Assignment 1

Zihan wei 2018124079

Task 1 – Hypothesis Testing

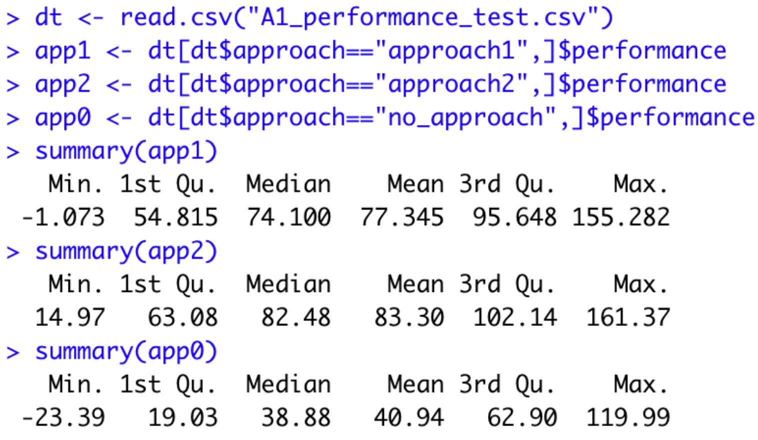
null hypothesis:

the two new learning approaches cannot effectively improve student learning performance.

alternative hypothesis:

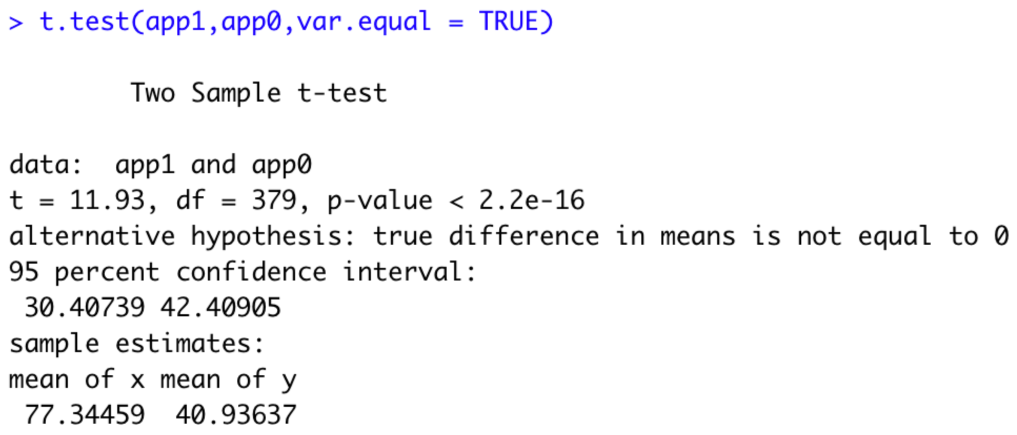
the two new learning approaches can effectively improve student learning performance.

First, divide the dataset into 3 parts



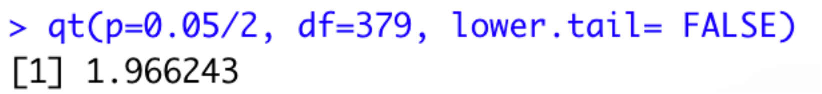
```
> dt <- read.csv("A1_performance_test.csv")
> app1 <- dt[dt$approach=="approach1",]$performance
> app2 <- dt[dt$approach=="approach2",]$performance
> app0 <- dt[dt$approach=="no_approach",]$performance
> summary(app1)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
 -1.073  54.815  74.100  77.345  95.648 155.282 
> summary(app2)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
  14.97   63.08   82.48   83.30  102.14  161.37 
> summary(app0)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
 -23.39   19.03   38.88   40.94   62.90  119.99 
```

Use t.test, for example:



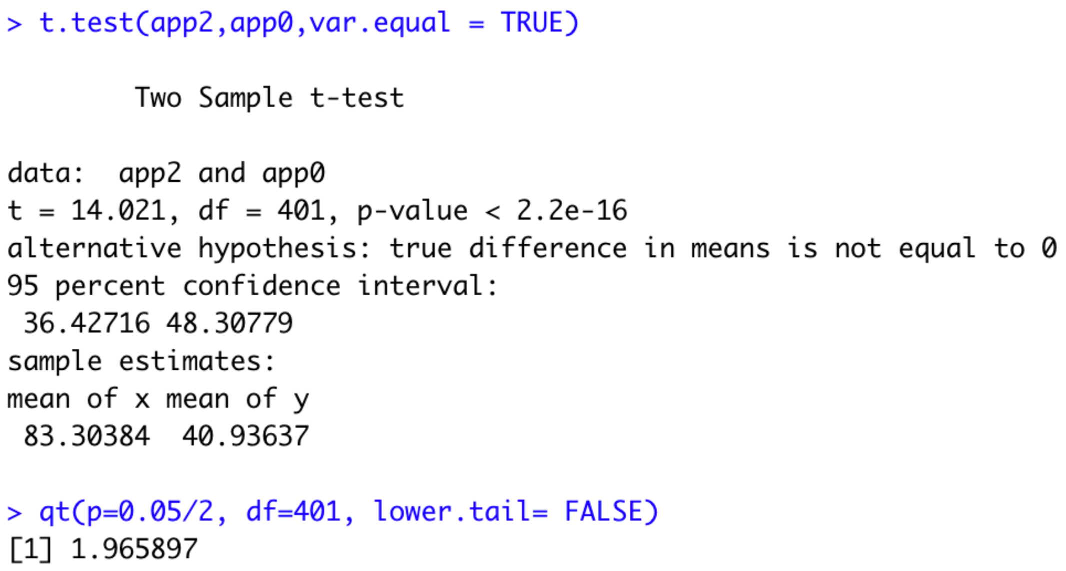
```
> t.test(app1,app0,var.equal = TRUE)

	Two Sample t-test

data:  app1 and app0
t = 11.93, df = 379, p-value < 2.2e-16
alternative hypothesis: true difference in means is not equal to 0
95 percent confidence interval:
 30.40739 42.40905
sample estimates:
mean of x mean of y 
 77.34459  40.93637 
```

We can see df=379, so that



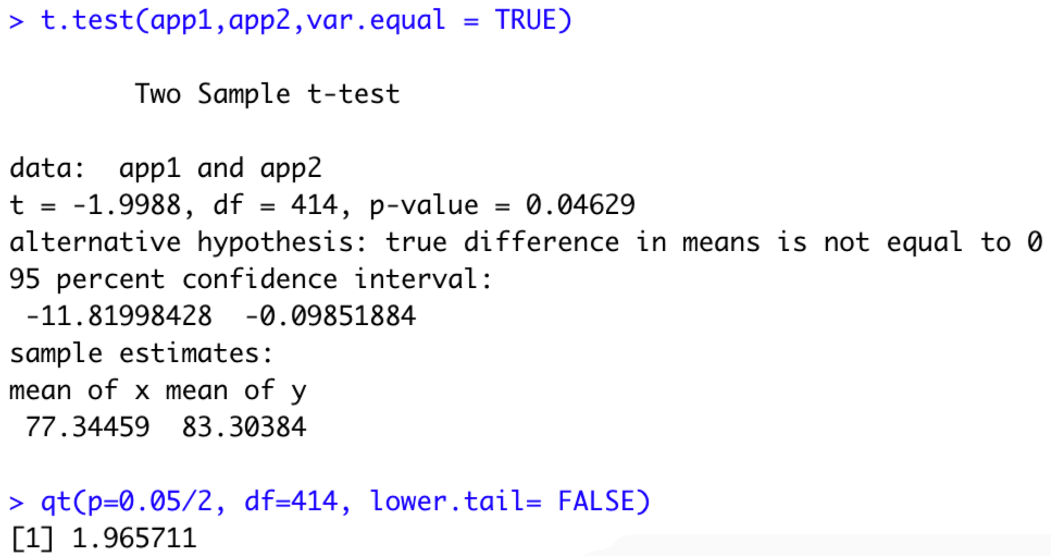
```
> qt(p=0.05/2, df=379, lower.tail= FALSE)
[1] 1.966243
```

|1-11.93|>1.966243, The original hypothesis will be denied.



```
> t.test(app2,app0,var.equal = TRUE)

        Two Sample t-test

data:  app2 and app0
t = 14.021, df = 401, p-value < 2.2e-16
alternative hypothesis: true difference in means is not equal to 0
95 percent confidence interval:
 36.42716 48.30779
sample estimates:
mean of x mean of y 
 83.30384  40.93637 

> qt(p=0.05/2, df=401, lower.tail= FALSE)
[1] 1.965897
```

The original hypothesis will be denied.
For approach 1 and approach 2:



```
> t.test(app1,app2,var.equal = TRUE)

        Two Sample t-test

data:  app1 and app2
t = -1.9988, df = 414, p-value = 0.04629
alternative hypothesis: true difference in means is not equal to 0
95 percent confidence interval:
 -11.81998428  -0.09851884
sample estimates:
mean of x mean of y 
 77.34459  83.30384 

> qt(p=0.05/2, df=414, lower.tail= FALSE)
[1] 1.965711
```

0.9988<1.965711, The original hypothesis will not be denied.

Conclusion :
the two new learning approaches can effectively improve student learning performance. In terms of improving student learning performance, the two approaches are not significantly different from each other.

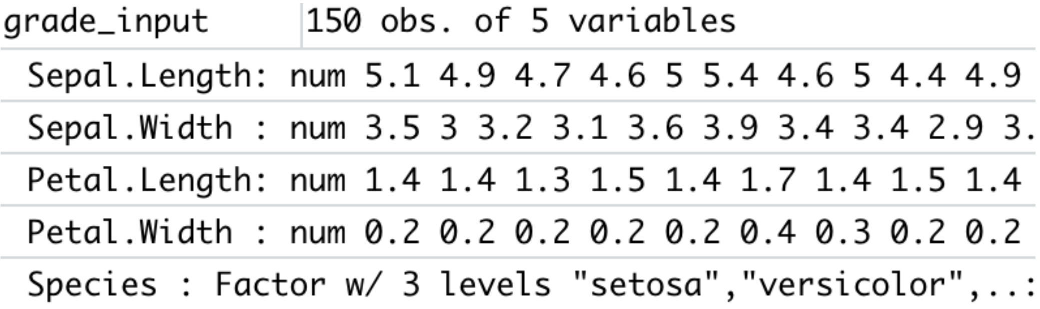
Code:

```
dt <- read.csv("A1_performance_test.csv")
app1 <- dt[dt$approach=="approach1",]$performance
app2 <- dt[dt$approach=="approach2",]$performance
```

```
app0 <- dt[dt$approach=="no_approach",]$performance

summary(app1)
summary(app2)
summary(app0)

t.test(app1,app0,var.equal = TRUE)
qt(p=0.05/2, df=379, lower.tail= FALSE)
t.test(app2,app0,var.equal = TRUE)
qt(p=0.05/2, df=401, lower.tail= FALSE)
t.test(app1,app2,var.equal = TRUE)
qt(p=0.05/2, df=414, lower.tail= FALSE)
```

Task 2 – Clustering

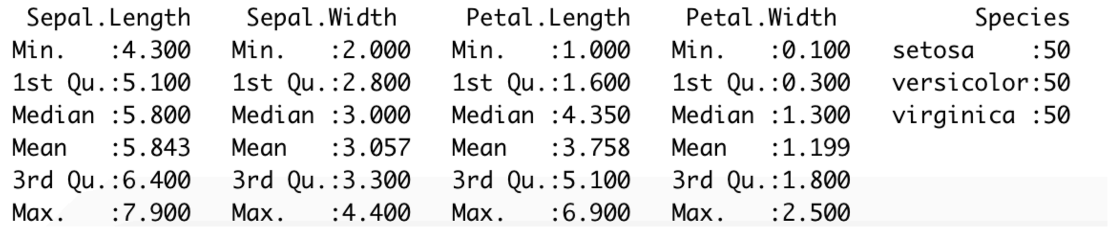
1. The Iris dataset includes 5 attributes as follow:



```
grade_input      150 obs. of 5 variables
 Sepal.Length: num 5.1 4.9 4.7 4.6 5 5.4 4.6 5 4.4 4.9
 Sepal.Width : num 3.5 3 3.2 3.1 3.6 3.9 3.4 3.4 2.9 3.
 Petal.Length: num 1.4 1.4 1.3 1.5 1.4 1.7 1.4 1.5 1.4
 Petal.Width : num 0.2 0.2 0.2 0.2 0.2 0.4 0.3 0.2 0.2
 Species : Factor w/ 3 levels "setosa","versicolor",..:
```

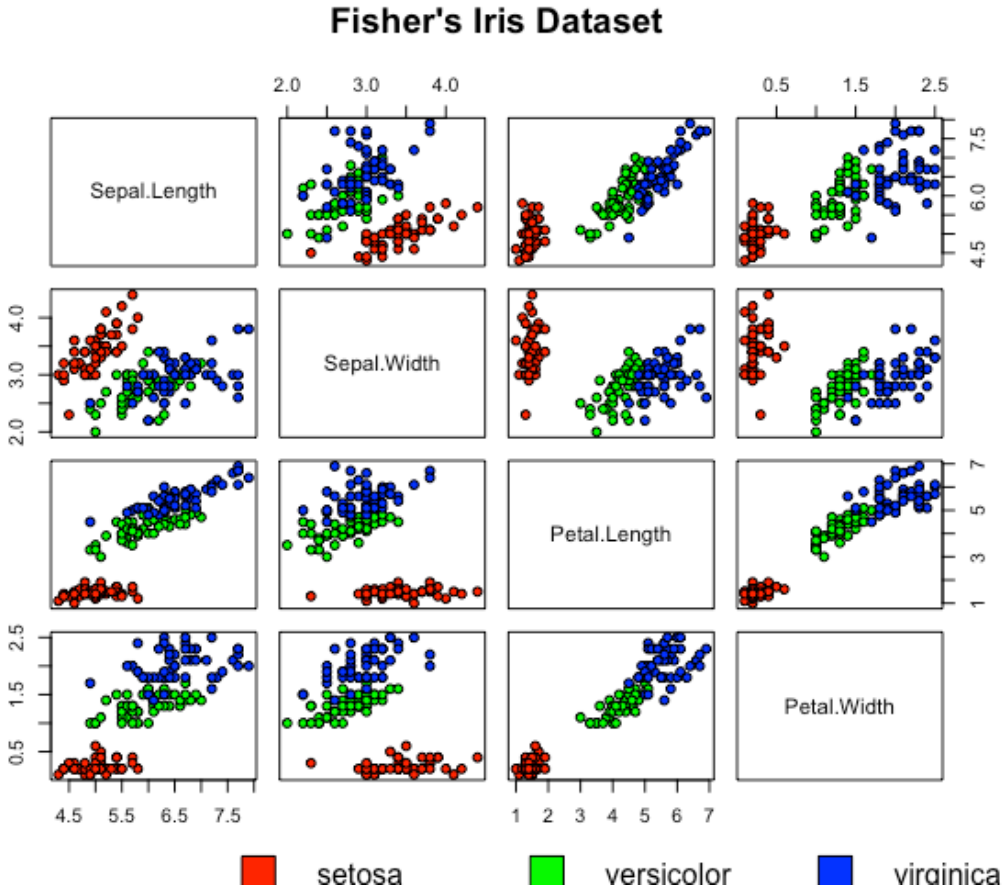
It has 150 data and each data belongs to one of 3 species: setosa, versicolor and virginica. It uses width and length of flowers' sepal and petal to describe species.

The summary is:

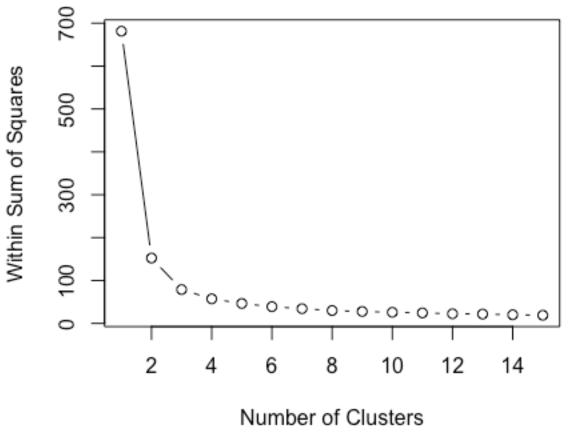


```
  Sepal.Length    Sepal.Width     Petal.Length    Petal.Width          Species  
 Min.   :4.300   Min.   :2.000   Min.   :1.000   Min.   :0.100   setosa    :50  
 1st Qu.:5.100   1st Qu.:2.800   1st Qu.:1.600   1st Qu.:0.300   versicolor:50  
 Median :5.800   Median :3.000   Median :4.350   Median :1.300   virginica :50  
 Mean   :5.843   Mean   :3.057   Mean   :3.758   Mean   :1.199                  
 3rd Qu.:6.400   3rd Qu.:3.300   3rd Qu.:5.100   3rd Qu.:1.800                  
 Max.   :7.900   Max.   :4.400   Max.   :6.900   Max.   :2.500                  
```

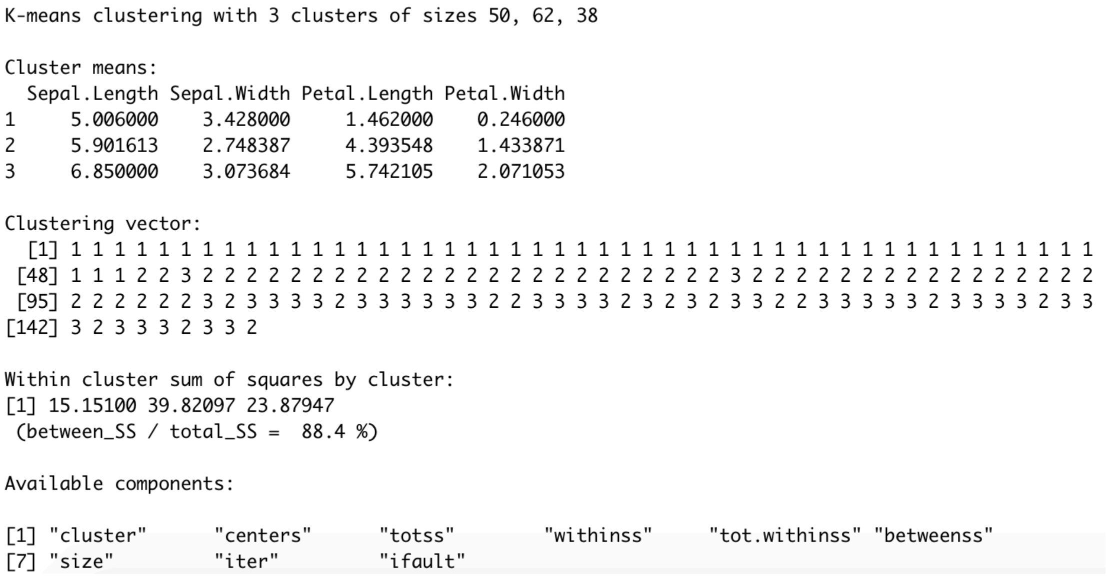
2. The figure is:



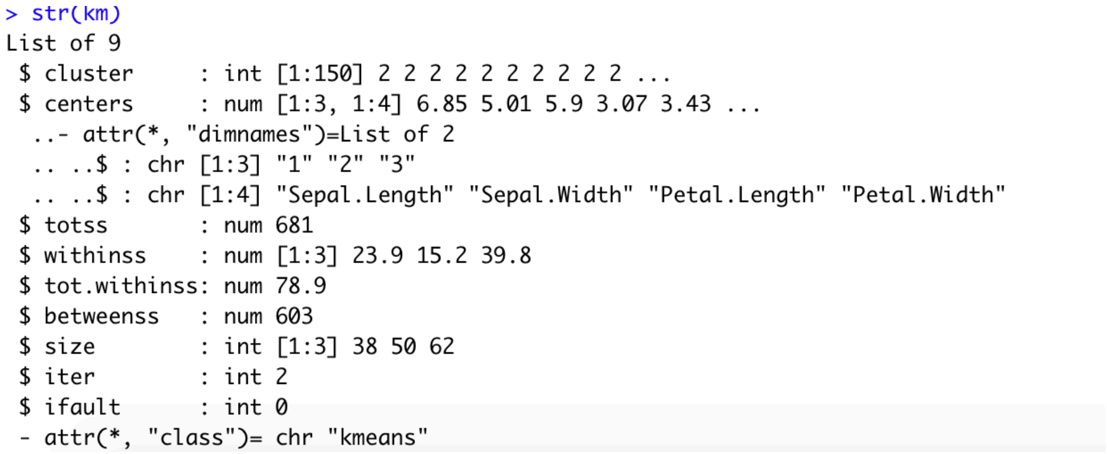
3. First, we need to do data preprocessing: remove species from the dataset.
To determine the appropriate value of k, we use the k-means clustering algorithm to calculate the clustering results for k = 1, 2, ..., 15. Calculate WSS for each k value. Then we get the figure:



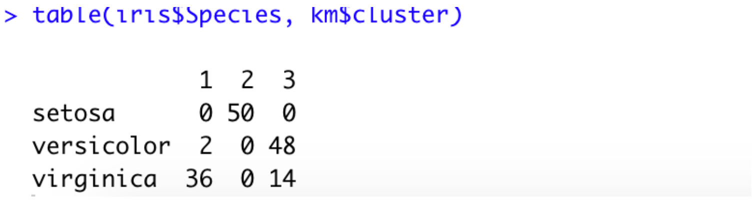
When k>3, the change in WSS tends to be linear. Therefore, the k-means analysis will select k=3.



```
K-means clustering with 3 clusters of sizes 50, 62, 38

Cluster means:
  Sepal.Length Sepal.Width Petal.Length Petal.Width
1     5.006000    3.428000     1.462000    0.246000
2     5.901613    2.748387     4.393548    1.433871
3     6.850000    3.073684     5.742105    2.071053

Clustering vector:
  [1] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
 [48] 1 1 1 2 2 3 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 3 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
 [95] 2 2 2 2 2 2 3 2 3 3 3 3 2 3 3 3 3 3 3 2 2 3 3 3 3 2 3 2 3 2 3 3 2 2 3 3 3 3 3 2 3 3 3 3 2 3 3
[142] 3 2 3 3 3 2 3 3 2

Within cluster sum of squares by cluster:
[1] 15.15100 39.82097 23.87947
 (between_SS / total_SS =  88.4 %)

Available components:

[1] "cluster"      "centers"      "totss"        "withinss"     "tot.withinss" "betweenss"
[7] "size"         "iter"         "ifault"
```

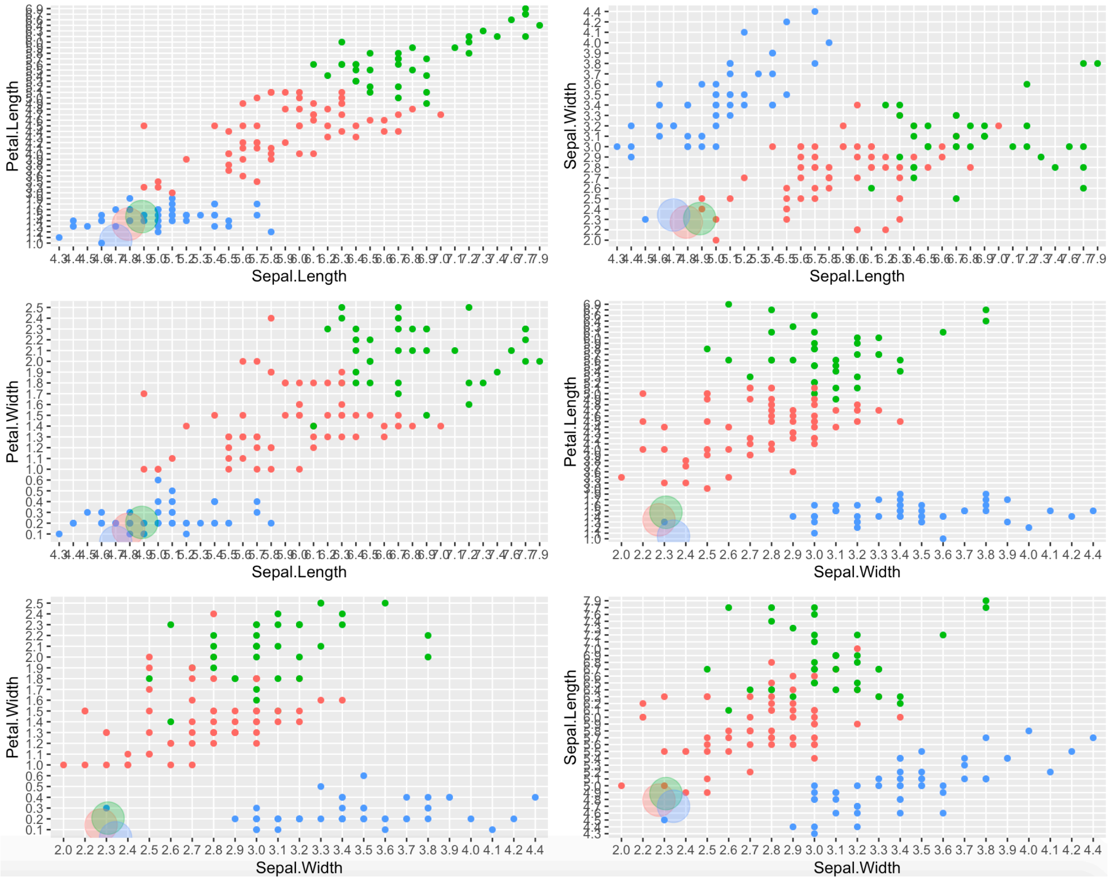


```
> str(km)
List of 9
 $ cluster     : int [1:150] 2 2 2 2 2 2 2 2 2 2 ...
 $ centers     : num [1:3, 1:4] 6.85 5.01 5.9 3.07 3.43 ...
  ..- attr(*, "dimnames")=List of 2
  .. ..$ : chr [1:3] "1" "2" "3"
  .. ..$ : chr [1:4] "Sepal.Length" "Sepal.Width" "Petal.Length" "Petal.Width"
 $ totss       : num 681
 $ withinss    : num [1:3] 23.9 15.2 39.8
 $ tot.withinss: num 78.9
 $ betweenss   : num 603
 $ size        : int [1:3] 38 50 62
 $ iter        : int 2
 $ ifault      : int 0
 - attr(*, "class")= chr "kmeans"
```



```
> table(iris$Species, km$cluster)

              1  2  3
  setosa      0 50  0
  versicolor  2  0 48
  virginica  36  0 14
```

4. Results after visualizing the data:



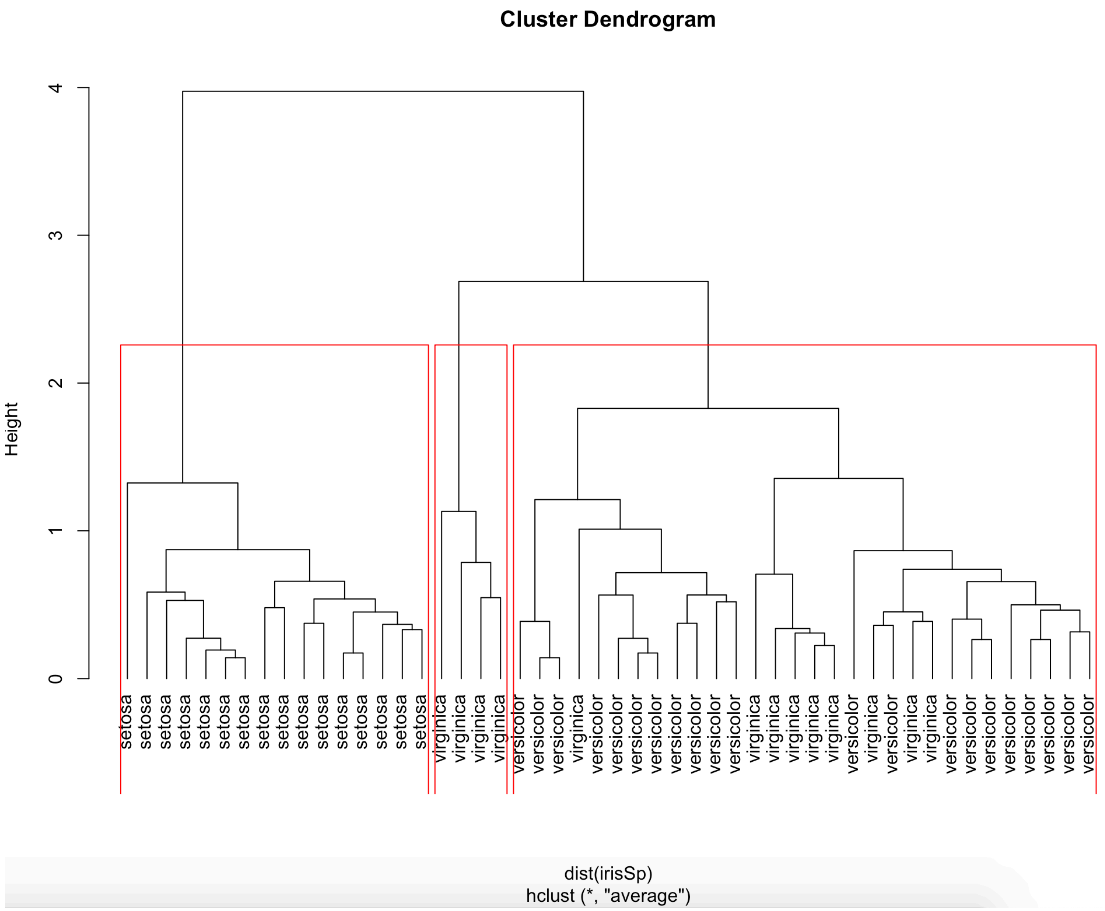
We find that the centroids appear to be too close to each other.

Most of points in different clusters are well separated from each other. But there are still some points located in other clusters.

There are no clusters have only a few points.

5. Randomly extract 50 data and use hierarchical agglomerative clustering:

```
idx <- sample(1:dim(iris)[1], 50)
irisSp <- iris[idx,]
irisSp$Species <- NULL

hc <- hclust(dist(irisSp), method="ave")
plot(hc, hang = -1, labels=iris$Species[idx])

rect.hclust(hc, k=3)
groups <- cutree(hc, k=3)
```



Code:

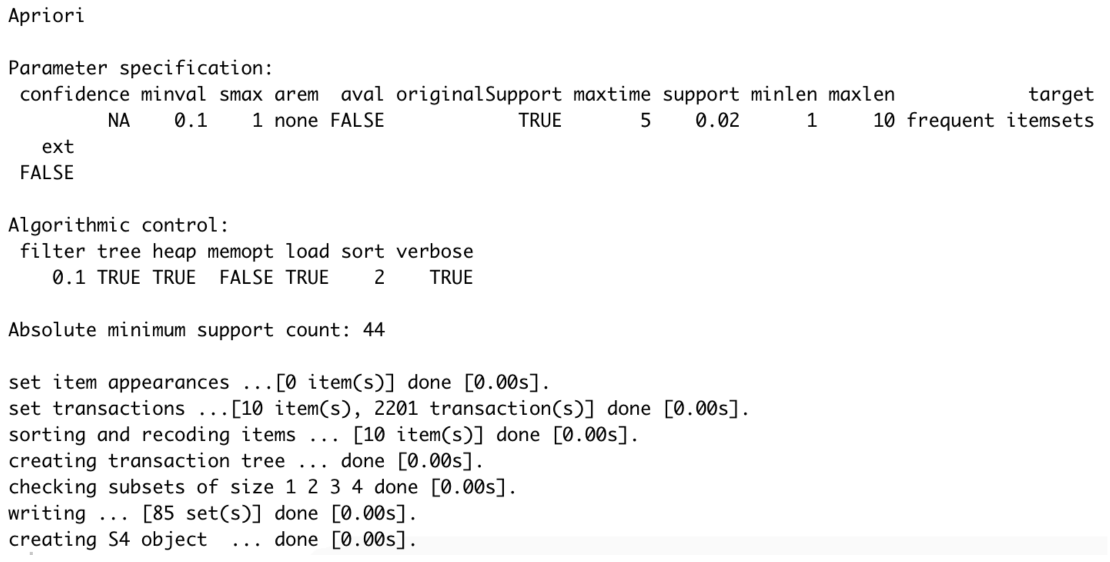
```
library(cluster)
library(plyr)
library(ggplot2)
library(cluster)
library(lattice)
library(graphics)
library(grid)
library(gridExtra)

grade_input = as.data.frame(iris)
kmdata_orig = as.matrix(grade_input[,c("Sepal.Length", "Sepal.Width", "Petal.Length",
"Petal.Width", "Species")])
summary(grade_input)

colors <- c("red", "green", "blue")
pairs(iris[1:4], main = "Fisher's Iris Dataset", pch = 21, bg = colors[unclass(iris$Species)] )
# set graphical parameter to clip plotting to the figure region
par(xpd = TRUE)
# add legend
legend(0.2, 0.02, horiz = TRUE, as.vector(unique(iris$Species)), fill = colors, bty = "n")
```

```
# remove species
kmdata <- kmdata_orig[,1:4]

wss <- numeric(15)
for(k in 1:15) wss[k] <- sum(kmeans(kmdata, centers = k, nstart = 25)$withinss)

plot(1:15, wss, type = "b", xlab = "Number of Clusters", ylab = "Within Sum of Squares")

km = kmeans(kmdata, 3)
km
str(km) # data structure
table(iris$Species, km$cluster)

dt = as.data.frame(kmdata_orig[,1:4])
dt$cluster = factor(km$cluster)
centers = as.data.frame(km$centers)
g1 =ggplot(data=dt, aes(x=Sepal.Length, y=Petal.Length, color=cluster ))+geom_point() +
geom_point(data=centers,aes(x=Sepal.Length,y=Petal.Length, color=as.factor(c(1,2,3))),size=10,
alpha=.3, show.legend = FALSE)

g2 =ggplot(data=dt, aes(x=Sepal.Length, y=Sepal.Width, color=cluster ))+geom_point() +
geom_point(data=centers,aes(x=Sepal.Length, y=Sepal.Width, color=as.factor(c(1,2,3))),size=10,
alpha=.3, show.legend = FALSE)

g3 =ggplot(data=dt, aes(x=Sepal.Length, y=Petal.Width, color=cluster ))+geom_point() +
geom_point(data=centers,aes(x=Sepal.Length,y=Petal.Width, color=as.factor(c(1,2,3))),size=10,
alpha=.3, show.legend = FALSE)

g4 =ggplot(data=dt, aes(x=Sepal.Width, y=Petal.Length, color=cluster ))+geom_point() +
geom_point(data=centers,aes(x=Sepal.Width,y=Petal.Length, color=as.factor(c(1,2,3))),size=10,
alpha=.3, show.legend = FALSE)

g5 =ggplot(data=dt, aes(x=Sepal.Width, y=Petal.Width, color=cluster ))+geom_point() +
geom_point(data=centers,aes(x=Sepal.Width,y=Petal.Width, color=as.factor(c(1,2,3))),size=10,
alpha=.3, show.legend = FALSE)

g6 =ggplot(data=dt, aes(x=Sepal.Width, y=Sepal.Length, color=cluster ))+geom_point() +
geom_point(data=centers,aes(x=Sepal.Width,y=Sepal.Length, color=as.factor(c(1,2,3))),size=10,
alpha=.3, show.legend = FALSE)

grid.arrange(arrangeGrob(g1 + theme(legend.position="none"), g2 +
theme(legend.position="none"),
                         g3 + theme(legend.position="none"), g4 +
theme(legend.position="none"),
                         g5 + theme(legend.position="none"), g6 +
theme(legend.position="none"),
                         ncol=2))
```

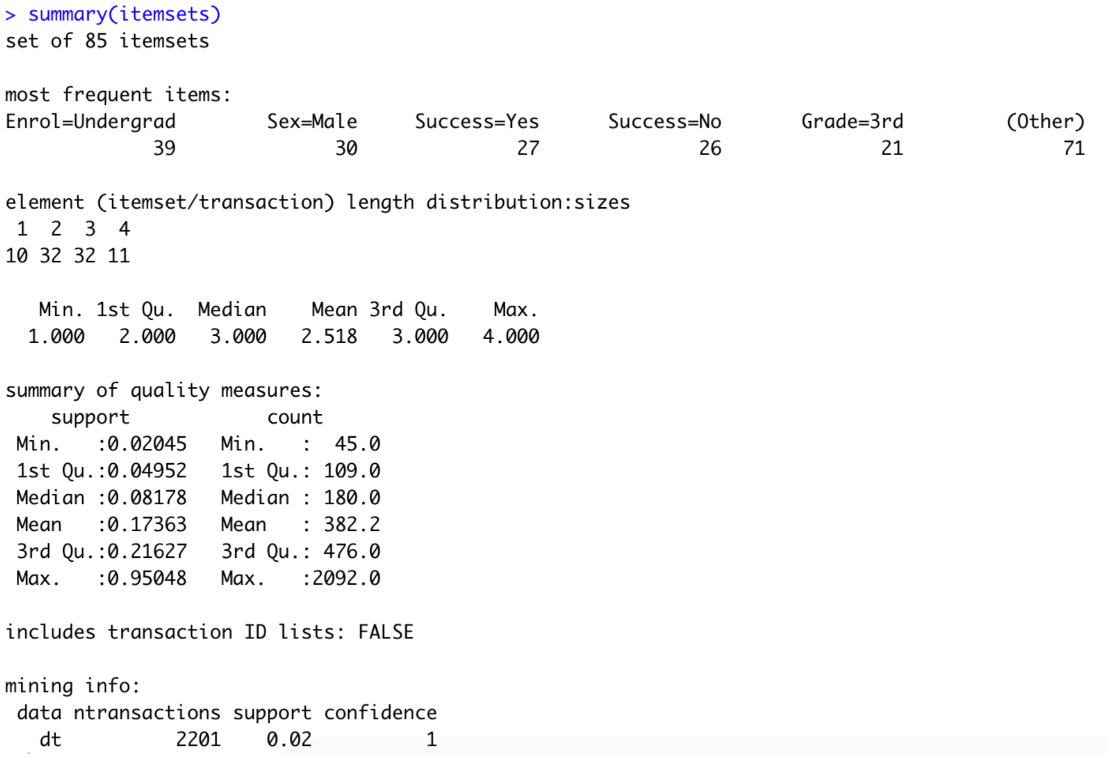
```
idx <- sample(1:dim(iris)[1], 50)
irisSp <- iris[idx,]
irisSp$Species <- NULL

hc <- hclust(dist(irisSp), method="ave")
plot(hc, hang = -1, labels=iris$Species[idx])

rect.hclust(hc, k=3)
groups <- cutree(hc, k=3)
```

Task 3 – Association Rule

1. Use support 0.02:



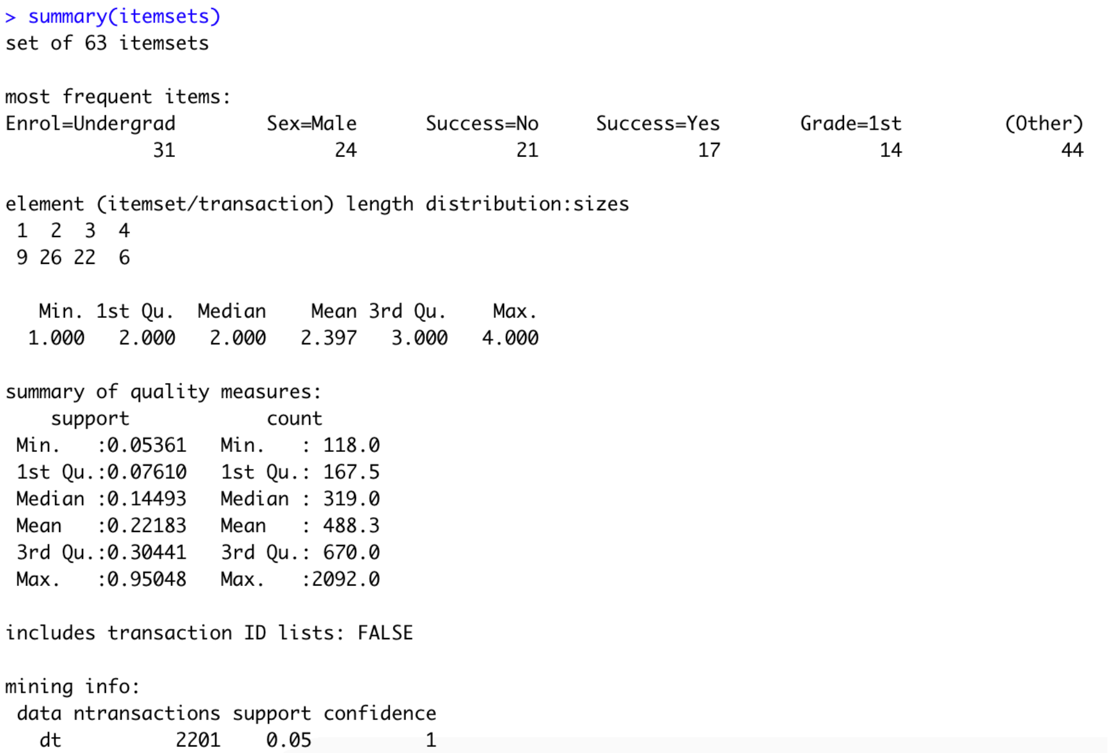
```
Apriori

Parameter specification:
 confidence minval smax arem  aval originalSupport maxtime support minlen maxlen            target
         NA    0.1    1 none FALSE            TRUE       5    0.02      1     10 frequent itemsets
   ext
 FALSE

Algorithmic control:
 filter tree heap memopt load sort verbose
    0.1 TRUE TRUE  FALSE TRUE    2    TRUE

Absolute minimum support count: 44

set item appearances ...[0 item(s)] done [0.00s].
set transactions ...[10 item(s), 2201 transaction(s)] done [0.00s].
sorting and recoding items ... [10 item(s)] done [0.00s].
creating transaction tree ... done [0.00s].
checking subsets of size 1 2 3 4 done [0.00s].
writing ... [85 set(s)] done [0.00s].
creating S4 object  ... done [0.00s].
```



```
> summary(itemsets)
set of 85 itemsets

most frequent items:
Enrol=Undergrad         Sex=Male      Success=Yes       Success=No        Grade=3rd          (Other) 
             39               30               27               26               21               71 

element (itemset/transaction) length distribution:sizes
 1  2  3  4 
10 32 32 11 

   Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
  1.000   2.000   3.000   2.518   3.000   4.000 

summary of quality measures:
    support             count       
 Min.   :0.02045   Min.   :  45.0  
 1st Qu.:0.04952   1st Qu.: 109.0  
 Median :0.08178   Median : 180.0  
 Mean   :0.17363   Mean   : 382.2  
 3rd Qu.:0.21627   3rd Qu.: 476.0  
 Max.   :0.95048   Max.   :2092.0  

includes transaction ID lists: FALSE 

mining info:
 data ntransactions support confidence
   dt          2201    0.02          1
```

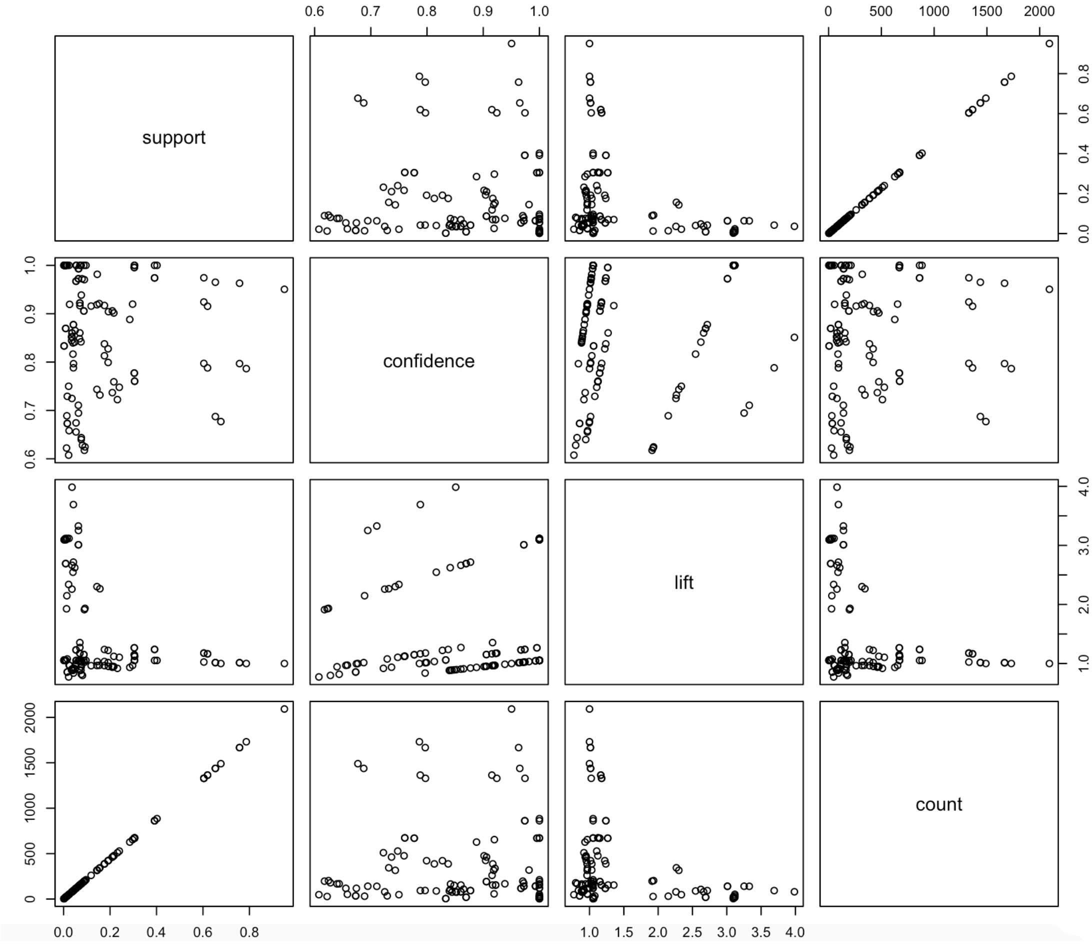
Use support 0.05:

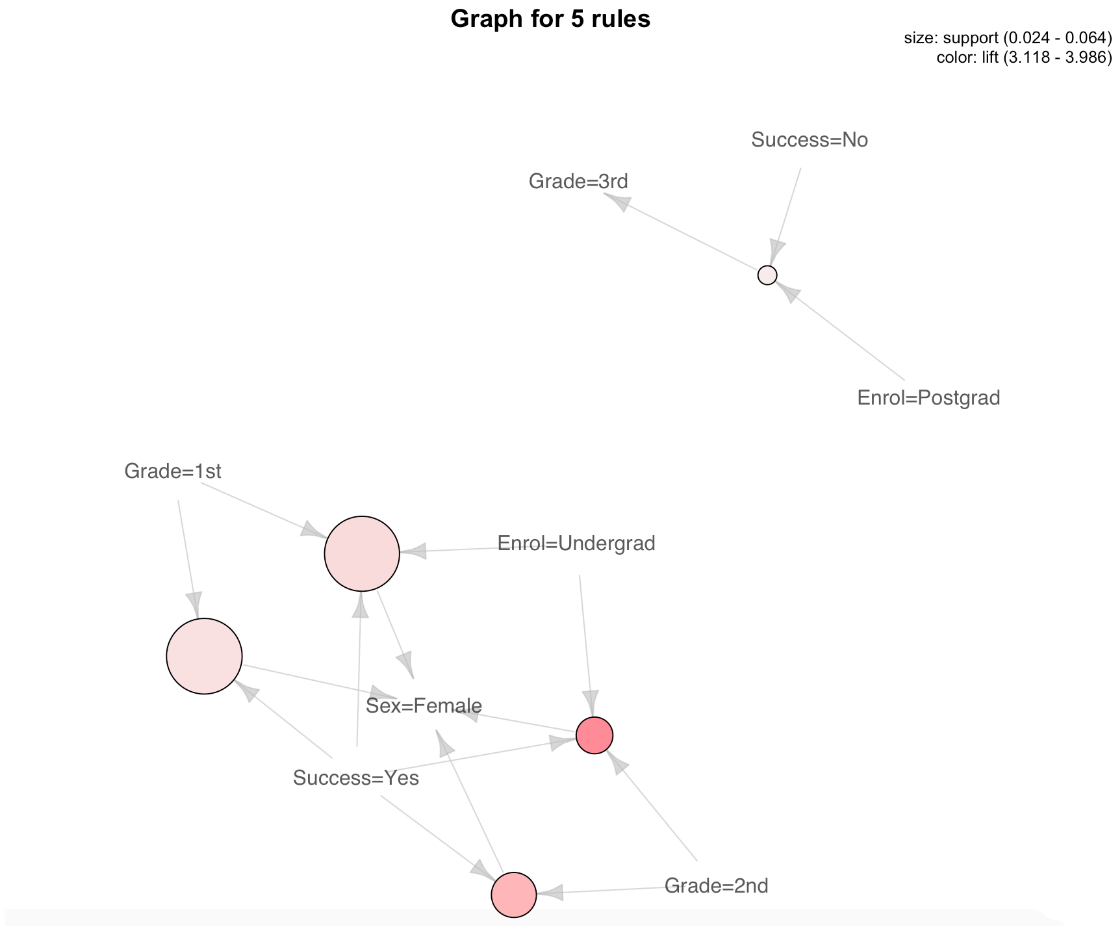
```
Apriori

Parameter specification:
 confidence minval smax arem  aval originalSupport maxtime support minlen maxlen            target
         NA    0.1    1 none FALSE            TRUE       5    0.05      1     10 frequent itemsets
   ext
 FALSE

Algorithmic control:
 filter tree heap memopt load sort verbose
    0.1 TRUE TRUE  FALSE TRUE    2    TRUE

Absolute minimum support count: 110 

set item appearances ...[0 item(s)] done [0.00s].
set transactions ...[10 item(s), 2201 transaction(s)] done [0.00s].
sorting and recoding items ... [9 item(s)] done [0.00s].
creating transaction tree ... done [0.00s].
checking subsets of size 1 2 3 4 done [0.00s].
writing ... [63 set(s)] done [0.00s].
creating S4 object  ... done [0.00s].
```



```
> summary(itemsets)
set of 63 itemsets

most frequent items:
Enrol=Undergrad        Sex=Male      Success=No     Success=Yes       Grade=1st         (Other)
             31              24              21              17              14              44

element (itemset/transaction) length distribution:sizes
 1  2  3  4
 9 26 22  6

   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
  1.000   2.000   2.000   2.397   3.000   4.000

summary of quality measures:
    support            count
 Min.   :0.05361   Min.   : 118.0
 1st Qu.:0.07610   1st Qu.: 167.5
 Median :0.14493   Median : 319.0
 Mean   :0.22183   Mean   : 488.3
 3rd Qu.:0.30441   3rd Qu.: 670.0
 Max.   :0.95048   Max.   :2092.0

includes transaction ID lists: FALSE

mining info:
 data ntransactions support confidence
   dt          2201    0.05          1
```

2.
3. show the relationship among support, confidence and lift





Code:

```
library('arules')

library('arulesViz')



dt <- read.csv("A1_success_data.csv")



itemsets <- apriori(dt, parameter = list(minlen=1, maxlen=10, support=0.02, target="frequent itemsets"))

summary(itemsets)

itemsets <- apriori(dt, parameter = list(minlen=1, maxlen=10, support=0.05, target="frequent itemsets"))
```

```
summary(itemsets)


rules <- apriori(dt, parameter = list(support=0.001, confidence=0.6, target="rules"))

plot(rules)

plot(rules@quality)


confidentRules <- rules[quality(rules)$confidence>0.9]

plot(confidentRules, method="matrix", measure=c("lift", "confidence"),
control=list(reorder=TRUE))


highLiftRules <- head(sort(rules, by="lift"), 5)

plot(highLiftRules, method="graph", control=list(type="items"))
```