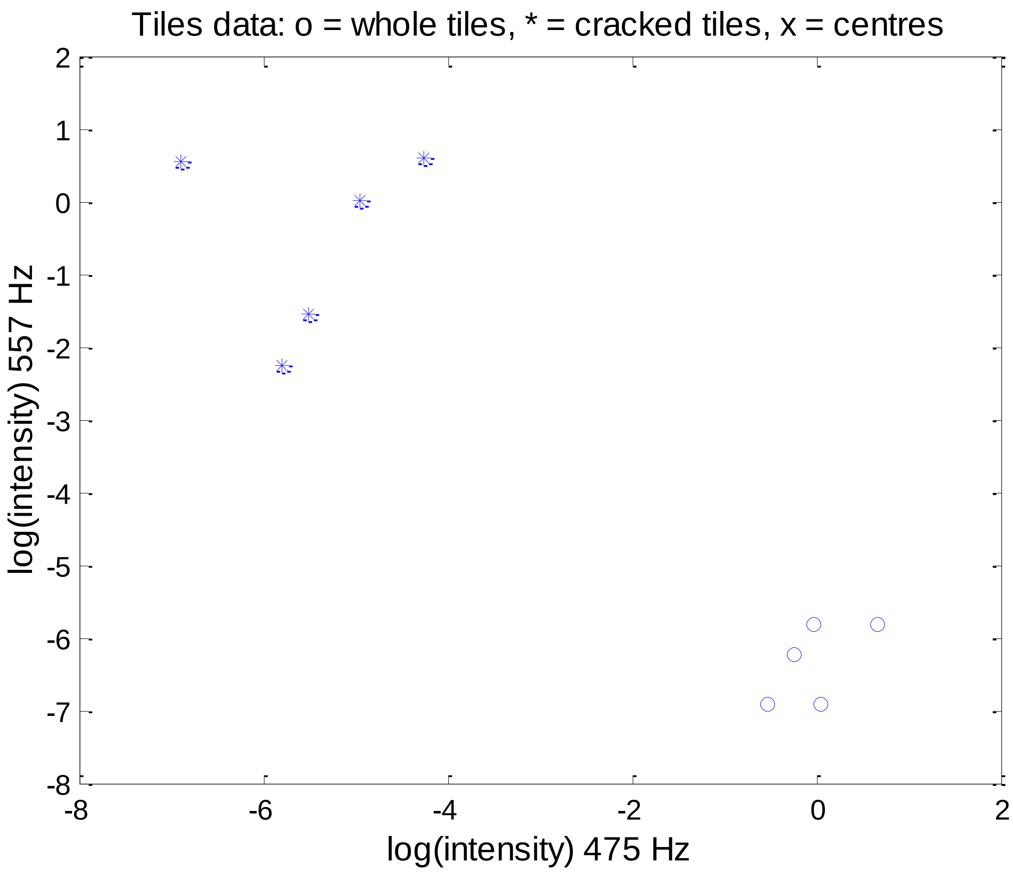
https://www.slideshare.net/picasso544/clustering-tutorial

475Hz	557Hz
0.958	0.003
1.043	0.001
1.907	0.003
0.780	0.002
0.579	0.001
0.003	0.105
0.001	1.748
0.014	1.839
0.007	1.021
0.004	0.214

Table 1: frequency intensities for ten tiles.

Tiles are made from clay moulded into the right shape, brushed, glazed, and baked. Unfortunately, the baking may produce invisible cracks. Operators can detect the cracks by hitting the tiles with a hammer, and in an automated system the response is recorded with a microphone, filtered, Fourier transformed, and normalised. A small set of data is given in TABLE 1 (adapted from MIT, 1997).

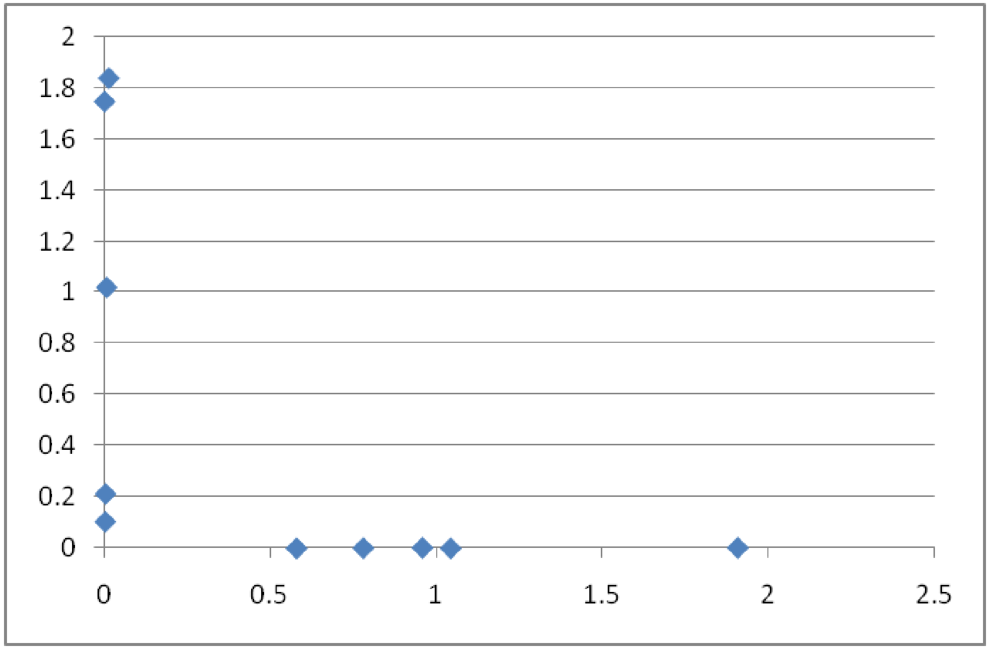
2. k-means clustering



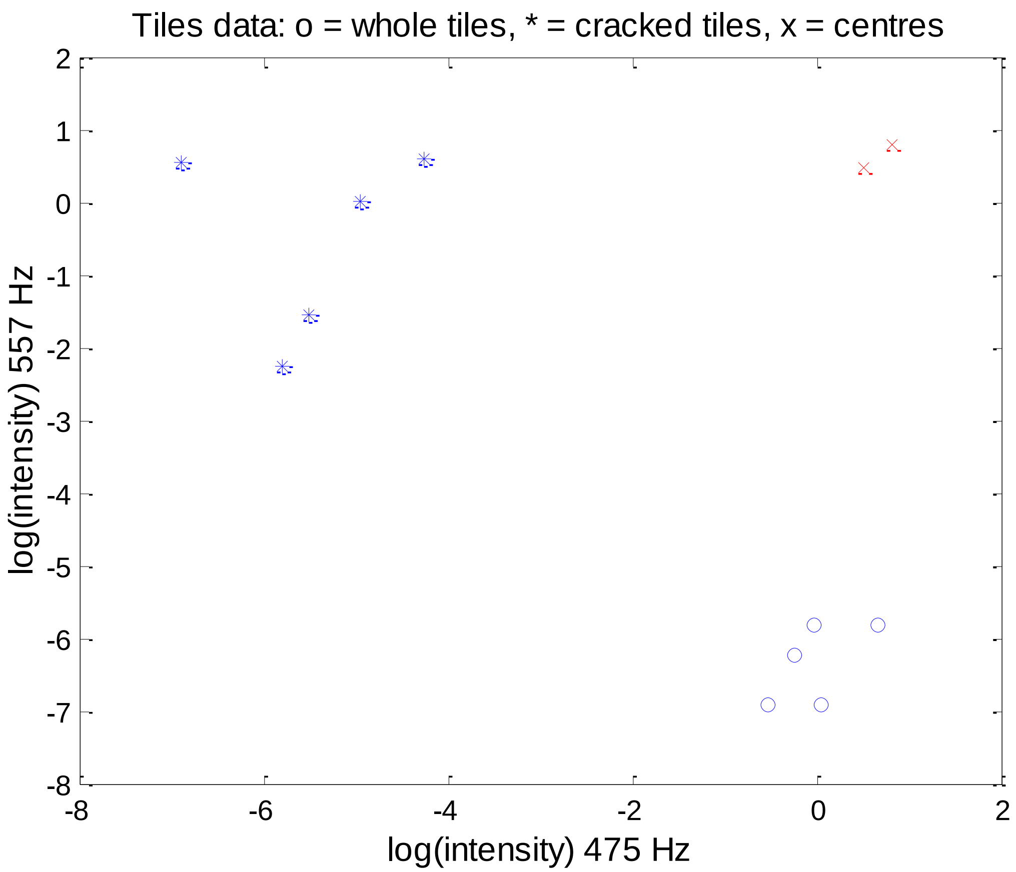
Plot of tiles by frequencies (logarithms). The whole tiles (o) seem well separated from the cracked tiles (*). The **objective** is to find the two clusters.

2. k-means clustering

- Before logarithms

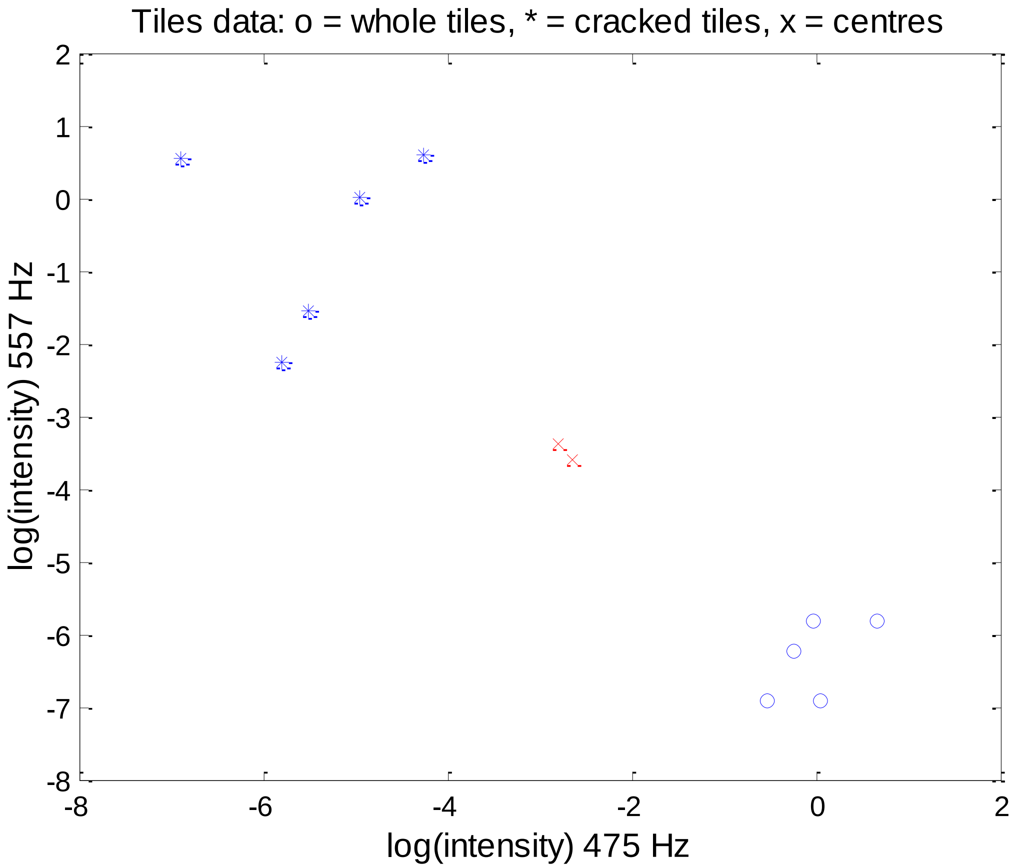


2. k-means clustering



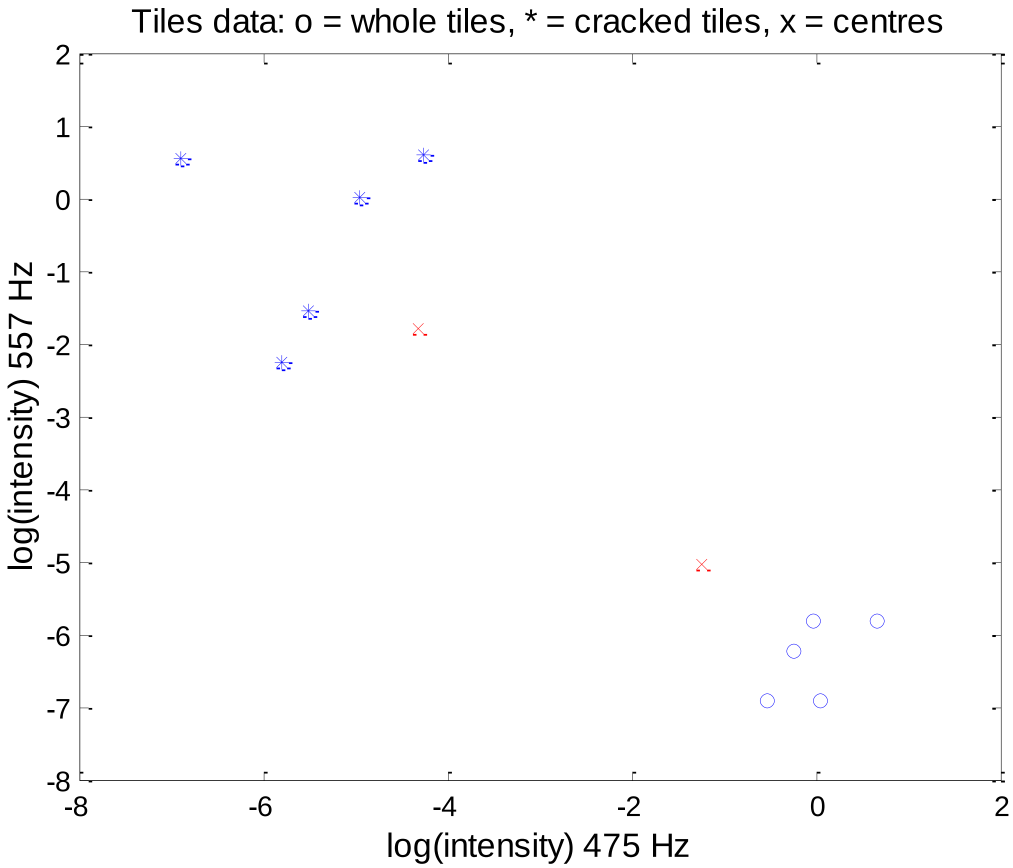
1. Place two cluster centres (x) at random.
2. Assign each data point (* and o) to the nearest cluster centre (x)

2. k-means clustering



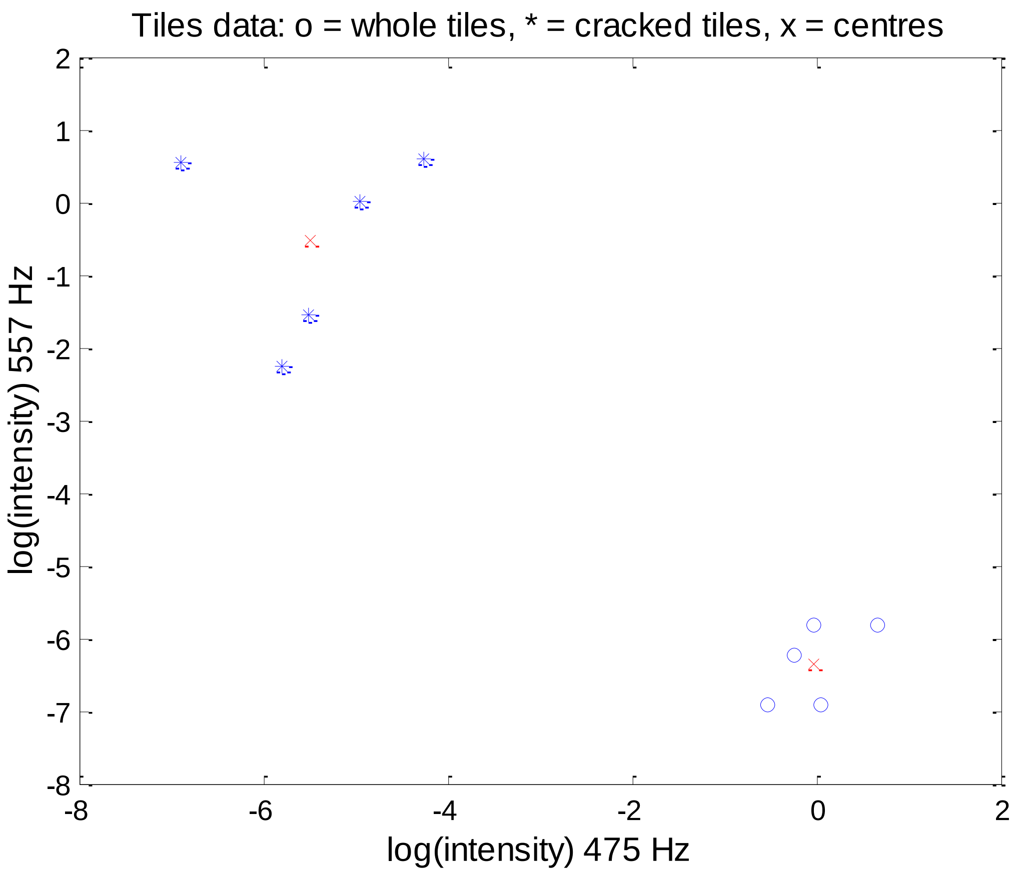
1. Compute the new centre of each class
2. Move the crosses (x)

2. k-means clustering



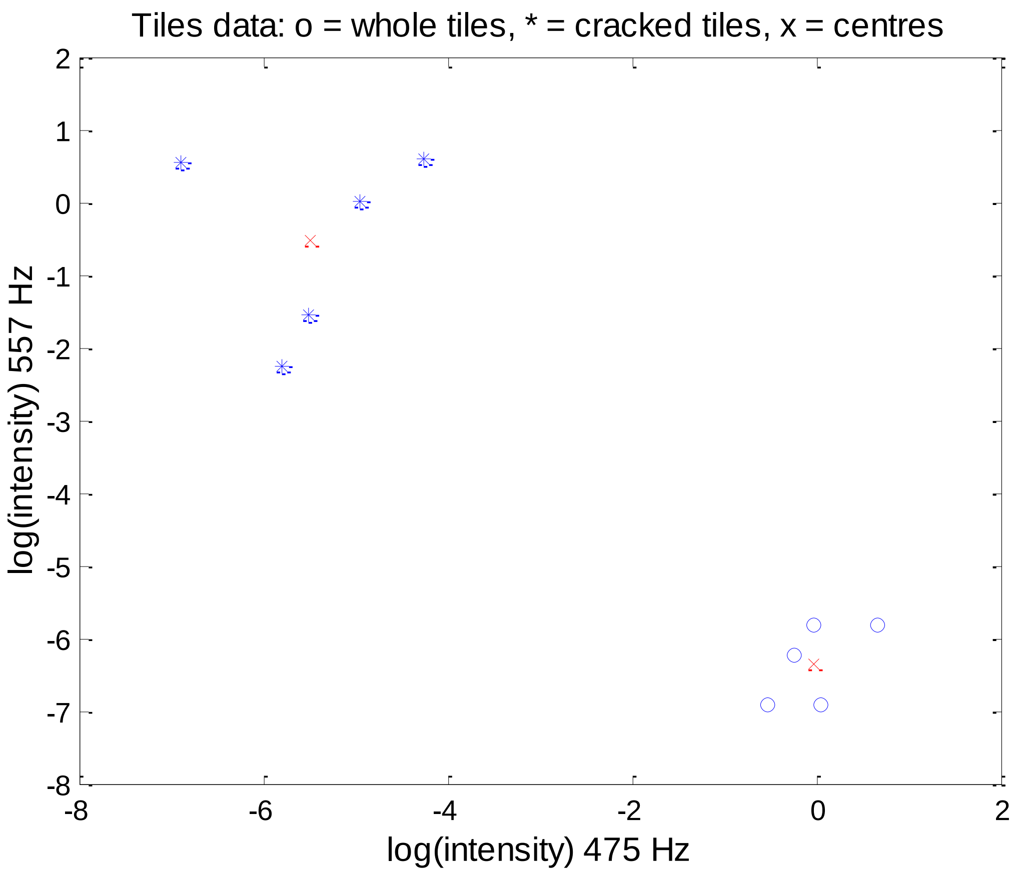
Iteration 2

2. k-means clustering



Iteration 3

2. k-means clustering



Iteration 4 (then stop, because no visible change)
Each data point belongs to the cluster defined by the nearest centre

2. k-means clustering

```
475Hz 557Hz
-----+------+            Result =
0.958 0.003                  1
1.043 0.001                  1
1.907 0.003                  1
0.780 0.002                  1
0.579 0.001                  1
0.003 0.105                  2
0.001 1.748                  2
0.014 1.839                  2
0.007 1.021                  2
0.004 0.214                  2
```

군집화 결과 :

1. The last five data points (rows) belong to the first cluster
2. The first five data points (rows) belong to the second cluster

2. k-means clustering

- 거리계산

p= (p_1,p_2, p_3 , …, p_n), q = (q_1,q_2, q_3 , …, q_n)

Euclidean distance

$$\mathrm{d}(\mathbf{p},\mathbf{q}) = \sqrt{(p_1-q_1)^2 + (p_2-q_2)^2 + \cdots + (p_n-q_n)^2} = \sqrt{\sum_{i=1}^{n}(p_i-q_i)^2}.$$

scolar

vector

Euclidean norm measure

$$\|\mathbf{p}\| = \sqrt{p_1^2 + p_2^2 + \cdots + p_n^2} = \sqrt{\mathbf{p}\cdot\mathbf{p}}$$

Distance using Euclidean norm measure

$$\|\mathbf{p}-\mathbf{q}\| = \sqrt{(\mathbf{p}-\mathbf{q})\cdot(\mathbf{p}-\mathbf{q})} = \sqrt{\|\mathbf{p}\|^2 + \|\mathbf{q}\|^2 - 2\mathbf{p}\cdot\mathbf{q}}.$$

(p•q = p_1q_1 + p_2q_2 + …+ p_nq_n)

2. k-means clustering

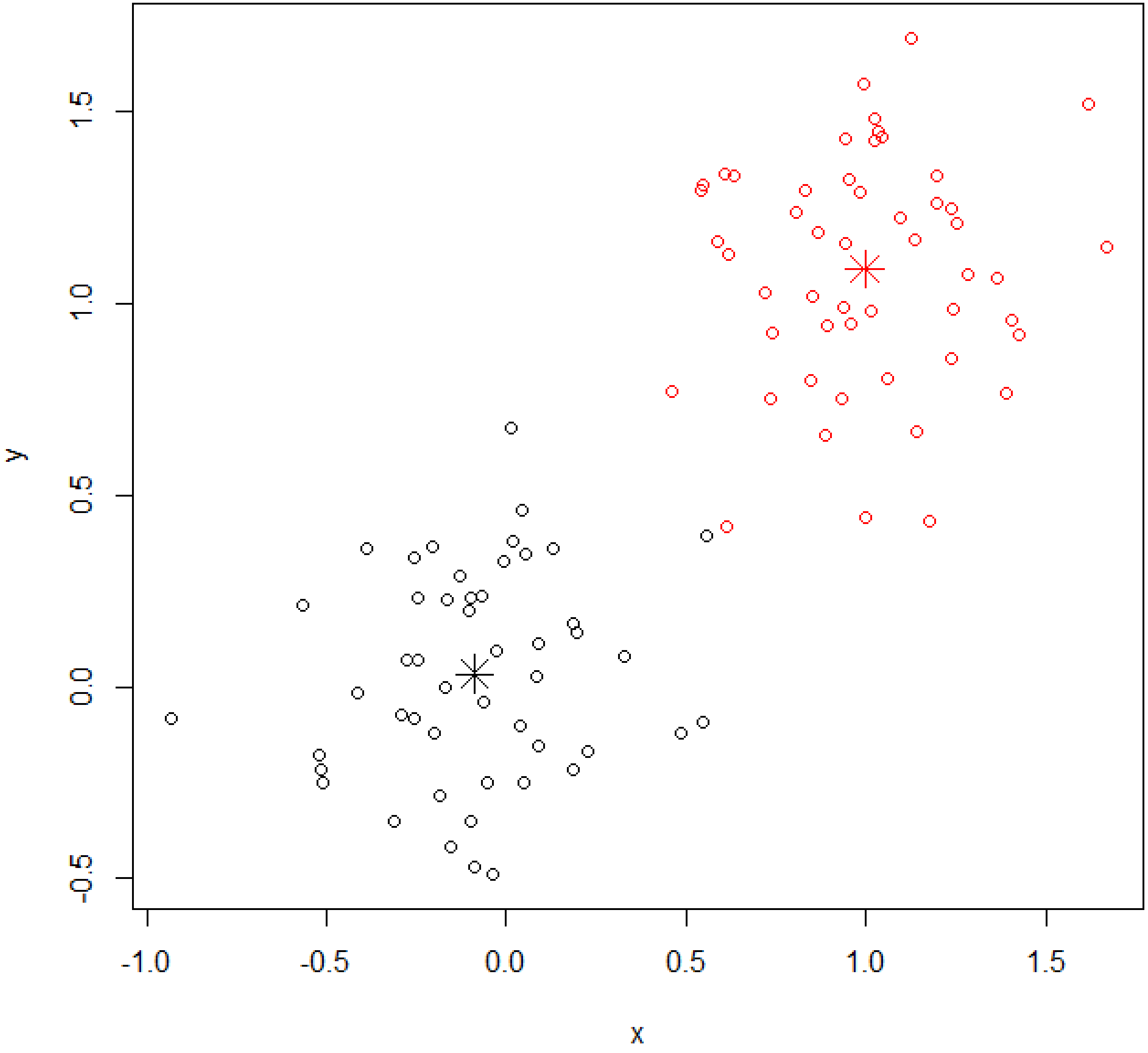
- R function: kmeans

```
kmeans(x, centers, iter.max = 10, nstart = 1,
algorithm = c("Hartigan-Wong", "Lloyd", "Forgy",
"MacQueen"))
```

- 매개변수
  - **x** : 수치형 데이터 matrix
  - **centers** : 몇 개의 그룹으로 나눌 것인가
  - **iter.max** : 그룹 중심점을 찾기 위한 최대 반복 횟수
  - **nstart** : 초기에 그룹 중심점을 임의로 잡을 때 몇 개의 점을 이용할 것인가
  - **algorithm** : 사용 알고리즘.

2. k-means clustering

```
require(graphics)
# create a 2-dimensional example
x <- rbind(matrix(rnorm(100, sd = 0.3), ncol = 2),
           matrix(rnorm(100, mean = 1, sd = 0.3),
           ncol = 2))
colnames(x) <- c("x", "y")
cl <- kmeans(x, 2)
cl # show clustering result

plot(x, col = cl$cluster)
points(cl$centers, col = 1:2, pch = 8, cex=2)

# random starts do help here with too many clusters
cl <- kmeans(x, 5, nstart = 25)
plot(x, col = cl$cluster)
points(cl$centers, col = 1:5, pch = 8)
```


-1.0
-0.5
0.0
0.5
1.0
1.5
x
-0.5
0.0
0.5
1.0
1.5
y

[연습문제 1]

- (1) iris 데이터셋에 대해 kmeans 클러스터링을 하고 결과를 그래프로 보이시오
  - iris 데이터셋에서 품종(Species) 컬럼은 제외하시오
  - 클러스터 수는 3 으로 하시오

- (2) state.x77 데이터셋에 대해 kmeans 클러스터링을 하고 결과를 그래프로 보이시오
  - 클러스터 수는 5로 하시오
  - state.x77 은 각 컬럼의 값들의 단위가 많이 차이가 나기 때문에 이를 적절히 맞추어줄 필요가 있다.

```
new.data = scale(state.x77)
```

3. KNN classification

- 분류(classification)

No	running hour	working hour	Category
1	0.27	0.65	Patient
2	0.34	0.68	patient
3	0.46	0.95	patient
4	0.37	0.75	patient
5	0.48	0.75	patient
6	0.36	0.86	patient
7	0.51	0.98	patient
8	0.43	0.91	patient
9	0.28	0.78	patient
10	0.46	0.86	patient
11	0.74	0.51	normal
12	0.67	0.46	normal
13	0.56	0.43	normal
14	0.67	0.34	normal
15	0.81	0.56	normal
16	0.81	0.43	normal
17	0.76	0.35	normal
18	0.65	0.42	normal
19	0.78	0.23	normal
20	0.88	0.26	normal

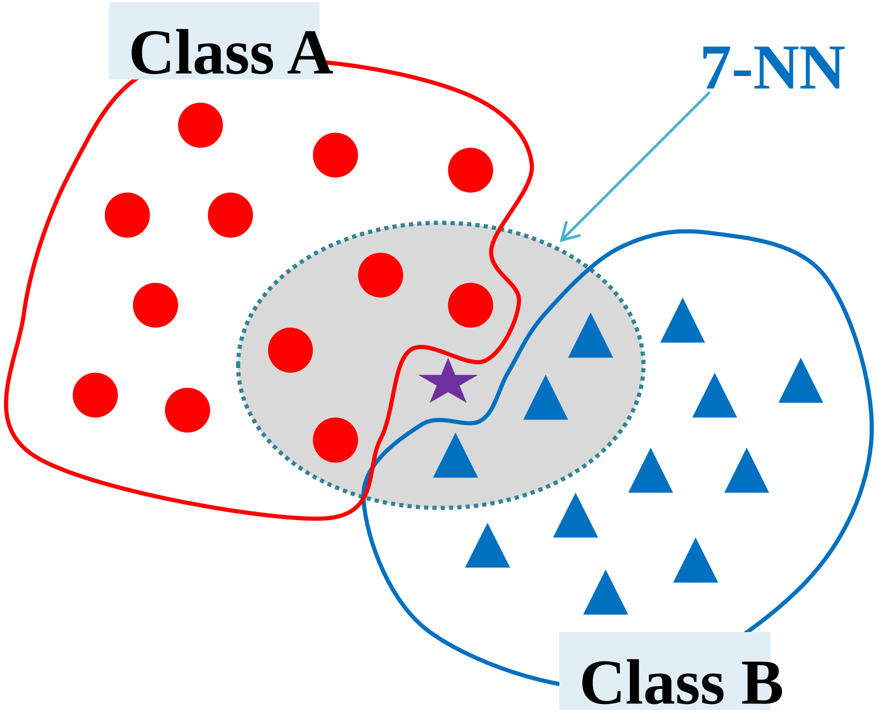
Given Classified Data

Patient or Normal ?

running hour	working hour
0.45	0.61

3. KNN classification

- Idea of KNN
  - Find K nearest neighbor for new point (★)
  - Decide new point belongs to major class (class A)
    - # of neighbor of Class A > # of neighbor of Class B

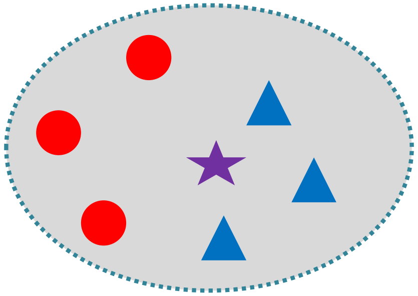


3. KNN classification

- Algorithm
  - Calculate distance between new point and every point of given classes
  - Choose K nearest points by the distance
  - Choose major class from K points

    (the class is for the new point)

6-NN



???

3. KNN classification

- How to calculate the distance between two element ?
  - Using Euclidean distance

$$\mathbf{p} = (p_1, p_2, \ldots, p_n)$$
$$\mathbf{q} = (q_1, q_2, \ldots, q_n)$$

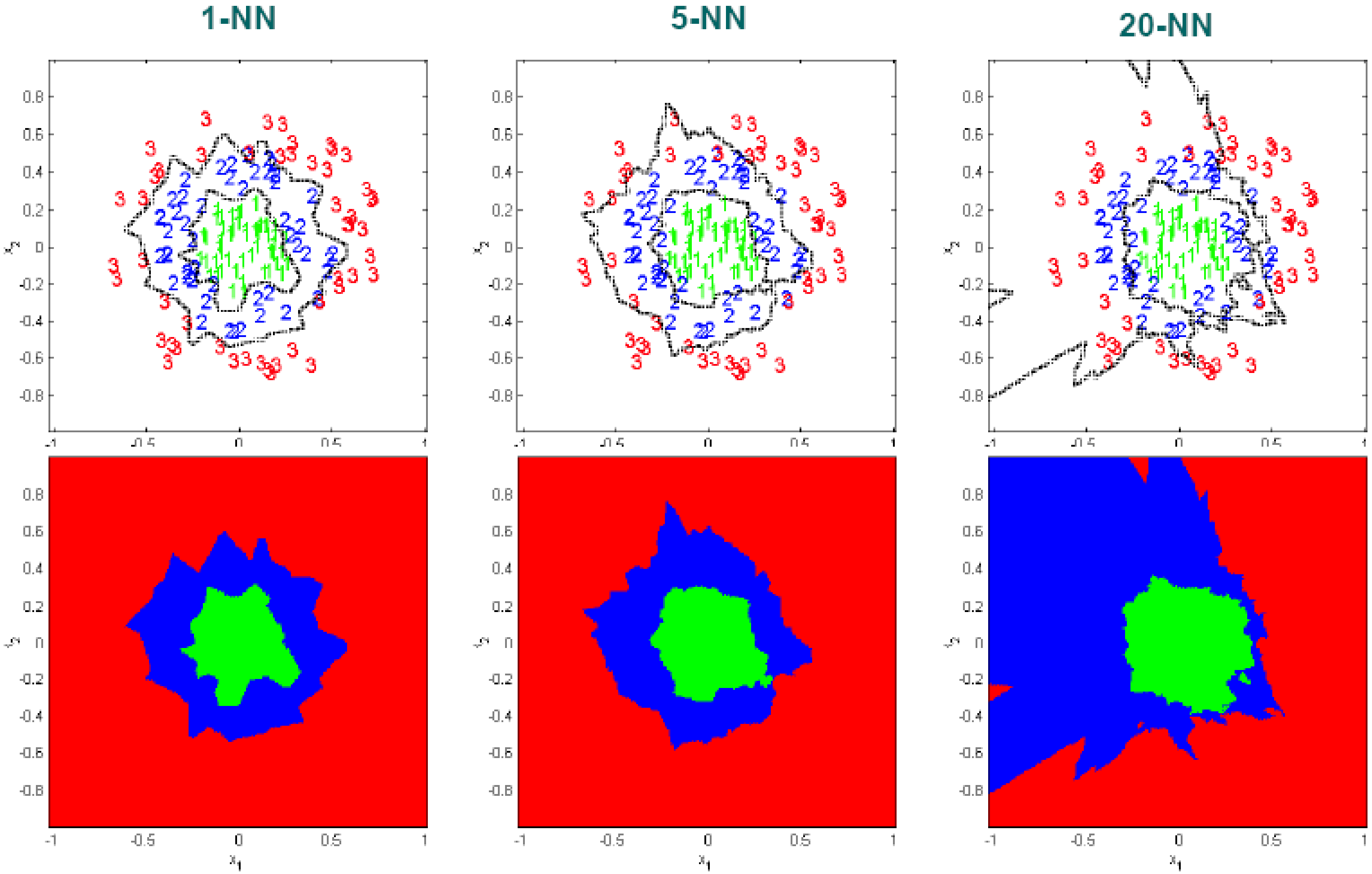
$$\mathrm{d}(\mathbf{p}, \mathbf{q}) = \sqrt{(p_1 - q_1)^2 + (p_2 - q_2)^2 + \cdots + (p_n - q_n)^2} = \sqrt{\sum_{i=1}^{n} (p_i - q_i)^2}.$$

3. KNN classification

- K 를 얼마로 하는 것이 좋은가
  - 크게 할 때와 작게 할 때 각각 장단점이 있다
  - 데이터 수가 N 이라고 할 때 K < sqrt(N) 을 권장

3. KNN classification

- 1NN vs kNN



1-NN

5-NN

20-NN

3. KNN classification

- 장점
  - 통계적 가정 불필요 (비모수적 방법)
  - 단순하다
  - 성능이 좋다
  - 모델을 훈련(학습)하는 시간이 필요 없다

- 단점
  - 데이터가 커질수록 많은 메모리 필요, 처리시간(분류시간) 증가

3. KNN classification

- R function: knn

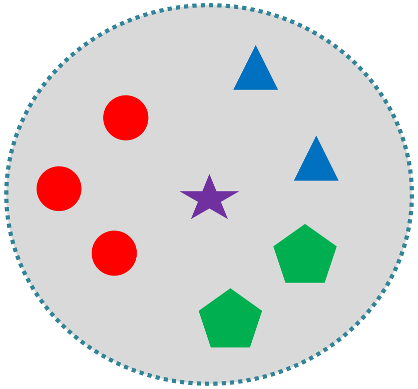
```
knn(train, test, cl, k = 1, l = 0,
    prob = FALSE, use.all = TRUE)
```

- 매개변수
  - **train** : 훈련 데이터셋 (matrix or data frame)
  - **test** : 테스트 데이터셋 (matrix or data frame)
  - **cl**: 훈련 데이터셋의 그룹(class) 정보 (factor)
  - **k** : 이웃(neighbour)의 수
  - **l** : minimum vote for definite decision, otherwise doubt.
  - **prob**: If this is true, the proportion of the votes for the winning class are returned as attribute prob.
  - **use.all:** controls handling of ties. If true, all distances equal to the kth largest are included. If false, a random selection of distances equal to the kth is chosen to use exactly k neighbours

3. KNN classification

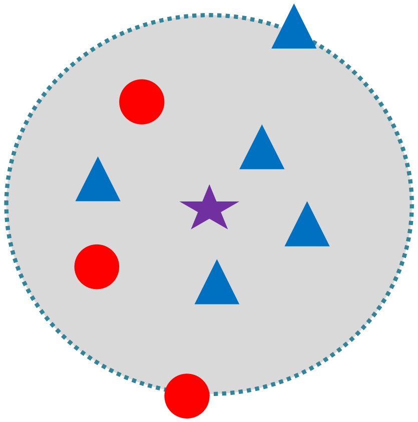
- `l` : 최다득표수가 `l` 이상이어야 유효한 예측으로 인정

K = 7



- `use.all`

K = 7



이런 경우 어떻게 처리?

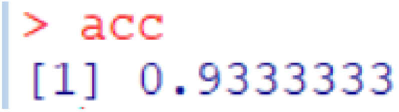
3. KNN classification

```
require("class")

# prepare train/test data
tr.idx <- c(1:25,51:75, 101:125)
ds.tr <- iris[tr.idx, 1:4]
ds.ts <- iris[-tr.idx, 1:4]
cl.tr <- factor(iris[tr.idx, 5])
cl.ts <- factor(iris[-tr.idx, 5])

pred <- knn(ds.tr, ds.ts, cl.tr, k = 3, prob=TRUE)
pred

acc <- mean(pred==cl.ts) # 예측 정확도
acc
```

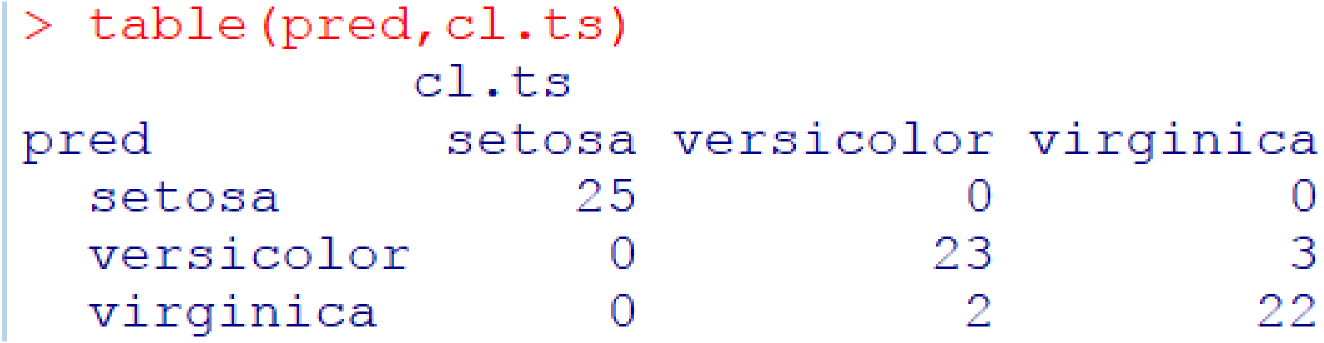
- require == library
- knn 을 이용하려면 “class” 라이브러리 필요

3. KNN classification

```
table(pred,cl.ts)
```



```
> acc
[1] 0.9333333
```



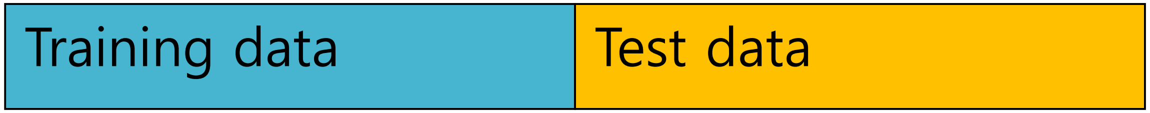
```
> table(pred,cl.ts)
            cl.ts
pred         setosa versicolor virginica
  setosa         25          0         0
  versicolor      0         23         3
  virginica       0          2        22
```

[연습문제 3]

- 다음의 데이터셋을 이용하여 KNN 알고리즘을 테스트하시오
- Target dataset : Breast Cancer Wisconsin (Diagnostic) Data Set
  - http://archive.ics.uci.edu/ml/machine-learning-databases/breast-cancer-wisconsin/wdbc.data
  - wdbc.csv 파일에 저장후 프로그램에서 읽어들인다
  - 첫번째 컬럼 : instance ID (삭제한다)
  - 두번째 컬럼 : class 정보 (M,B)
- 홀수번째 instance는 training data 로, 짝수번째 instance는 test data 로 이용한다
- K = 3,5,7 로 하여 accuracy 를 비교한다.

4. K-fold Cross Validation

- Only one classification experiment is enough ?



Training data

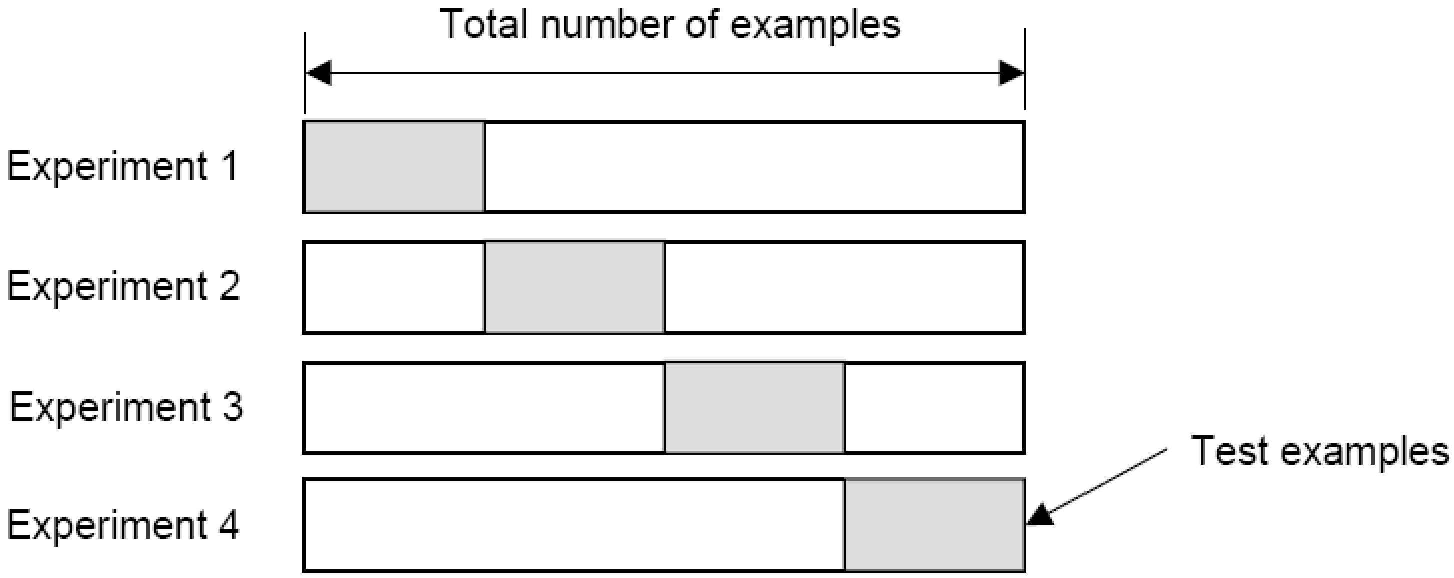
Test data

  - Classification accuracy = 0.87 (???)

- 위의 예에서 Test 데이터셋을 다르게 만들면 accuracy 가 달라질 것이다
- Test 데이터셋이 어떻게 구성되었는가에 따라 accuracy 가 원래 성능보다 높거나 낮게 나올 수도 있다.
- 그렇다면 어떻게 해야 분류 모델 또는 분류 알고리즘의 성능을 보다 정확히 알 수 있을까?

4. K-fold Cross Validation

- Create a K-fold partition of the dataset
  - For each of K experiments, use K-1 folds for training and the remaining one for testing (일반적으로 k=10 을 많이 사용)

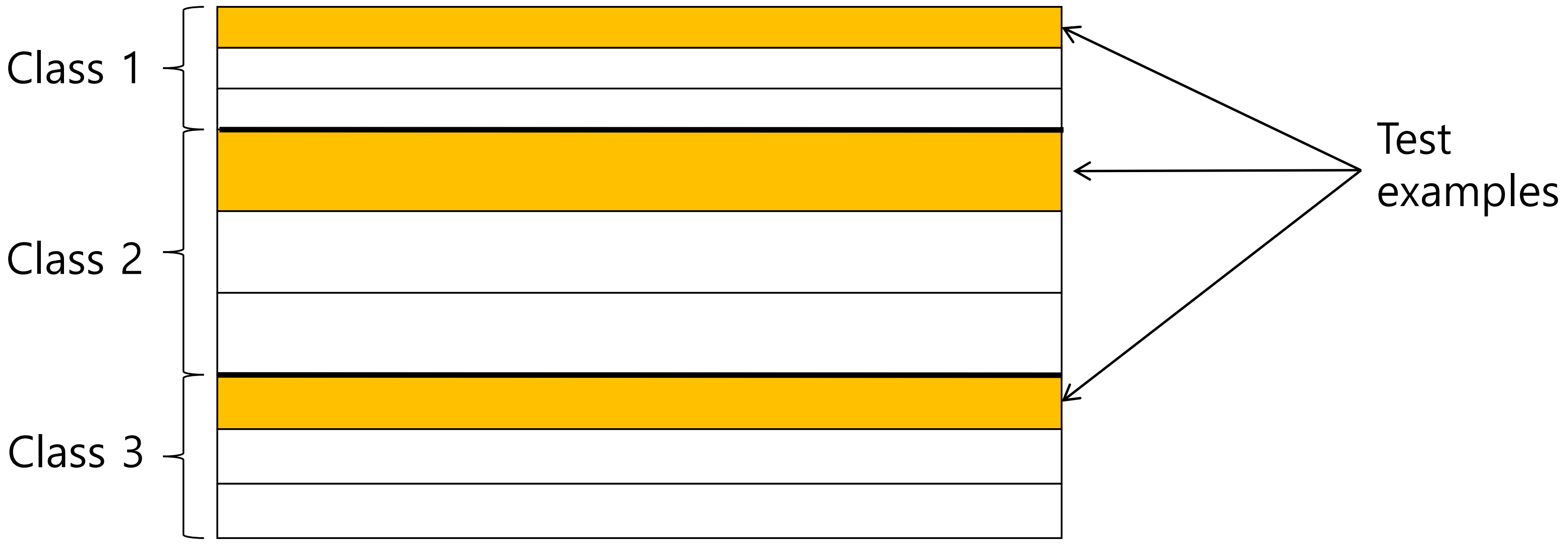


  - 모델(or 알고리즘)의 정확도는 각 fold 의 정확도들의 평균으로 계산

$$Acc = \frac{1}{K}\sum_{i=1}^{K} Acc_i$$

4. K-fold Cross Validation

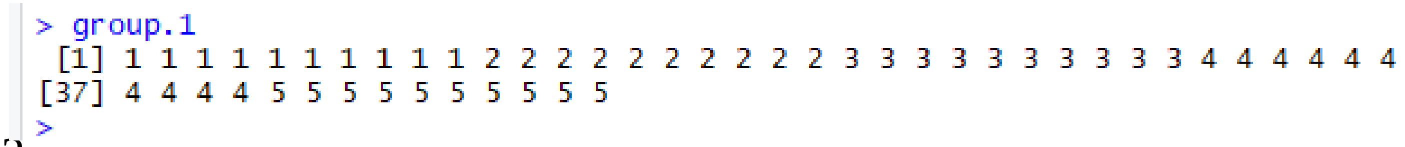
- 3-fold cross validation
  - Collect test examples from all classes by even rate (33%) of samples in the classes



4. K-fold Cross Validation

- R code : 5-fold cross validation for iris dataset

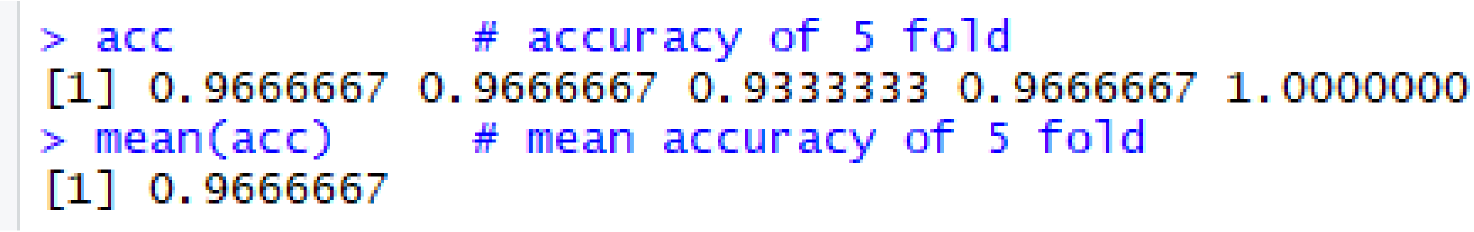
```
require("class")

# get fold no for each rows
group.1 <- cut(seq(1,50),breaks=5,labels=FALSE)
group.2 <- cut(seq(51,100),breaks=5,labels=FALSE)
group.3 <- cut(seq(101,150),breaks=5,labels=FALSE)
fold <- c(group.1, group.2, group.3)

acc <- c() # accuracy for each fold
for (i in 1:5){
  ds.tr <- iris[fold != i, 1:4]
  ds.ts <- iris[fold == i, 1:4]
  cl.tr <- factor(iris[fold != i, 5])
  cl.ts <- factor(iris[fold == i, 5])
```



```
> group.1
 [1] 1 1 1 1 1 1 1 1 1 1 2 2 2 2 2 2 2 2 2 2 3 3 3 3 3 3 3 3 3 3 4 4 4 4 4 4
[37] 4 4 4 4 5 5 5 5 5 5 5 5 5 5
>
```

4. K-fold Cross Validation

```
  pred <- knn(ds.tr, ds.ts, cl.tr, k = 3)
  acc[i] <- mean(pred==cl.ts) # 예측 정확도
}

acc             # accuracy of 5 fold
mean(acc)       # mean accuracy of 5 fold
```

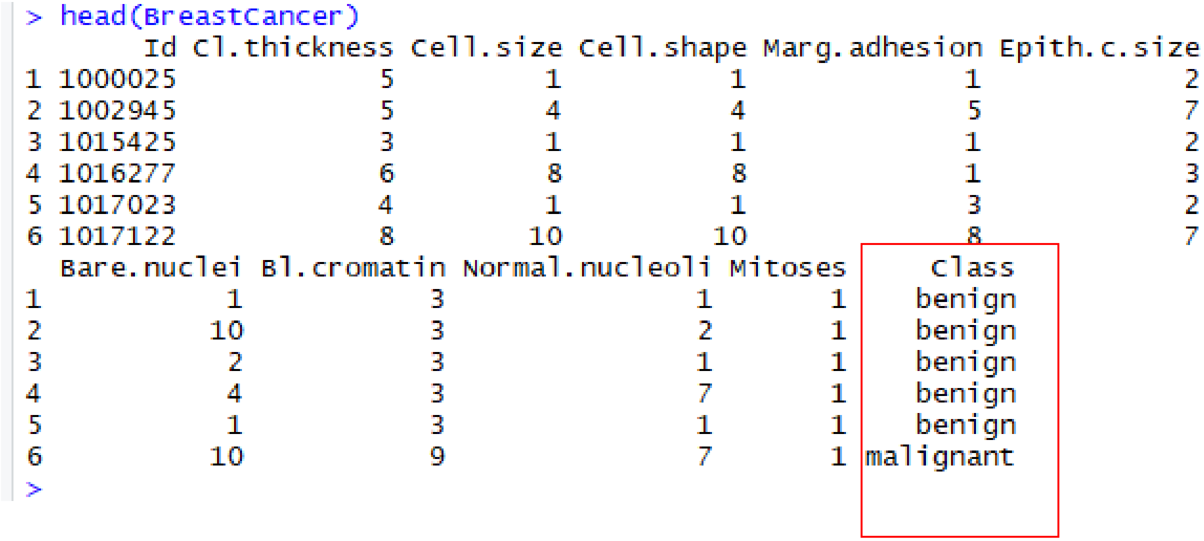


```
> acc             # accuracy of 5 fold
[1] 0.9666667 0.9666667 0.9333333 0.9666667 1.0000000
> mean(acc)       # mean accuracy of 5 fold
[1] 0.9666667
```

[연습문제 4]

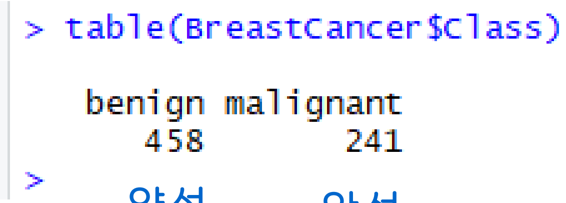
- mlbench 패키지에 포함된 유방암 데이터셋() 에 대하여 KNN 으로 예측 정확도를 알아보되 10-fold cross validation 으로 하시오

```
library(mlbench)
data(BreastCancer)
head(BreastCancer)
```



```
> head(BreastCancer)
       Id Cl.thickness Cell.size Cell.shape Marg.adhesion Epith.c.size
1 1000025            5         1          1             1            2
2 1002945            5         4          4             5            7
3 1015425            3         1          1             1            2
4 1016277            6         8          8             1            3
5 1017023            4         1          1             3            2
6 1017122            8        10         10             8            7
  Bare.nuclei Bl.cromatin Normal.nucleoli Mitoses     Class
1           1           3               1       1    benign
2          10           3               2       1    benign
3           2           3               1       1    benign
4           4           3               7       1    benign
5           1           3               1       1    benign
6          10           9               7       1 malignant
>
```

Group information



```
> table(BreastCancer$Class)

   benign malignant 
      458       241 
>
```

양성 악성