**IE 7275 Data Mining in Engineering**

**Homework 1**

**Deadline 9/21**

**Jing Li**

**Problem1 (twitter account)**

## load the data file

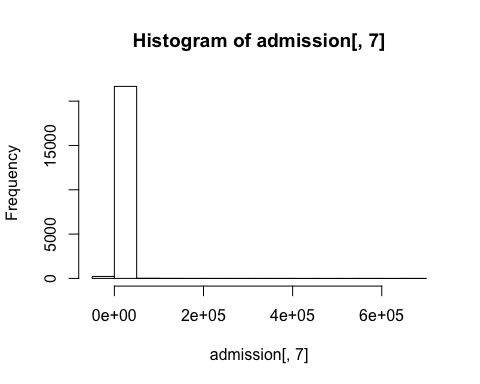
> hw1 <- read.csv("/Users/jingli/Desktop/M01\_quasi\_twitter.csv")

> View(hw1)

1. **how are the data distributed for friends\_count**

## Create the distribution of friends\_count

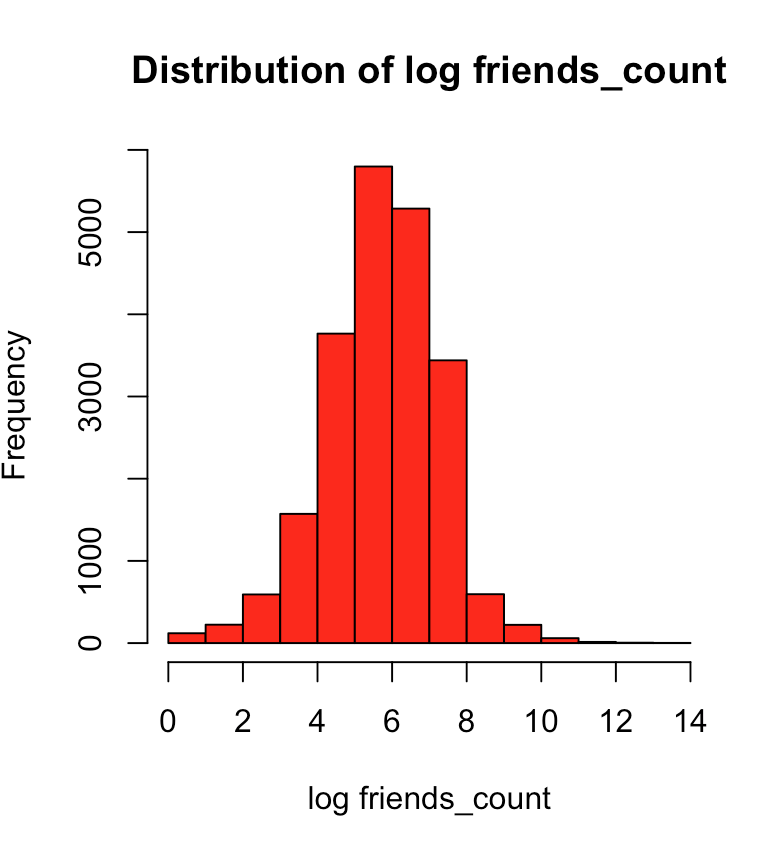
hist(hw1$friends\_count)



It is hard to tell the distribution from the histogram of original data. After plotting the histogram of log of data the distribution is close to Normal Distribution

install. packages(“logging”).

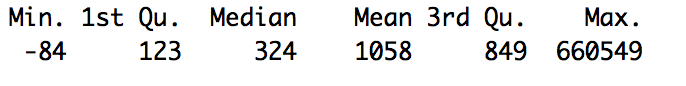
hist(log(hw1$friends\_count), xlab = "log friends\_count", col = "red", main = "Distribution of log friends\_count")



Compute the summery statistics ( min, 1Q mean, mean, median, 3Q, max ) on friend\_count

##Compute the summery statistics on friends\_count?

summary(hw1$friends\_count)

****

1. **How are the data quality in friend\_count variable? Interpret your answer.**

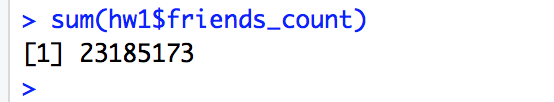
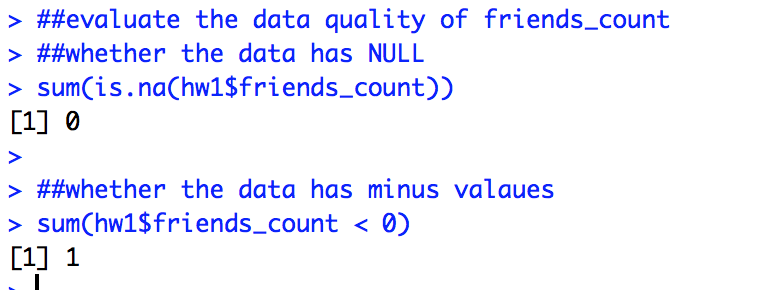
##evaluate the data quality of friends\_count

##whether the data has NULL

sum(is.na(hw1$friends\_count))

##whether the data has minus valaues

sum(hw1$friends\_count < 0)

****

Answer: Based on the above outcomes, there is only one negative value. Comparing to the total number of the data, this value can be ignored. So the data quality of friends\_counts is good.

1. **Produce a 3D scatter plot with highlighting**

## d. Produce a 3D scatter plot with highlighting

install.packages("scatterplot3d")

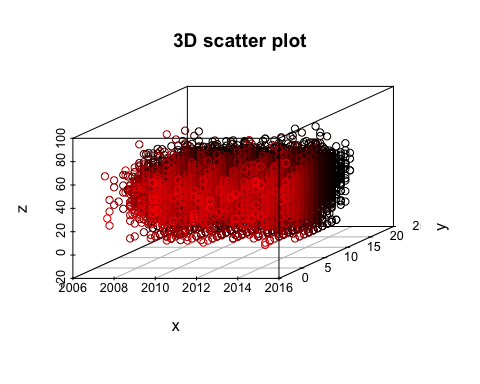
library(scatterplot3d)

x <- hw1$created\_at\_year

y <- hw1$education

z <- hw1$age

scatterplot3d(x, y, z, highlight.3d = TRUE, main = "3D scatter plot")



1. **Pie chart with percentage**

## e. pie chart with percentages

library(plotrix)

par(mfrow=c(1,2))

twitter\_account <- c(650,1000,900,300,14900)

twitter\_country <- c("UK","Canada","India","Australia","US")

percent <- round(twitter\_account/sum(twitter\_account)\*100)

twitter\_country <- paste(twitter\_country, percent)# add percents to labels

twitter\_country <- paste(twitter\_country, "%", sep ="")# ad % to labels

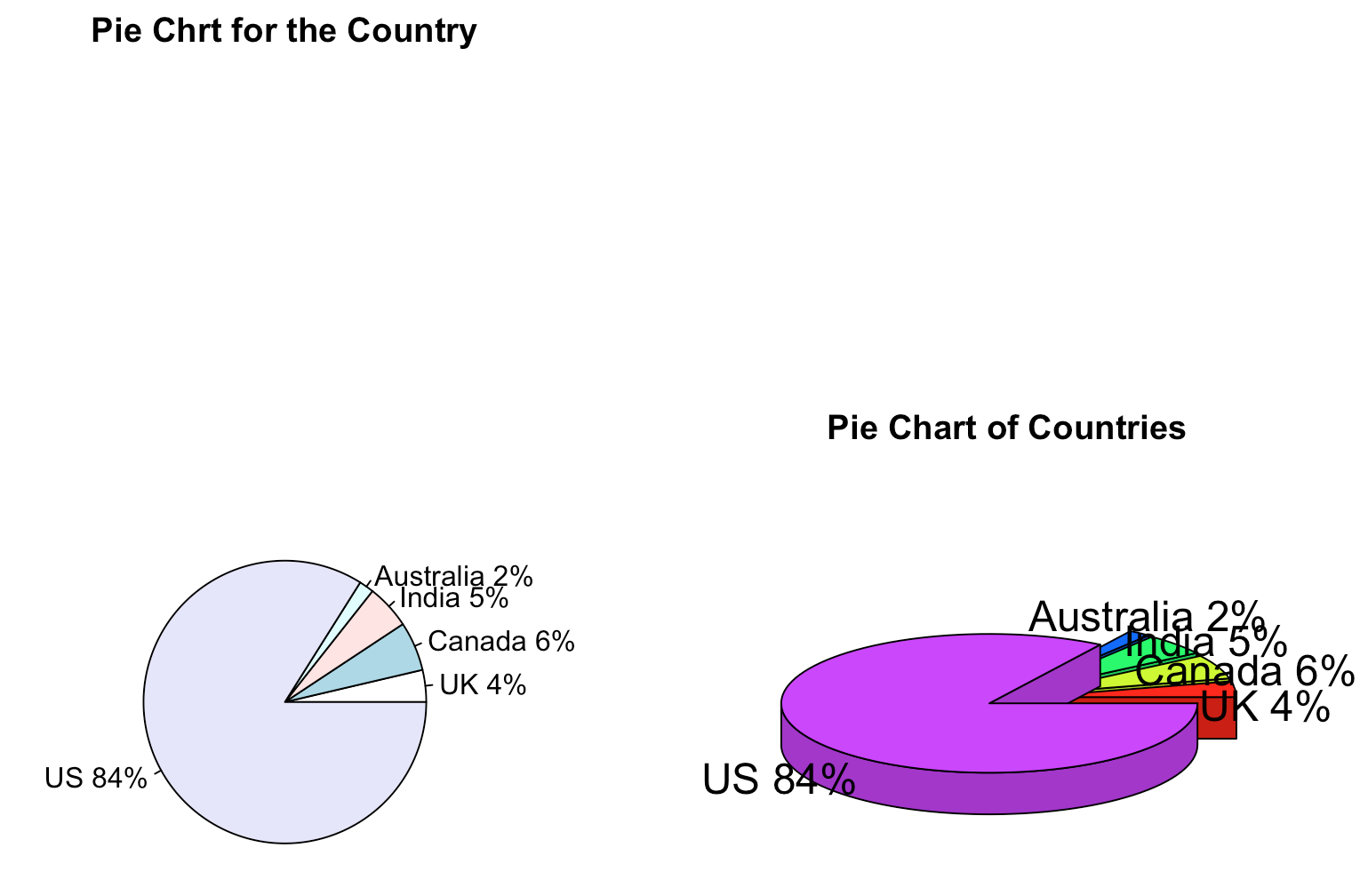
pie(twitter\_account,labels = twitter\_country,main = "Pie Chrt for the Country")

##3D pie chart

install.packages("plotrix")

library(plotrix)

pie3D(twitter\_account,labels = twitter\_country, explode = 0.1,main = "Pie Chart of Countries")

****

1. **Density plot**

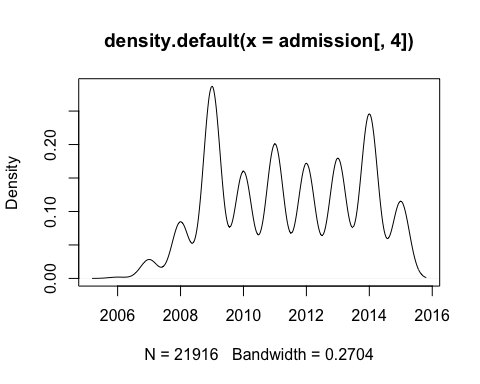
## f. kernel density plot

par(mfrow=c(1,1))

kernel\_density <- density(hw1$created\_at\_year)

plot(kernel\_density)

## in 2006, there is a few twitter user.Since 2008,the user of twitter start to increase. It reaches the maximum number,around 0.28, in 2009.The curve of density of created user is wavy from 2010 to 2015. The tendency is decreasing from 2015 to 2016



The number of twitter users grows rapidly during 2008 and 2009, and peaked at around 0.27 in 2019.Then it fluctuated between 2012 and 2014, After which it dropped dramatically to the lowest level in 2016.

**Problem2 (Cereals Analysis)**

a.

Answer: The quantitative variables are calories, protein, fat, sodium, fiber, carbo, sugars, potassium, vitamins, weight, cups and rating.

The ordinal variable is shelf.

The nominal variables are name, mfr, and type.

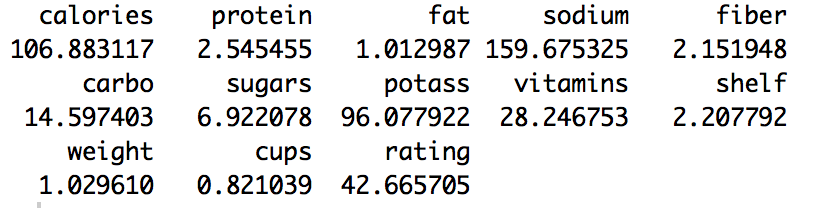
**b.**

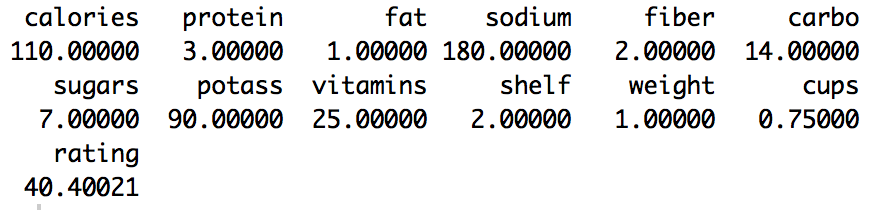
##load data cereals.csv

cereals<- read.csv("/Users/jingli/Desktop/Cereals.csv")

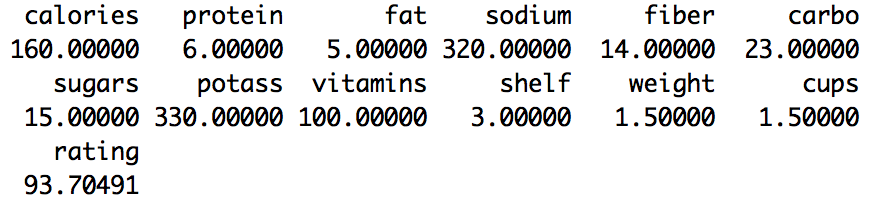
View(cereals)

sapply(cereals[,4:16],mean,na.rm=TRUE)

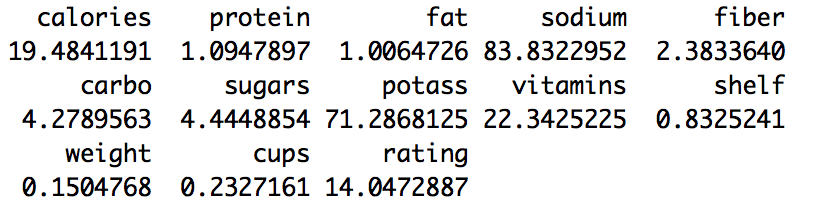
****

> sapply(cereals[,4:16],median,na.rm=TRUE)****

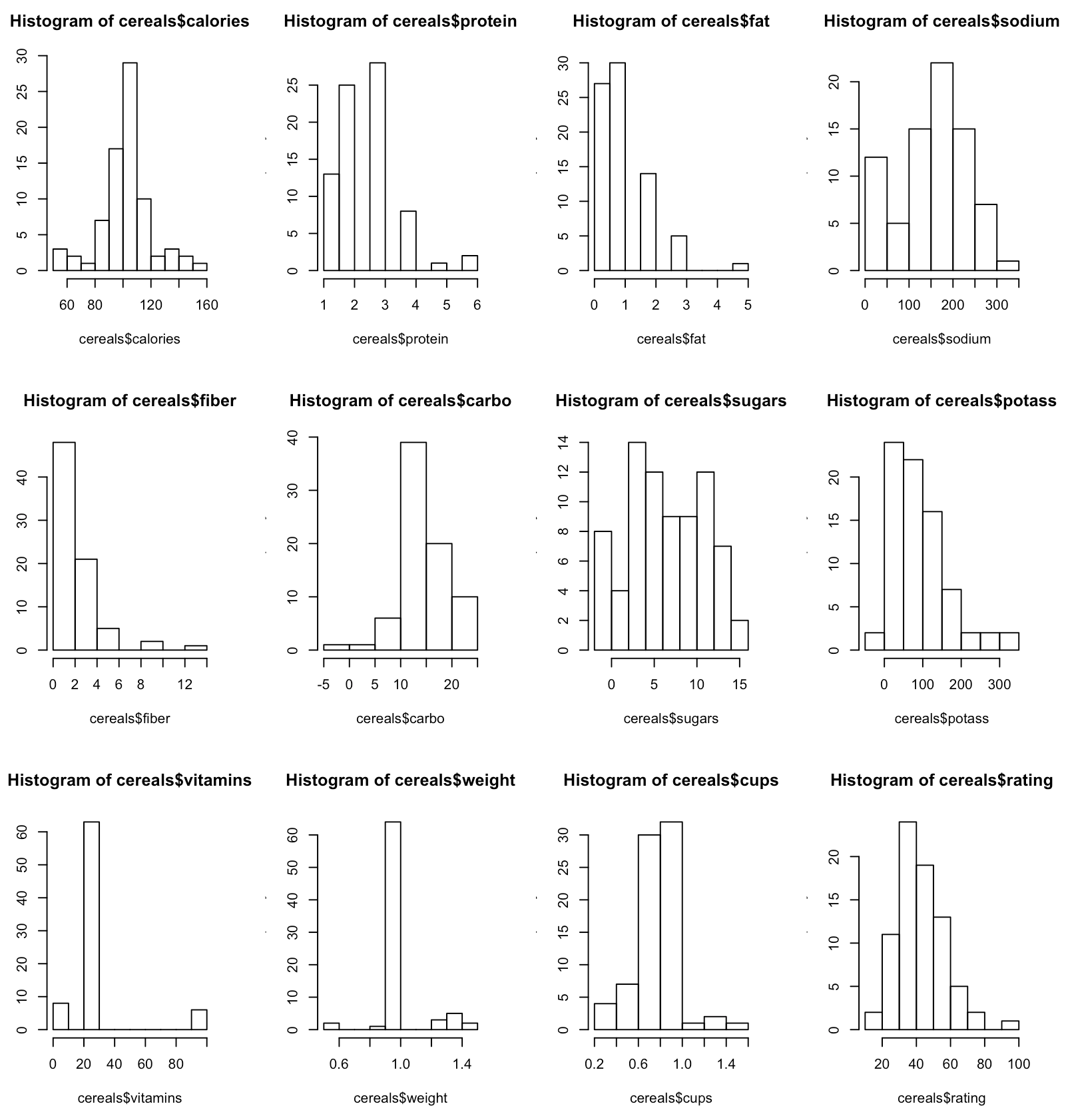
sapply(cereals[,4:16],max,na.rm=TRUE)

****

sapply(cereals[,4:16],sd,na.rm=TRUE)



1. **summary of quantities values except name, mfr shelf**

****

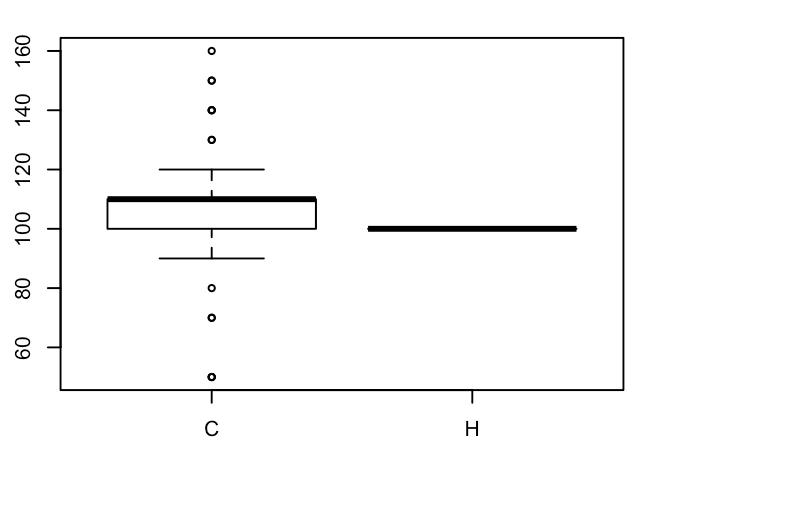
Values have largest variability?

Potassium has the highest variability and sodium has the second highest variability.

Fat Fiber Protein Carbo Potass Vitamin Shelf are the variables that seem to be skewed.

From the histograms and the table, we see that the extreme variables are Protein Fiber Fat Vitamins Weight rating.

boxplot(cereals$calories~cereals$type)

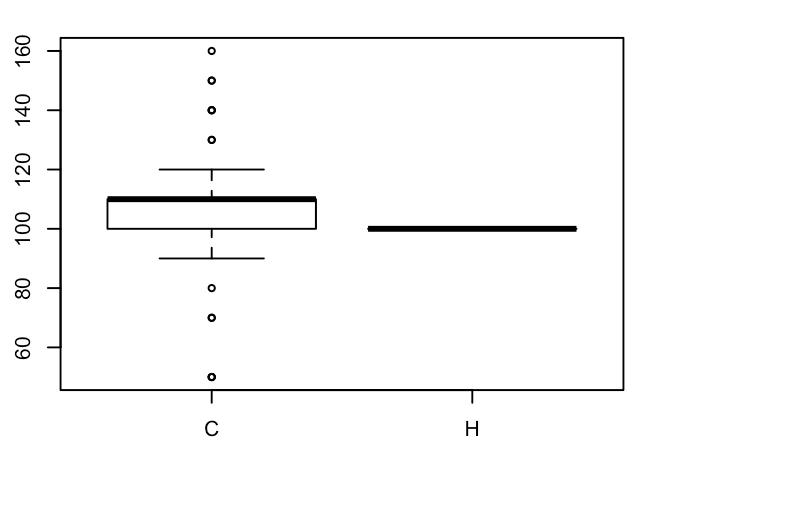


The boxplots shows cold cereals have some variables that have extreme calories.

The hot cereal has no variable.

**e.**

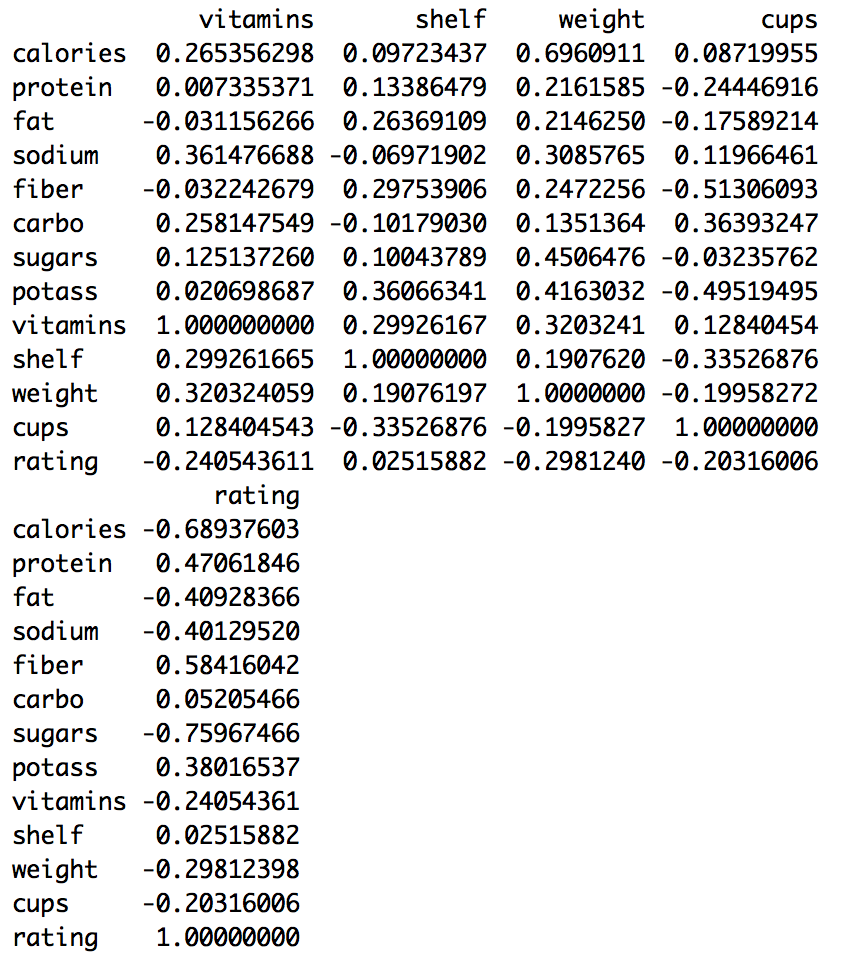
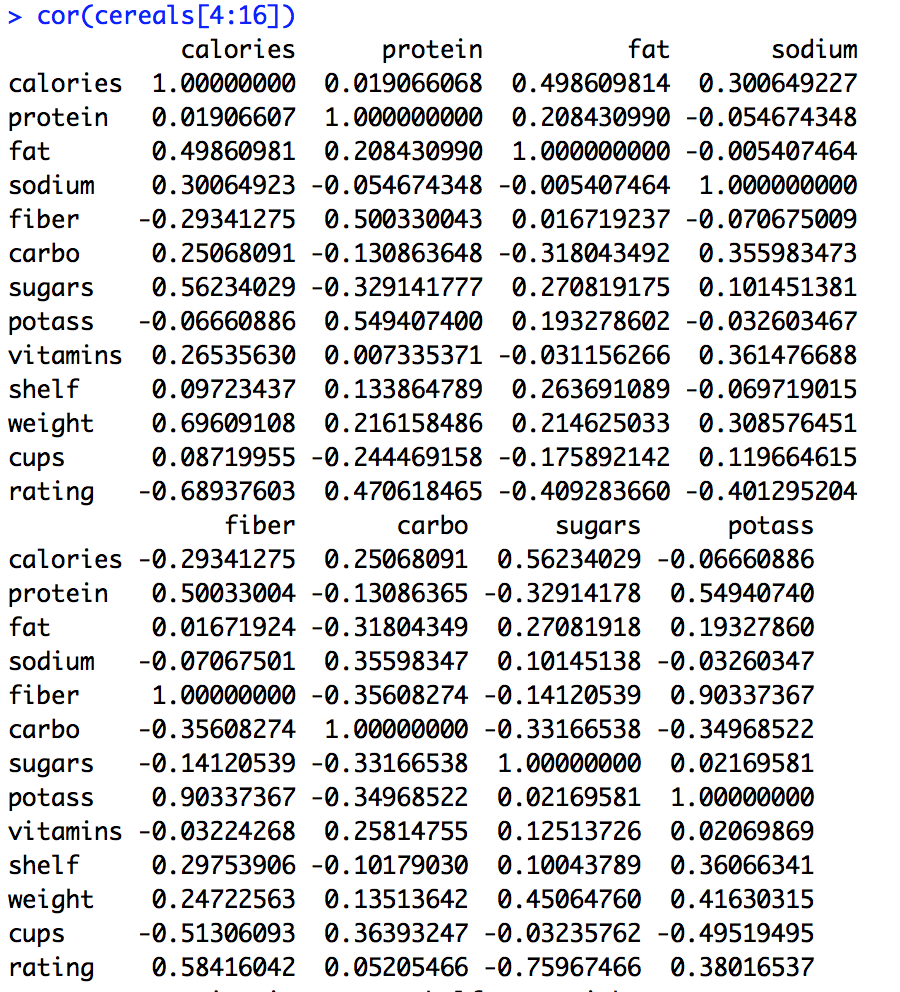
boxplot(cereals$rating~cereals$shelf)



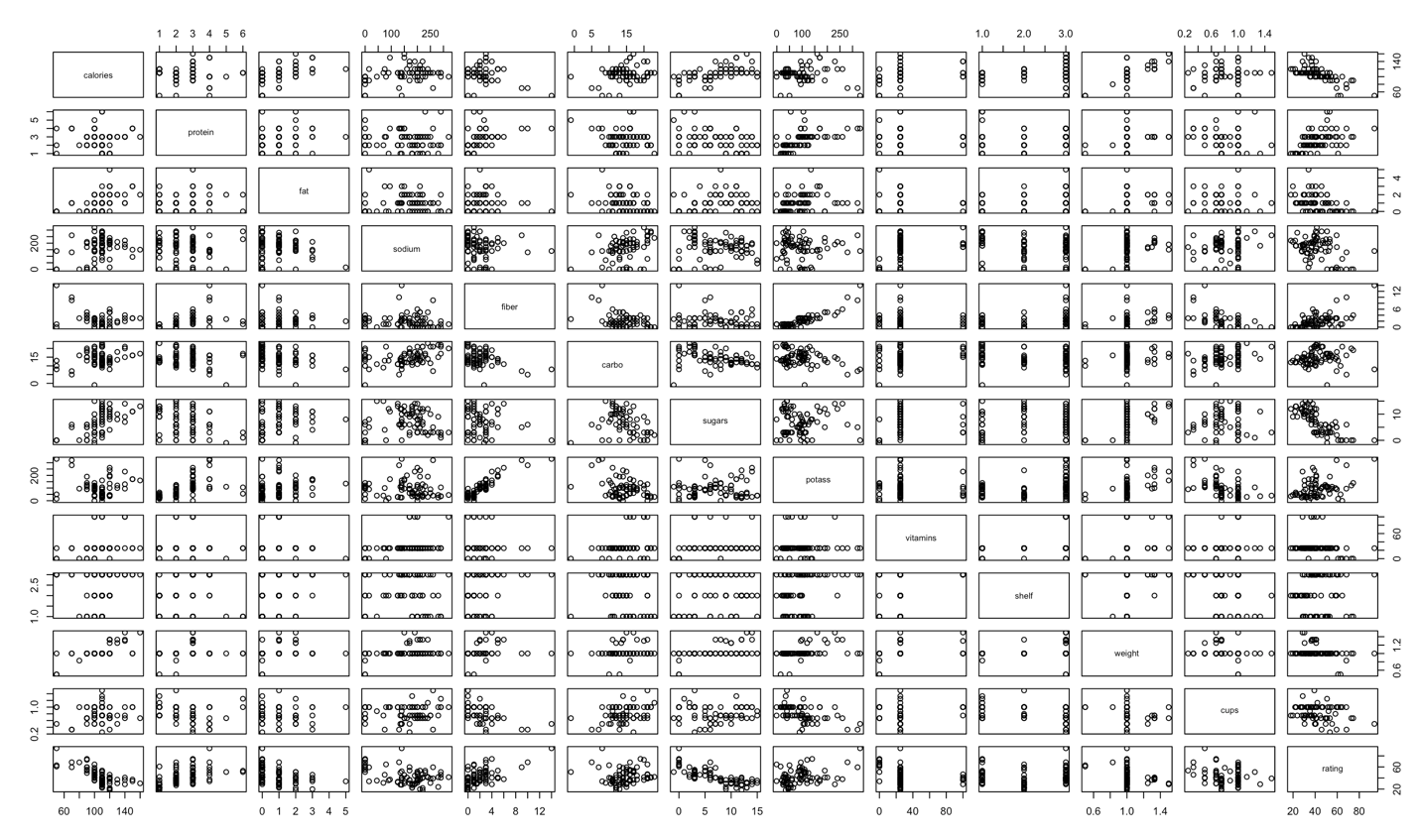
The mean of shelf\_1 is similar to shelf 3 so we can choose shelf\_1 and shelf\_2 or shelf\_1 and shelf\_3

**f.**

cor(cereals[4:16])

 cor

plot(cereals[4:16])



i

strongly correlated: Potassium and fiber are strong correlated.

ii

In the graph, there is a strong correlation between two variables. So we can dismiss one of them.

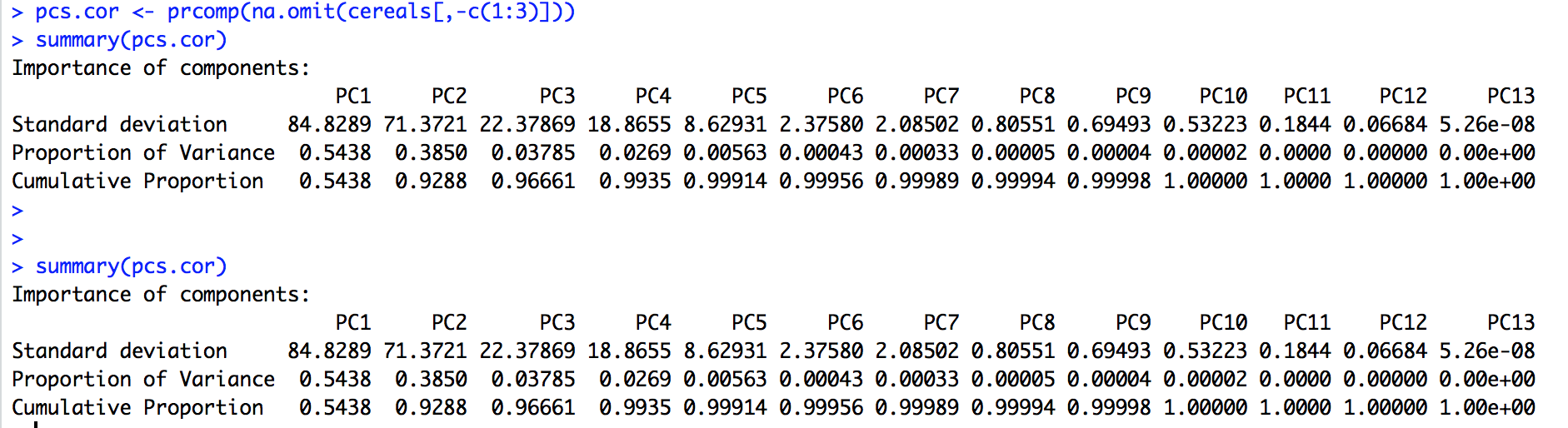
iii

After we normalize the data, we can conclude there is no change in the correlation matrix of the variables.

**g.**

pcs.cor <- prcomp(na.omit(cereals[,-c(1:3)]))

summary(pcs.cor)



The first principal component is the projection of the data sets onto the new basis vectors and the variance of the PC1 covers 54.38% of the whole information. We only need two 2 principal components to cover the 92.88%, which is more than 90%, of the total variability. Then we can reduce the variables from 13 to 2 to do the analysis.

**Problem3( House Median Price in Boston)**

bostonhousing<-read.csv("/Users/jingli/Desktop/BostonHousing.csv")

View(bostonhousing)

**a.**

based on the original data, we can find that categorical data is binary, so we use MIN MAX to rescale data from 0 to 1.

min <- sapply(bostonhousing,function(x) min(x))

max<-sapply(bostonhousing,function(x) max(x))

max\_min <- bostonhousing

for(i in 1:14){max\_min[,i]= (max\_min[,i]-min[i])/(max[i]-min[i])}

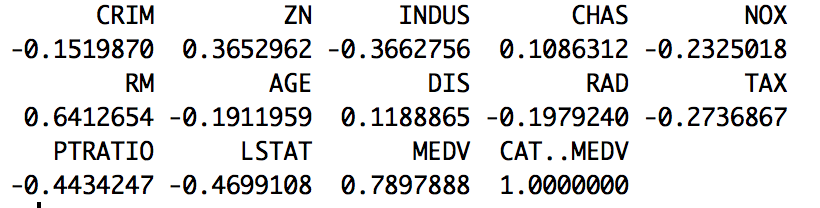
**b.**

Implementcorrelationanalysisforallfeaturescomparedtotheclass(MEDV>30)in the last column. Rank features based on their correlation coefficients and identify the top 5 features.

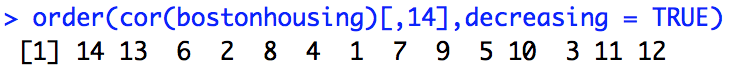
The top 5 features are

MEDV RM ZN DIS CHAS(top 5 features)

bostonhousing<-read.csv("/Users/jingli/Desktop/BostonHousing.csv")

cor(bostonhousing)[,14]

**order**(**cor**(bostonhousing)[,14],decreasing = TRUE)

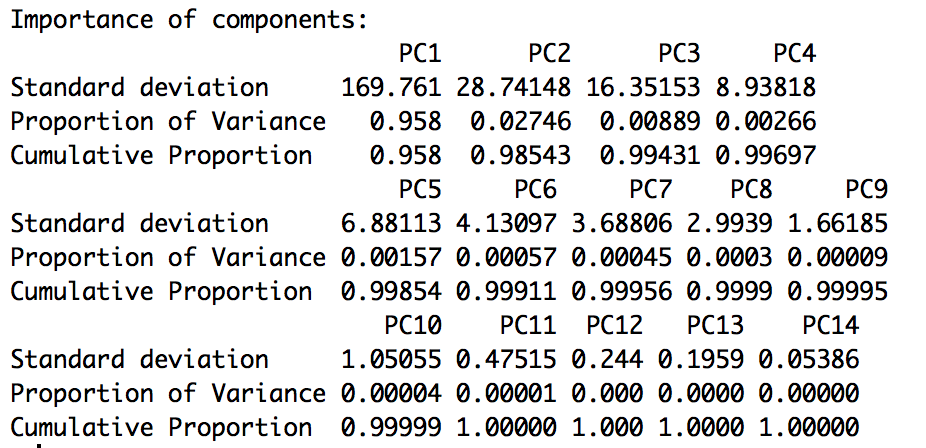


**c.**

bostonhousing<-read.csv("/Users/jingli/Desktop/BostonHousing.csv")

pca\_bh<-prcomp(bostonhousing)

summary(pca\_bh)

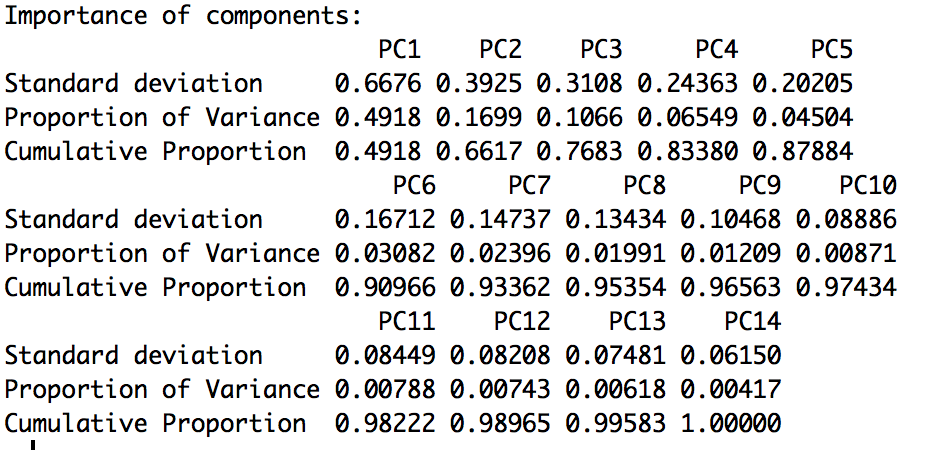


bostonhousing<-read.csv("/Users/jingli/Desktop/BostonHousing.csv")

pca\_bh<-prcomp(bostonhousing)

summary(pca\_bh)

pca\_bhs<-prcomp(max\_min)

summary(pca\_bhs)

intall.packages("devtools")

install.packages("ggplot2")

library(devtools)

install\_github("ggbiplot", "vqv")

## Warning: Username parameter is deprecated. Please use vqv/ggbiplot

## Skipping install of 'ggbiplot' from a github remote, the SHA1 (7325e880) has not changed since last install.

## Use `force = TRUE` to force installation

library(ggbiplot)

## Loading required package: ggplot2

##

## Attaching package: 'ggplot2'

## The following objects are masked from 'package:psych':

##

## %+%, alpha

## Loading required package: plyr

## Loading required package: scales

##

## Attaching package: 'scales'

## The following objects are masked from 'package:psych':

##

## alpha, rescale

## The following object is masked from 'package:plotrix':

##

## rescale

## Loading required package: grid

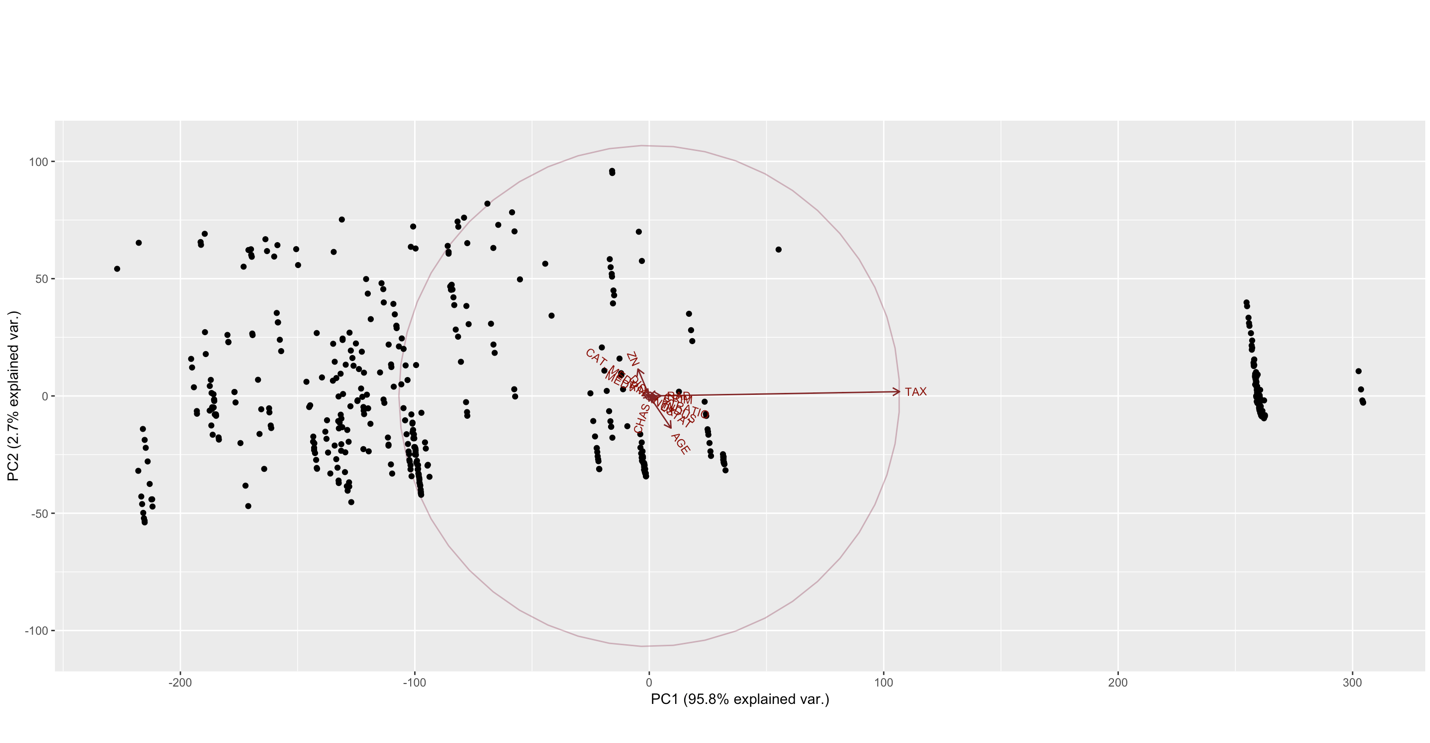
pca\_bh<-prcomp(bostonhousing)

g1 <- ggbiplot(pca\_bh, obs.scale = 1, var.scale = 1, ellipse = TRUE, circle = TRUE)

g1 <- g1 + scale\_color\_discrete(name = '')

g1 <- g1 + theme(legend.direction = 'horizontal', legend.position = 'top')

print(g1)



min <- sapply(bostonhousing,function(x) min(x))

max<-sapply(bostonhousing,function(x) max(x))

max\_min <- bostonhousing

for(i in 1:14){max\_min[,i]= (max\_min[,i]-min[i])/(max[i]-min[i])}

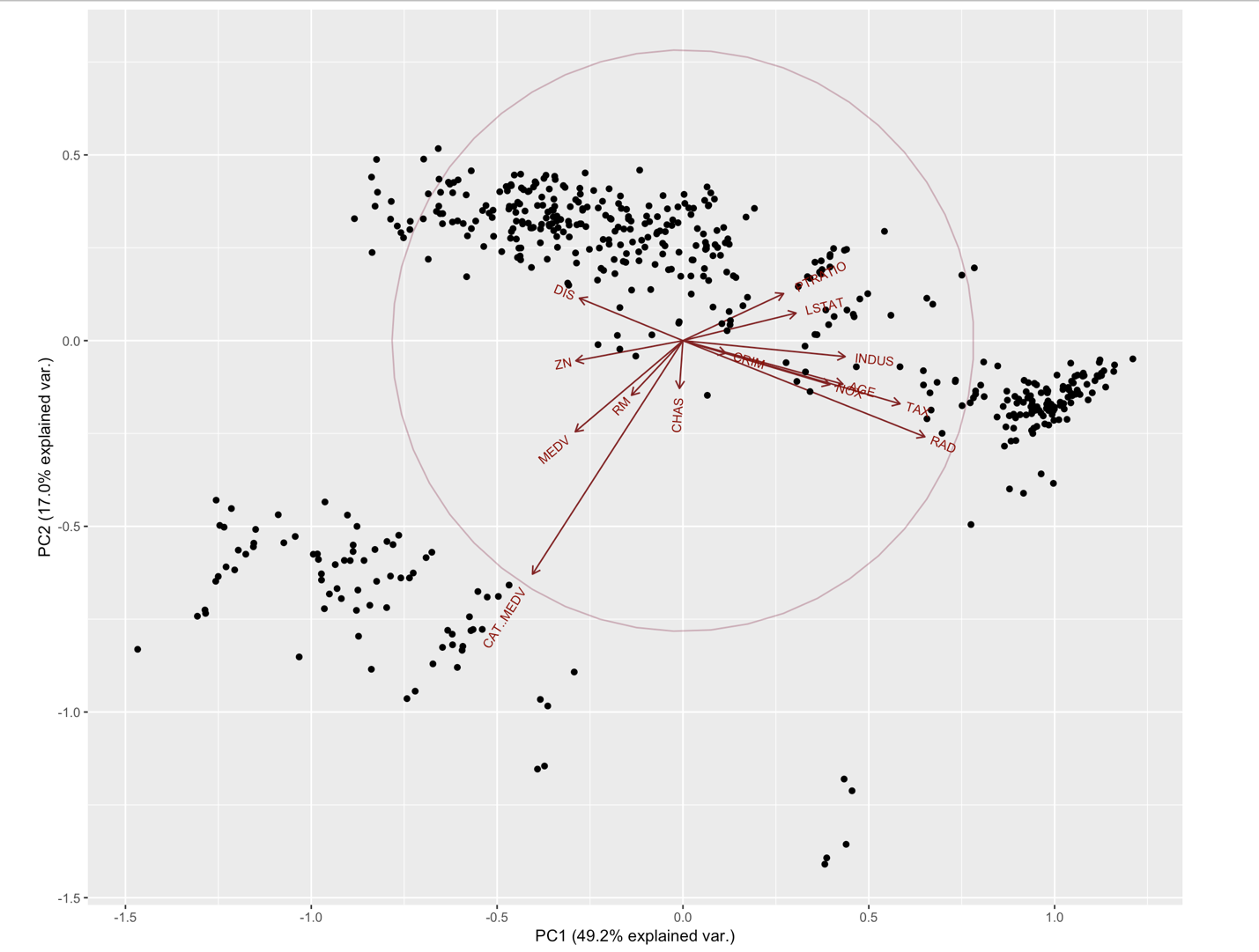
pca\_bhs<-prcomp(max\_min)

g2 <- ggbiplot(pca\_bhs, obs.scale = 1, var.scale = 1, ellipse = TRUE, circle = TRUE)

g2 <- g2 + scale\_color\_discrete(name = '')

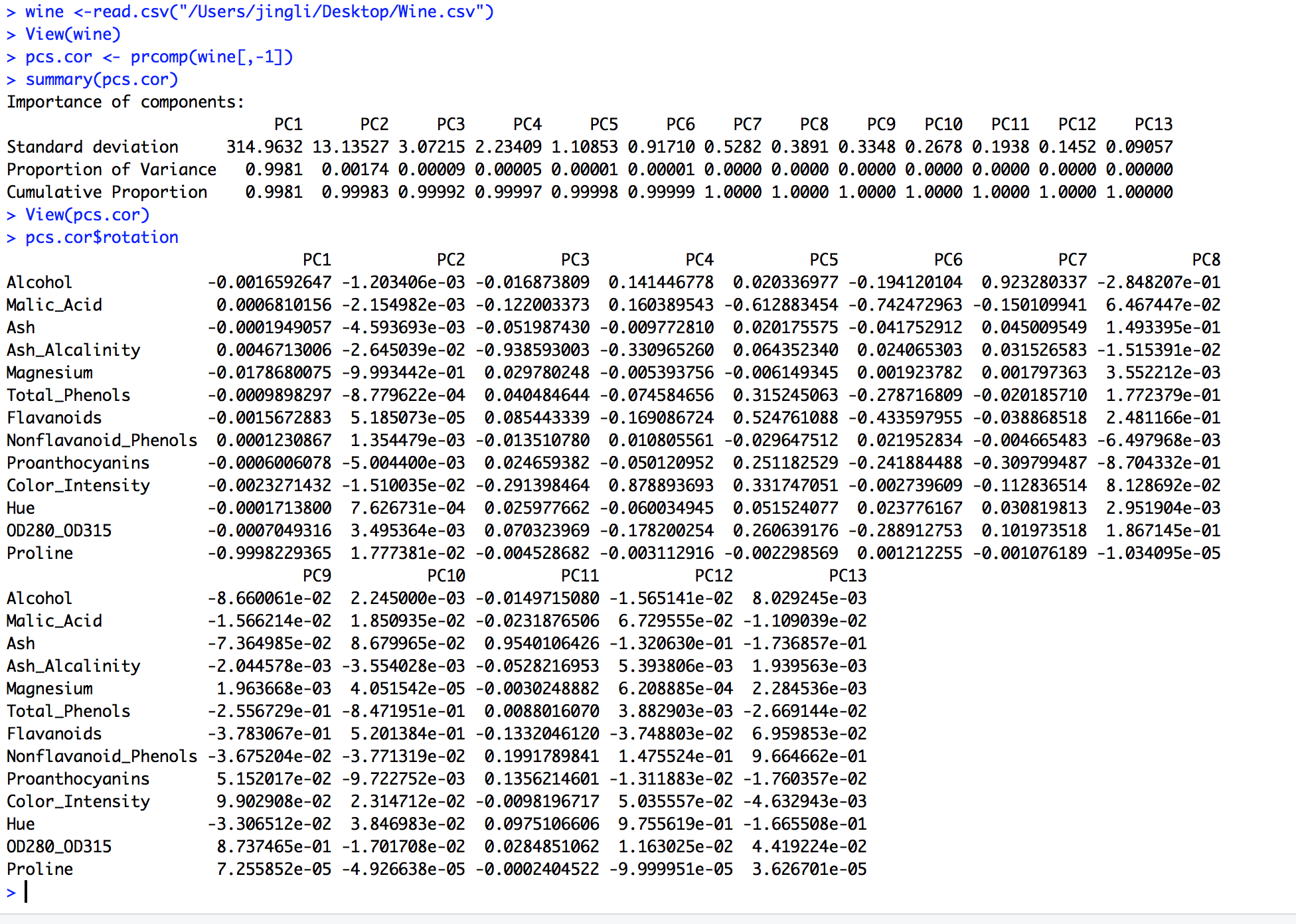
g2 <- g2 + theme(legend.direction = 'horizontal', legend.position = 'top')

print(g2)



**Problem4**

**a**. the first principal component is much bigger than others, all features are not at the same scale . Consequently, results will be influenced by original data, and it has the highest proportion of original variance



**b.**

After we normalized data, all features data are at the same scale, which means all variables have equal variability. so results are not influent by original data. we can find that we need 8 PCs to account for more than 90% of total variability. The Alternative to performing PCA is to perform PCA on correction matrix instead of the covariance matrix.