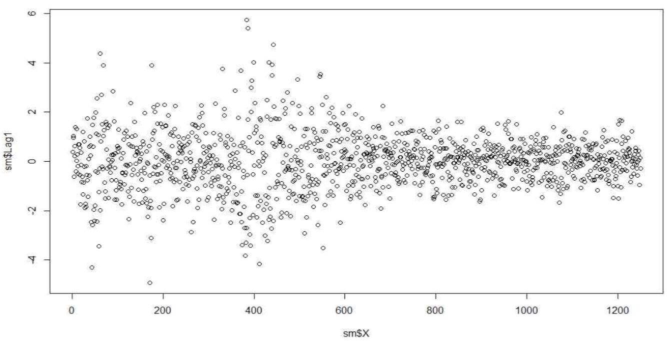
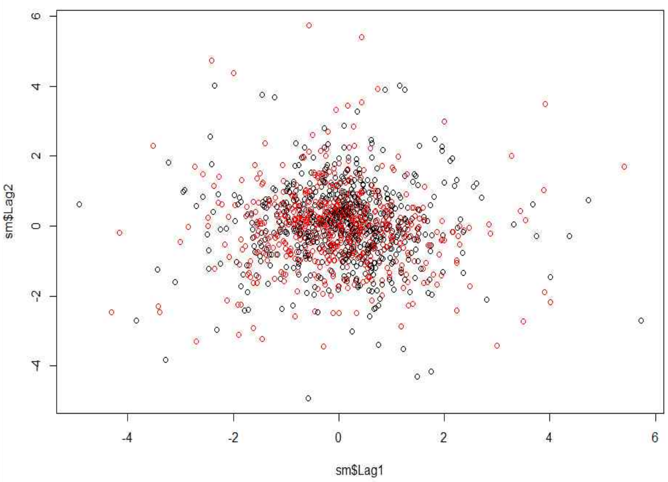
-----------------------1번--------------------

```
sm <- read.csv("C:/Users/sunni/OneDrive/바탕 화면/태양/`19년 1학기/다변량통계분석/smarket_판별분석.csv",header = TRUE)
> head(sm)
  X   Lag1   Lag2 Direction
1 1  0.381 -0.192        Up
2 2  0.959  0.381        Up
3 3  1.032  0.959      Down
4 4 -0.623  1.032        Up
5 5  0.614 -0.623        Up
6 6  0.213  0.614        Up

> plot(sm$X, sm$Lag1)
```




```
>plot(sm$Lag1,sm$Lag2,
 col=as.numeric(sm$Direction))
```



=> 데이터 산포도를 확인해 봤을 때, 첫 번째 그림은 시간과 시간에 따른 수익률을 나타낸다. 500일 전까지는 수익률의 폭이 큰 것을 볼 수 있고, 그 이후로는 폭이 작은 것을 알 수 있다.

두 번째 그림은 하루전, 이틀전 수익률을 Up과 Down으로 나눠봤는데 그림을 봤을 때, Up과 Down이 고르게 분포되어 있다는 것을 볼 수 있다.

```
> rs <- lda(sm$Direction ~ sm$Lag1 + sm$Lag2, data = sm)
> rs
Call:
lda(sm$Direction ~ sm$Lag1 + sm$Lag2, data = sm)

Prior probabilities of groups:
  Down     Up 
0.4816 0.5184 

Group means:
         sm$Lag1     sm$Lag2
Down  0.05068605  0.03229734
Up   -0.03969136 -0.02244444

Coefficients of linear discriminants:
               LD1
sm$Lag1 -0.7567605
sm$Lag2 -0.4707872
```

=> 선형 판별분석을 수행해 선형판별함수를 얻었다.

- 판별식 : Lag1 * -0.7567605 + Lag2 * -0.4707872

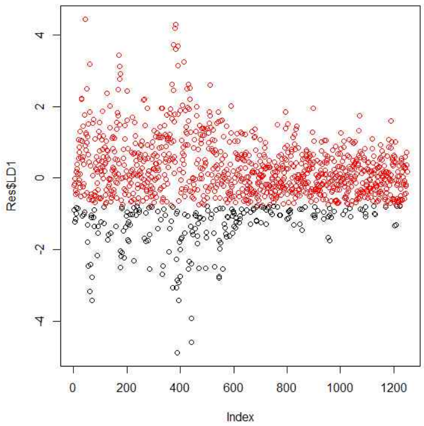
```
> calc <- with(X, Lag1 * (-0.7567605) + Lag2 * (-0.4707872))
> head(calc)
[1] -0.1979346 -0.9051032 -1.2324618 -0.0143906 -0.1713505 -0.4502533

> p <- predict(rs)
> X <- cbind(sm, p$x)
> head(X)
  X   Lag1   Lag2 Direction          LD1
1 1  0.381 -0.192        Up -0.193187790
2 2  0.959  0.381        Up -0.900356413
3 3  1.032  0.959      Down -1.227714911
4 4 -0.623  1.032        Up -0.009643717
5 5  0.614 -0.623        Up -0.166603724
6 6  0.213  0.614        Up -0.445506476
```

=> Lag1, Lag2의 값을 판별함수에 대입해 LD1이라는 값이 나왔다.

```
> pc = predict(rs, sm)$class
> head(pc)
[1] Up   Down Down Up   Up   Up
Levels: Down Up

> pc=as.numeric(pc)
> head(pc)
[1] 2 1 1 2 2 2

> res = cbind(sm$Direction, pc)

> Res <- cbind(X, res)
> head(Res)
  X   Lag1   Lag2 Direction          LD1 V1 pc
1 1  0.381 -0.192        Up -0.193187790  2  2
2 2  0.959  0.381        Up -0.900356413  2  1
3 3  1.032  0.959      Down -1.227714911  1  1
4 4 -0.623  1.032        Up -0.009643717  2  2
5 5  0.614 -0.623        Up -0.166603724  2  2
6 6  0.213  0.614        Up -0.445506476  2  2
```

V1 => Direction을 수치형으로 변환한 값
pc => 판별분석을 통해 예측된 값을 수치화 한 값
이 두가지 값을 X에 추가해 Res라는 변수로 만들었다.

```
> plot(Res$LD1, col=Res$pc)
```



=> 데이터를 판별함수에 넣어봤더니 어떤 값을 기준으로 Up, Down으로 나누는 것을 확인 할 수 가 있다.

```
> max(Res[Res$pc == 1,]$LD1)
[1] -0.7863482
> min(Res[Res$pc == 2,]$LD1)
[1] -0.7792976
```

어떤 값을 기준으로 나누는지 확인해봤더니
대략 판별함수로 나온 값이
-0.78기준으로 클 경우는 Up, 작을 경우는 Down

```
> correct.rate = dim(correct)[[1]]/n
> correct.rate
[1]   0.528 165.000

> table(res[,1],res[,2])

      1   2
  1 114 488
  2 102 546
```

=> 위 판별함수는 오류율이 1-0.528 = 0.472 이다.

- 이차 판별분석

```
> x = sm[,2:3]
> qd = qda(x,sm$Direction)
> qc = predict(qd)$class
> head(qc)
[1] Up   Up   Down Up   Up   Up
Levels: Down Up
> qc = as.numeric(qc)
> head(qc)
[1] 2 2 1 2 2 2

> resq=cbind(sm$Direction,qc)
> correctq = resq[(resq[,1]==resq[,2]),]
> correctq.rate=dim(correctq)[[1]]/n

> correctq.rate
[1]   0.5304 165.7500
```

=> 이차 판별분석의 오류율은 1-0.5304 = 0.4696으로 선형 판별분석보다 나은 결과가 나온 것을 확인할 수 있다.

- 교차타당성 검정

=> 위의 오분류율은 전체데이터를 가지고 모델링 후 다시 전체데이터를 테스트 데이터로 썼기 때문에 제대로 된 검정 방법이 아니다.

=> 1250개중 1개의 데이터를 제외한 1249로 모델링 하고 1개로 검정을 1250번 반복해 나온 값을 오분류율로 정했다.

```
> ldc = lda(sm$Direction ~ sm$Lag1 + sm$Lag2, data = sm, CV = TRUE, prior = c(0.4816,0.5184))
> results = data.frame(sm$Direction, ldc$class, ldc$posterior)
> head(results)
  sm.Direction ldc.class      Down        Up
1           Up        Up 0.4861599 0.5138401
2           Up      Down 0.5030997 0.4969003
3         Down      Down 0.5098580 0.4901420
4           Up        Up 0.4822116 0.5177884
5           Up        Up 0.4857059 0.5142941
6           Up        Up 0.4921773 0.5078227
> class.table = table(sm$Direction, ldc$class)
> class.table

       Down  Up
  Down  109 493
  Up    109 539
```

정분류율 = (109 + 539) / 1250 = 0.5184
오분류율 = 1 - 0.5184 = 0.4816

=> 오분류율이 위의 두 방법보다 당연히 증가함을 볼 수 있다. 데이터 1개의 차이라고 볼 수 있다.
=> 판별분석해 본 결과 1,2일전 주식 수익률 가지고 현 시점의 수익률 예측하는 것은 어렵다고 결론을 내렸다.

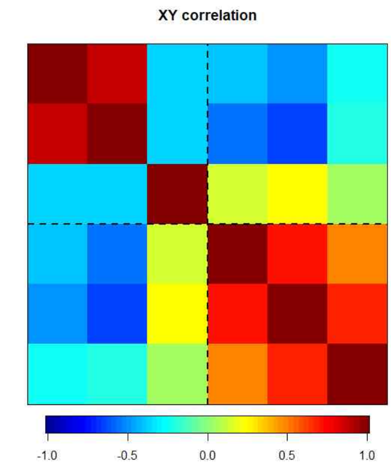
-----------------------2번--------------------

```
> fit <- read.csv("C:/Users/sunni/OneDrive/바탕 화면/태양/`19년 1학기/다변량통계분석/Fitness Club_정준상관분석.csv",header = TRUE)

> head(fit)
  Weight Waist Pulse Chins Situps Jumps
1    191    36    50     5    162    60
2    189    37    52     2    110    60
3    193    38    58    12    101   101
4    162    35    62    12    105    37
5    189    35    46    13    155    58
6    182    36    56     4    101    42

> x = fit[1:3]
> y = fit[4:6]
```

Physiological 변수와 Exercise 변수를 X,Y로 나눔

```
> matcor(x,y)
$Xcor
           Weight     Waist      Pulse
Weight  1.0000000  0.8702435 -0.3657620
Waist   0.8702435  1.0000000 -0.3528921
Pulse  -0.3657620 -0.3528921  1.0000000

$Ycor
           Chins    Situps     Jumps
Chins  1.0000000 0.6957274 0.4957602
Situps 0.6957274 1.0000000 0.6692061
Jumps  0.4957602 0.6692061 1.0000000

$XYcor
           Weight      Waist       Pulse
Chins      Situps      Jumps
Weight     1.0000000   0.8702435   -0.36576203
-0.3896937 -0.4930836 -0.22629556
Waist      0.8702435   1.0000000   -0.35289213
-0.5522321 -0.6455980 -0.19149937
Pulse      -0.3657620  -0.3528921  1.00000000
0.1506480  0.2250381  0.03493306
Chins      -0.3896937  -0.5522321  0.15064802
1.0000000  0.6957274  0.49576018
Situps     -0.4930836  -0.6455980  0.22503808
0.6957274  1.0000000  0.66920608
Jumps      -0.2262956  -0.1914994  0.03493306
0.4957602  0.6692061  1.00000000
```

```
> library(CCA)
> img.matcor(mtc,type=1)
```



위 표를 봤을 때, Weight, Waist는 높은 양의 상관관계가 있다는 것을 볼 수 있고, Exercise 변수 끼리는 서로 양의 상관관계가 있음을 볼 수 있다.

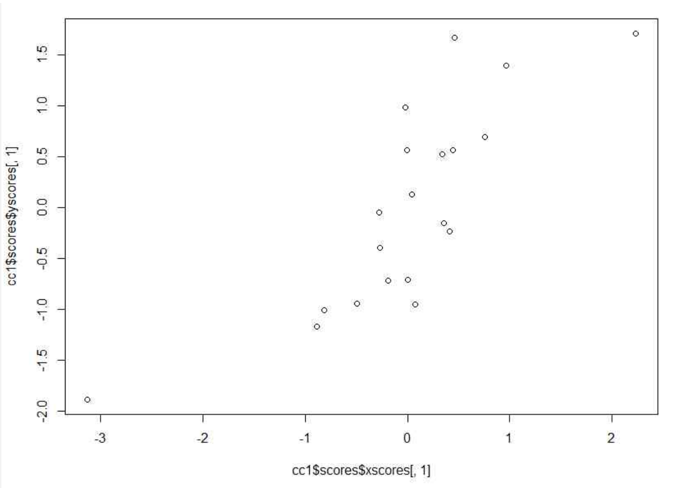
또한 Weight, Waist 와 Exercise변수끼리는 음의 상관관계가 있음을 볼 수 있다.

```
> cc1 <- cc(x,y)
> cc1$cor
[1] 0.79560815 0.20055604 0.07257029
```

0.79이므로 X,Y간의 양의 상관관계가 있는 걸 확인

첫 번째 정준 변수 그래프

```
plot(cc1$scores$xscores[,1],cc1$scores$yscores[,1])
```



첫 번째 정준변수와 X와의 상관계수

```
> cc1$scores$corr.X.xscores
             [,1]       [,2]        [,3]
Weight -0.6206424  0.7723919 -0.13495886
Waist  -0.9254249  0.3776614 -0.03099486
Pulse   0.3328481 -0.0414842  0.94206752
```

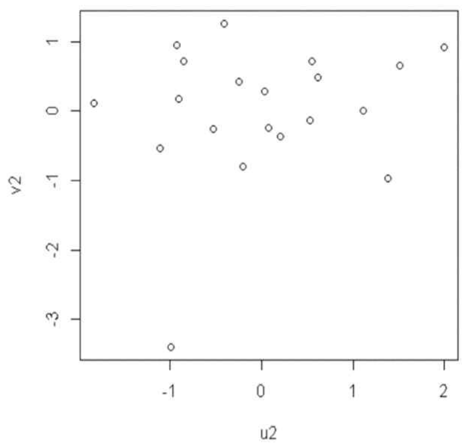
```
//U1 = - 0.0661 * X1 - 0.0168 * X2 + 0.0140 * X3
//U2 = 0.0710 * X1 - 0.0020 * X2 - 0.0207 * X3
//U3 = 0.2453 * X1 - 0.0198 * X2 + 0.0082 * X3
```

첫 번째 정준변수와 Y와의 상관계수

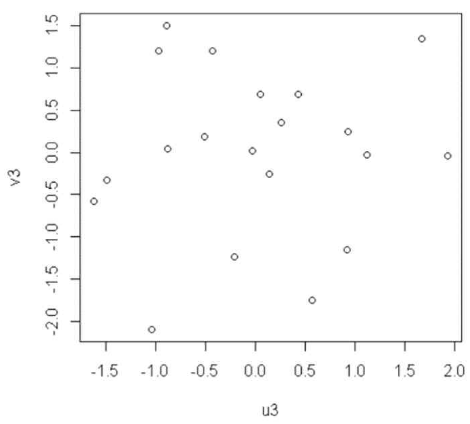
```
> cc1$scores$corr.Y.yscores
            [,1]       [,2]        [,3]
Chins  0.7276254 -0.2369522 -0.64375064
Situps 0.8177285 -0.5730231  0.05444915
Jumps  0.1621905 -0.9586280 -0.23393722
```

```
//V1 = -0.0314 * Y1 + 0.4932 * Y2 - 0.0082 * Y3
//V2 = 0.0763 * Y1 - 0.3687 * Y2 + 0.0321 * Y3
//V3 = 0.0077 * Y1 - 0.1580 * Y2 - 0.1457 * Y3
```

```
> plot(u2,v2)
```

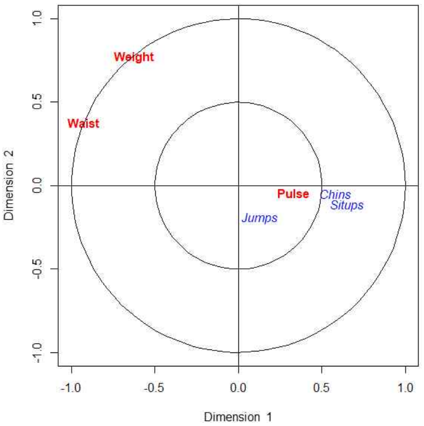



```
> plot(u3,v3)
```



=> 두번째 정준변수와 세번째 정준상관변수 사이의 상관계수가 낮고 위의 표에서도 상관성이 없다고 볼 수 있다.

> plt.cc(cc1,type="v",var.label=TRUE)



> cc2=comput(x,y,cc1)

> cc2[3:6]

$corr.X.xscores

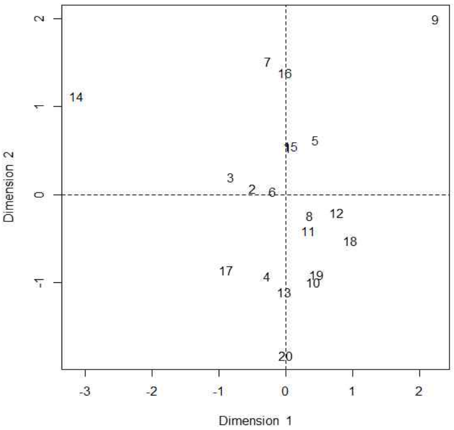
	[,1]	[,2]	[,3]
Weight	-0.6206424	0.7723919	-0.13495886
Waist	-0.9254249	0.3776614	-0.03099486
Pulse	0.3328481	-0.0414842	0.94206752

$corr.Y.xscores

	[,1]	[,2]	[,3]
Chins	0.5789047	-0.0475222	-0.04671717
Situps	0.6505914	-0.1149232	0.00395139
Jumps	0.1290401	-0.1922586	-0.01697689

=> 위 그래프는 X와 U의 상관계수 행렬과 Y와 U의 상관계수 행렬에서 앞의 두 열을 사용해 평면에 나타낸 그래프이다.

> plt.cc(cc1,type="i",var.label=TRUE)



$scores

$scores$xscores

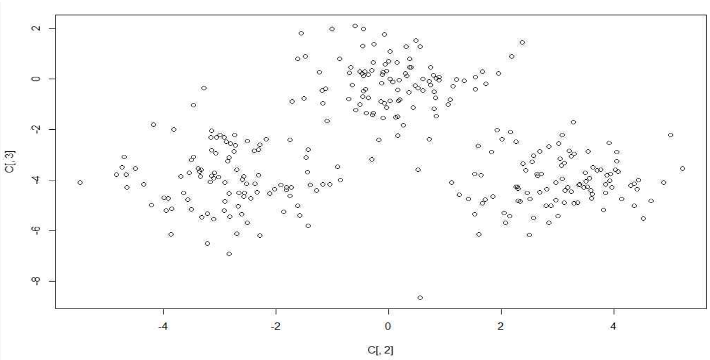
	[,1]	[,2]	[,3]
[1,]	0.043457300	0.52961092	-0.89006107
[2,]	-0.496195120	0.07235291	-0.42509284
[3,]	-0.814622152	0.20121991	0.57639407
[4,]	-0.275645187	-0.93030784	0.92500854
[5,]	0.441092338	0.61748692	-1.61555359
[6,]	-0.189988998	0.03504733	0.05394781
[7,]	-0.265736402	1.51086703	0.14569875
[8,]	0.358221297	-0.24409131	0.43683518
[9,]	2.235378930	1.99768113	1.93337021
[10,]	0.410404769	-0.99572988	-0.20357183
[11,]	0.339037518	-0.41197224	-1.03595733
[12,]	0.754463761	-0.20810378	-0.87932136
[13,]	-0.017242384	-1.10803692	1.12031975
[14,]	-3.130296936	1.11627338	0.25711279
[15,]	0.073469414	0.55404196	-1.48846013
[16,]	-0.005941024	1.38502643	0.93167397
[17,]	-0.888057432	-0.85569669	-0.03307275
[18,]	0.965063246	-0.52625635	-0.96773958
[19,]	0.456815513	-0.90719486	-0.51050641
[20,]	0.006321548	-1.83221805	1.66897582

=> 위 그래프는 X의 score점수의 앞의 두 열을 사용한 그래프이다.

이 때, xscores, yscores는 정준변수에 각 케이스의 값을 대입한 정준변수 점수를 의미합니다.

----------------------3번--------------------

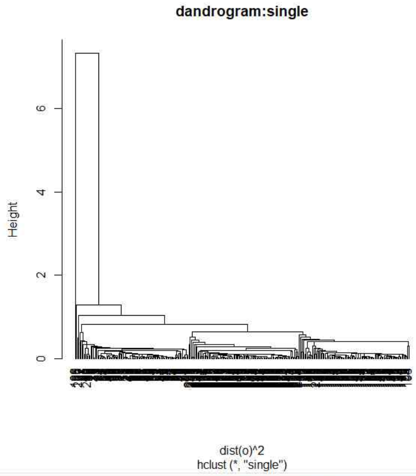
```
> C <- read.csv("C:/Users/sunni/OneDrive/바탕 화면/태양/`19년 1학기/다변량통계분석/Clustering.csv",header = TRUE)

> head(C)
  X         V1        V2
1 1 3.9191803 -2.538902
2 2 0.5241628 -3.584091
3 3 4.8818801 -4.082057
4 4 1.6654923 -4.915347
5 5 4.0784496 -3.655298
6 6 4.1445951 -4.754012

> plot(C[,2],C[,3])
```



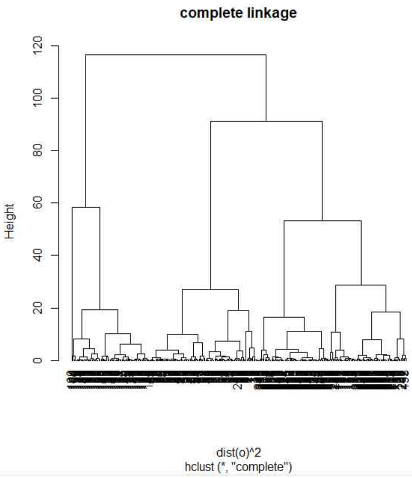
- 산포도 확인

```
> o = C[,2:3]
> head(o)
         V1        V2
1 3.9191803 -2.538902
2 0.5241628 -3.584091
3 4.8818801 -4.082057
4 1.6654923 -4.915347
5 4.0784496 -3.655298
6 4.1445951 -4.754012
```

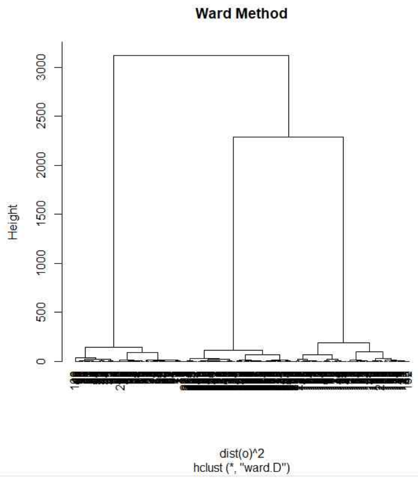
```
> hc1=hclust(dist(o)^2,method="single")
> plot(hc1,hang=-1,main="dandrogram:single")
```




```
> hc2=hclust(dist(o)^2,method="complete")
> plot(hc2,hang=-1,main="complete linkage")
```

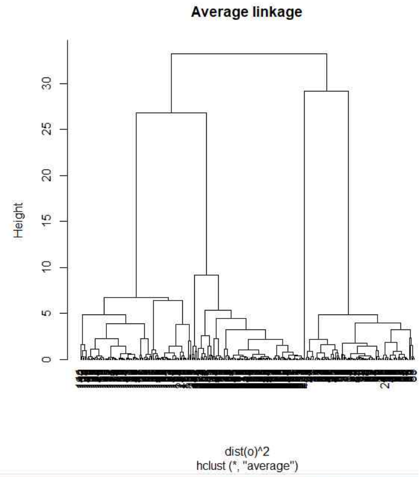



```
> hc3=hclust(dist(o)^2,method="ward.D")
> plot(hc3,hang=-1,main="Ward Method")
```



> hc4=hclust(dist(o)^2,method="average")

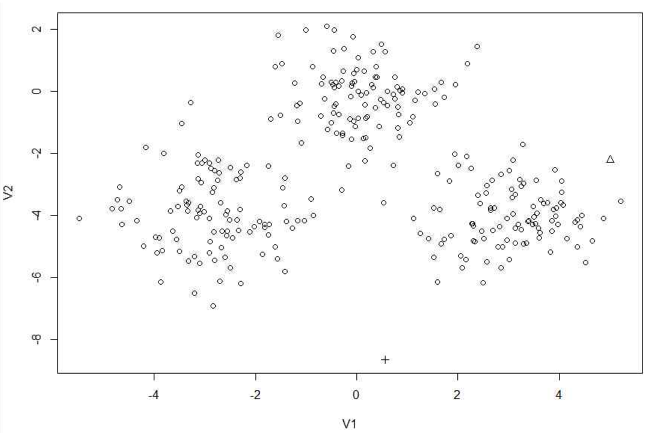
> plot(hc4,hang=-1,main="Average linkage")



=> 4가지 군집분석 방법을 확인하니 3개의 군집으로 나누는 것이 가장 적합하다고 예상

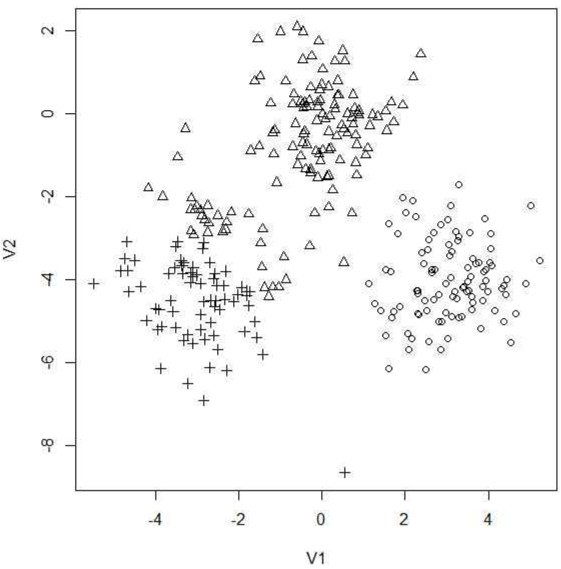
> hc1.result = cutree(hc1,k=3)

> plot(o,pch=hc1.result)



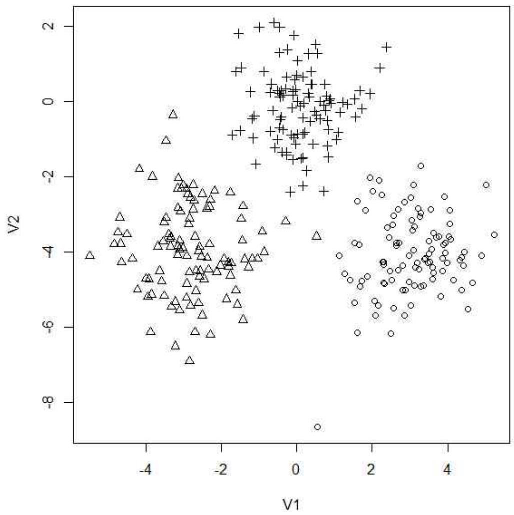
> hc2.result=cutree(hc2,k=3)

> plot(o,pch=hc2.result)



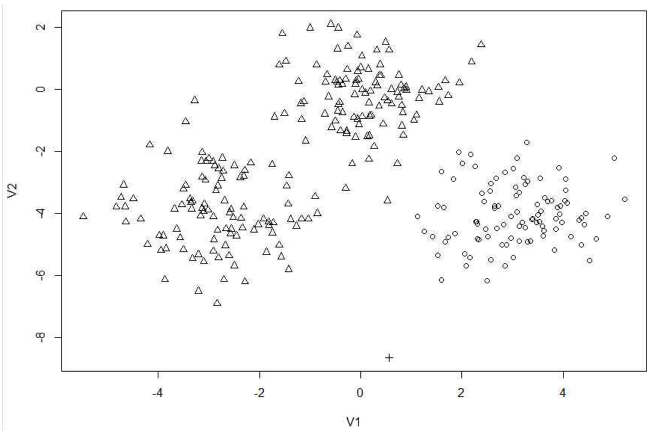
> hc3.result=cutree(hc3,k=3)

> plot(o,pch=hc3.result)



> hc4.result = cutree(hc4,k=3)

> plot(o,pch=hc4.result)



=> 4가지 군집방법으로 군집화 하니 직관적으로 최장거리법과 왈드 방법이 잘 군집화 되어 있는 것을 볼 수 있습니다. 이 두가지 중 어떤 것이 더 군집화 되어있는지 육안으로 확인하기 어려워 각 군집별 거리의 총 합을 구해봤습니다.

- 최장거리법의 군집별 평균과의 거리의 합을 구한다.

> hc2_o = cbind(o,hc2.result)

> head(hc2_o)

	V1	V2	hc2.result
1	3.9191803	-2.538902	1
2	0.5241628	-3.584091	2
3	4.8818801	-4.082057	1
4	1.6654923	-4.915347	1
5	4.0784496	-3.655298	1
6	4.1445951	-4.754012	1

```
> mean(hc2_o[hc2_o[,3] == 1,][,1])
[1] 3.059381
> mean(hc2_o[hc2_o[,3] == 1,][,2])
[1] -4.033241
> mean(hc2_o[hc2_o[,3] == 2,][,1])
[1] -0.4943919
> mean(hc2_o[hc2_o[,3] == 2,][,2])
[1] -0.7914221
> mean(hc2_o[hc2_o[,3] == 3,][,1])
[1] -3.010122
> mean(hc2_o[hc2_o[,3] == 3,][,2])
[1] -4.557119
```

=> 위에서부터
1로 군집화된 V1,V2의 평균
2로 군집화된 V1,V2의 평균
3로 군집화된 V1,V2의 평균

```
> sum((hc2_o[hc2_o[,3] == 1,][,1] - (3.059381))^2 + (hc2_o[hc2_o[,3] == 1,][,2] - (-4.03241))^2)
[1] 169.2459
> sum((hc2_o[hc2_o[,3] == 2,][,1] - (-0.4943919))^2 + (hc2_o[hc2_o[,3] == 2,][,2] - (-0.7914221))^2)
[1] 528.465
> sum((hc2_o[hc2_o[,3] == 3,][,1] - (-3.010122))^2 + (hc2_o[hc2_o[,3] == 3,][,2] - (-4.557119))^2)
[1] 134.3898
```

=> 위 값은 각 군집의 평균에서 각 데이터의 거리의 합을 나타낸 값이다. 각 평균에서의 거리 합을 보면 두번째 군집은 다른 두 군집보다 평균과 각 데이터간의 거리가 있다는 것을 알 수 있고, 다른 두 군집은 각각 평균에 잘 뭉쳐져 있다는 것을 수치적으로 알 수 있다.
위 세값을 더하면 832.1007이 나온다.

두 번째로 왈드방법으로 군집별 평균과의 거리의 합.

```
> hc3_o = cbind(o,hc3.result)
> head(hc3_o)
         V1        V2 hc3.result
1 3.9191803 -2.538902          1
2 0.5241628 -3.584091          2
3 4.8818801 -4.082057          1
4 1.6654923 -4.915347          1
5 4.0784496 -3.655298          1
6 4.1445951 -4.754012          1
```

```
> mean(hc3_o[hc3_o[,3] == 1,][,1])
[1] 3.034109
> mean(hc3_o[hc3_o[,3] == 1,][,2])
[1] -4.07974
> mean(hc3_o[hc3_o[,3] == 2,][,1])
[1] -2.79909
> mean(hc3_o[hc3_o[,3] == 2,][,2])
[1] -3.935931
> mean(hc3_o[hc3_o[,3] == 3,][,1])
[1] 0.09072205
> mean(hc3_o[hc3_o[,3] == 3,][,2])
[1] -0.1349941
```

```
> sum((hc3_o[hc3_o[,3] == 1,][,1] - (3.034109))^2 + (hc3_o[hc3_o[,3] == 1,][,2] - (-4.07974))^2)
[1] 196.42
> sum((hc3_o[hc3_o[,3] == 2,][,1] - (-2.79909))^2 + (hc3_o[hc3_o[,3] == 2,][,2] - (-3.935931))^2)
[1] 248.4246
> sum((hc3_o[hc3_o[,3] == 3,][,1] - (0.09072205))^2 + (hc3_o[hc3_o[,3] == 3,][,2] - (-0.1349941))^2)
[1] 160.5232
```

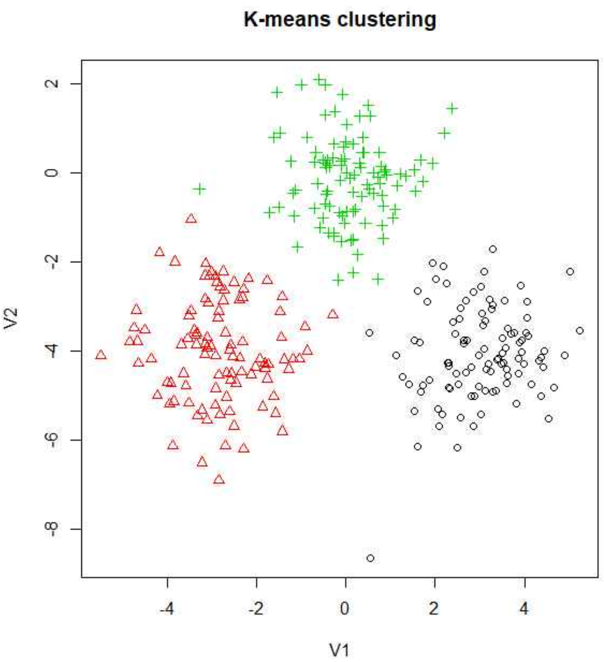
=> 위 세 값은 대체적으로 작은 편이고 위 결과로 모두 평균에 뭉쳐져 있다는 것을 알 수 있다.
총 합을 구하면 605.3678라는 값이 나온다.

=> 최장거리법의 평균제곱합은 832.1007, 왈드 방법의 평균제곱합은 605.3678로 수치적으로 왈드 방법이 더욱 이상적인 군집방법이라고 볼 수 있다.

=> 위의 평균제곱합을 기준으로 모델을 평가했는데, 평균제곱합을 기준으로 최소가 되는 모델링의 방법이 있는데 바로 K-Means이다.

- K-Means

```
> clustering_k=kmeans(o,centers=3)
> attributes(clustering_k)
$names
[1] "cluster"      "centers"      "totss"
[4] "withinss"     "tot.withinss" "betweenss"
[7] "size"         "iter"         "ifault"

$class
[1] "kmeans"

> head(clustering_k$cluster)
[1] 1 1 1 1 1 1

> kc=table(clustering_k$cluster)
> kc

  1   2   3
100 100 100

>plot(o,pch=clustering_k$cluster,col=clustering_k$cluster,main="K-means clustering")
```



=> 위 표를 봤을 때 군집별로 잘 묶여져 있는 것을 확인할 수 있다. 객관적으로 확인하기 위해 평균제곱합으로 확인해보겠다.

```
> Kmeans_o = cbind(o,clustering_k$cluster)
> head(Kmeans_o)
         V1        V2 clustering_k$cluster
1 3.9191803 -2.538902                    1
2 0.5241628 -3.584091                    1
3 4.8818801 -4.082057                    1
4 1.6654923 -4.915347                    1
5 4.0784496 -3.655298                    1
6 4.1445951 -4.754012                    1

> mean(Kmeans_o[Kmeans_o[,3] == 1,][,1])
[1] 3.00901
> mean(Kmeans_o[Kmeans_o[,3] == 1,][,2])
[1] -4.074784
> mean(Kmeans_o[Kmeans_o[,3] == 2,][,1])
[1] -2.827515
> mean(Kmeans_o[Kmeans_o[,3] == 2,][,2])
[1] -3.975202
> mean(Kmeans_o[Kmeans_o[,3] == 3,][,1])
[1] 0.05701639
> mean(Kmeans_o[Kmeans_o[,3] == 3,][,2])
[1] -0.1372511

> sum((Kmeans_o[Kmeans_o[,3] == 1,][,1] - (3.00901))^2 + (Kmeans_o[Kmeans_o[,3] == 1,][,2] - (-4.074784))^2)
[1] 202.9
> sum((Kmeans_o[Kmeans_o[,3] == 2,][,1] - (-2.827515))^2 + (Kmeans_o[Kmeans_o[,3] == 2,][,2] - (-3.975202))^2)
[1] 224.0083
> sum((Kmeans_o[Kmeans_o[,3] == 3,][,1] - (0.05701639))^2 + (Kmeans_o[Kmeans_o[,3] == 3,][,2] - (-0.1372511))^2)
[1] 171.8208
```

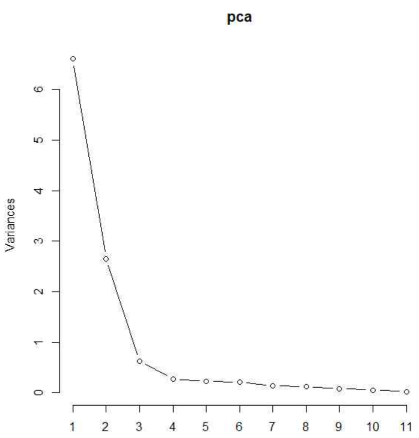
=> K-Means의 평균제곱합은 598.7291이 나왔다.
이로써 평균제곱합을 기준으로 봤을 때 K-Means가 가장 군잡을 잘 한 모델이라고 볼 수 있다.

-----------------------4번--------------------

```
> mt <- read.csv("C:/Users/sunni/OneDrive/바탕 화면/태양/`19년  1학기/다변량통계분석/mtcars_dat_주성분.csv",header = TRUE)
> head(mt)
                  X  mpg cyl disp  hp drat    wt  qsec vs am gear carb
1         Mazda RX4 21.0   6  160 110 3.90 2.620 16.46  0  1    4    4
2     Mazda RX4 Wag 21.0   6  160 110 3.90 2.875 17.02  0  1    4    4
3        Datsun 710 22.8   4  108  93 3.85 2.320 18.61  1  1    4    1
4    Hornet 4 Drive 21.4   6  258 110 3.08 3.215 19.44  1  0    3    1
5 Hornet Sportabout 18.7   8  360 175 3.15 3.440 17.02  0  0    3    2
6           Valiant 18.1   6  225 105 2.76 3.460 20.22  1  0    3    1

> ps = mt[,-1]
> head(ps)
   mpg cyl disp  hp drat    wt  qsec vs am gear carb
1 21.0   6  160 110 3.90 2.620 16.46  0  1    4    4
2 21.0   6  160 110 3.90 2.875 17.02  0  1    4    4
3 22.8   4  108  93 3.85 2.320 18.61  1  1    4    1
4 21.4   6  258 110 3.08 3.215 19.44  1  0    3    1
5 18.7   8  360 175 3.15 3.440 17.02  0  0    3    2
6 18.1   6  225 105 2.76 3.460 20.22  1  0    3    1

> pca <- prcomp(ps, scale = TRUE)
```

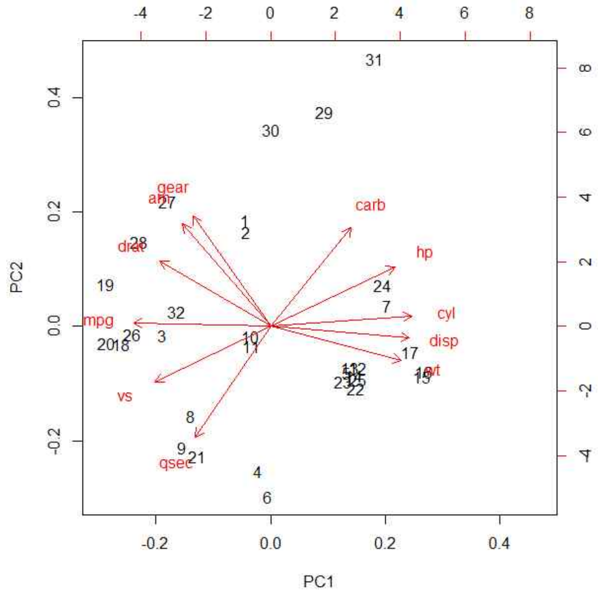
=> 전체 데이터의 단위가 다르기 때문에 표준화 하거나, 상관행렬로 PCA를 구할 수 있다.

```
> summary(pca)
Importance of components:
                          PC1    PC2     PC3     PC4     PC5     PC6
Standard deviation     2.5707 1.6280 0.79196 0.51923 0.47271 0.46000
Proportion of Variance 0.6008 0.2409 0.05702 0.02451 0.02031 0.01924
Cumulative Proportion  0.6008 0.8417 0.89873 0.92324 0.94356 0.96279
                          PC7     PC8    PC9    PC10   PC11
Standard deviation     0.3678 0.35057 0.2776 0.22811 0.1485
Proportion of Variance 0.0123 0.01117 0.0070 0.00473 0.0020
Cumulative Proportion  0.9751 0.98626 0.9933 0.99800 1.0000

> screeplot(pca,type = "lines", npcs = length(pca$sdev))
```



변수 2개만으로도 전체 데이터의 84%를 설명할 수 있다.

```
> biplot(pca)
```

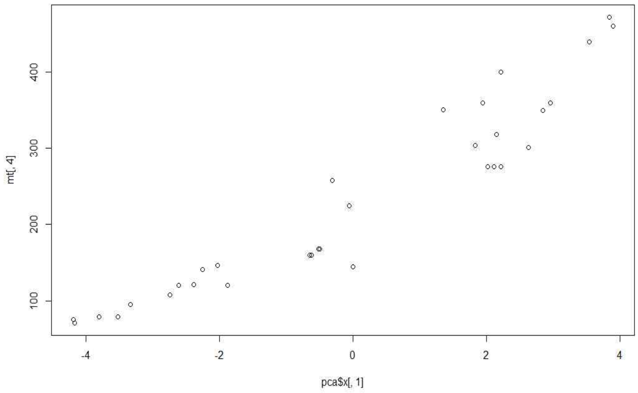


위 표를 봤을 때, 먼저 PC1을 봤을 때, 크게 0보다 큰 값 (carb, hp, cyl, disp, wt)과 0보다 작은 (gear, am, drat, mpg, vs, qsec)으로 나눌 수 있다. PC1을 기준으로 보면 0보다 큰 값들은 서로 양의 상관관계, 0보다 작은 값들도 마찬가지로 음의 상관관계가 있음을 볼 수 있다.

두 번째 변수 PC2을 보면 mpg, cyl, disp는 PC2의 축과 거의 수평에 가깝기 때문에 PC2에 거의 영향을 주지 않는다. 주로 플러스 쪽으로는 gear, carb, am의 영향이 크다는 것을 알 수 있고, 마이너스 쪽으로는 qsec의 영향이 가장 크다고 볼 수 있다.

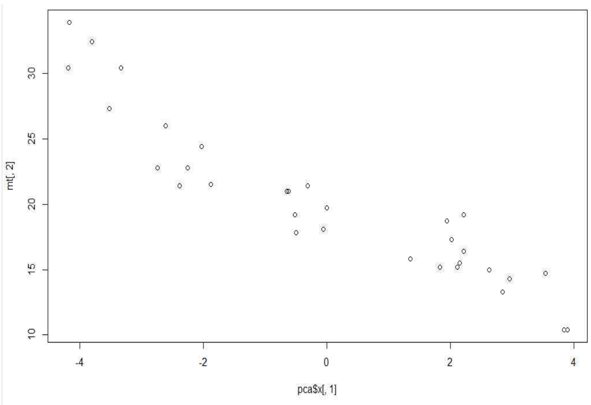
mpg : Miles/(US) gallon 연비
cyl : Number of cylinders 엔진의 기통수
disp : Displacement (cu.in.) 배기량 (cc, 변위)
hp : Gross horsepower 마력
drat : Rear axle ratio 뒤차축비
wt : Weight (1000 lbs) 무게
qsec : 1/4 mile time 1/4mile 도달시간
vs : V/S V engine / Straight engine
am : Transmission (0 = automatic, 1 = manual) 변속기어
gear : Number of forward gears 전진기어 개수
carb : Number of carburetors 기화기 개수

이를 확인하면 아래와 같다.
> plot(pca$x[,1], mt[,4])



=> PC1과 disp가 양의 상관관계가 있음을 볼 수 있다.

> plot(pca$x[,1], mt[,2])



=> PC1과 mpg가 음의 상관관계가 있음을 볼 수 있다.

> pca
Standard deviations (1, .., p=11):
 [1] 2.5706809 1.6280258 0.7919579 0.5192277 0.4727061 0.4599958
 [7] 0.3677798 0.3505730 0.2775728 0.2281128 0.1484736

Rotation (n x k) = (11 x 11):

	PC1	PC2	PC3	PC4	PC5
mpg	-0.3625305	0.01612440	-0.22574419	-0.022540255	0.10284468
cyl	0.3739160	0.04374371	-0.17531118	-0.002591838	0.05848381
disp	0.3681852	-0.04932413	-0.06148414	0.256607885	0.39399530
hp	0.3300569	0.24878402	0.14001476	-0.067676157	0.54004744
drat	-0.2941514	0.27469408	0.16118879	0.854828743	0.07732727
wt	0.3461033	-0.14303825	0.34181851	0.245899314	-0.07502912
qsec	-0.2004563	-0.46337482	0.40316904	0.068076532	-0.16466591
vs	-0.3065113	-0.23164699	0.42881517	-0.214848616	0.59953955
am	-0.2349429	0.42941765	-0.20576657	-0.030462908	0.08978128
gear	-0.2069162	0.46234863	0.28977993	-0.264690521	0.04832960
carb	0.2140177	0.41357106	0.52854459	-0.126789179	-0.36131875

위의 표를 봤을 때도 PC1과 0보다 큰 값들은 양의 상관관계, 0보다 작은 값들은 음의 상관관계가 있음을 알 수 있다.