IE 7275 Data Mining in Engineering
Homework 1

Deadline 9/21

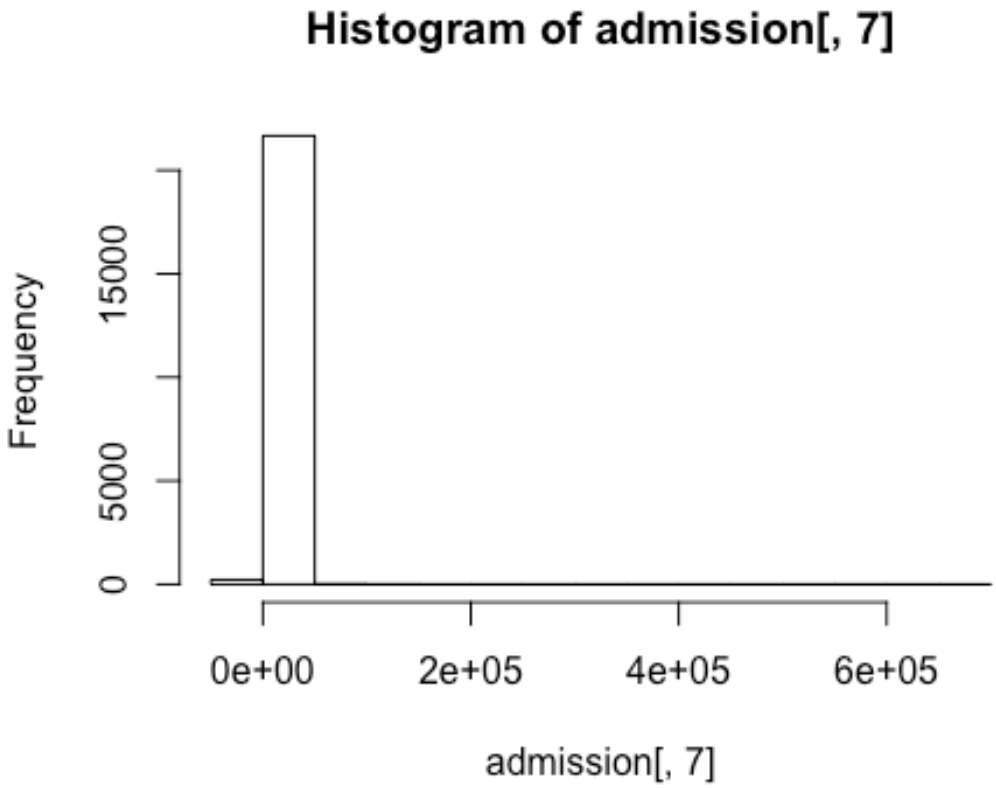
Jing Li

Problem1 (twitter account)

```
## load the data file
> hw1 <- read.csv("/Users/jingli/Desktop/M01_quasi_twitter.csv")
> View(hw1)
```

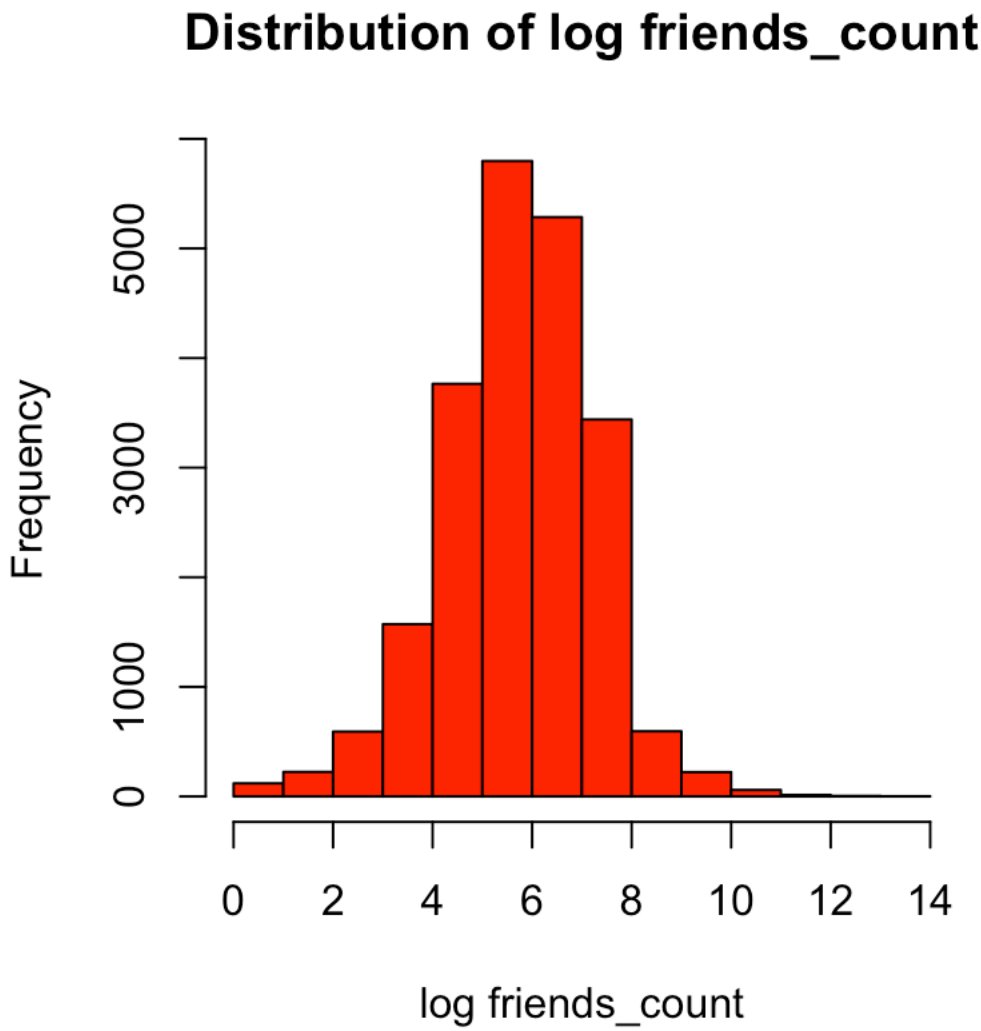
a. how are the data distributed for friends_count

```
## 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)

```
Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
 -84     123     324    1058     849  660549
```

b. 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)

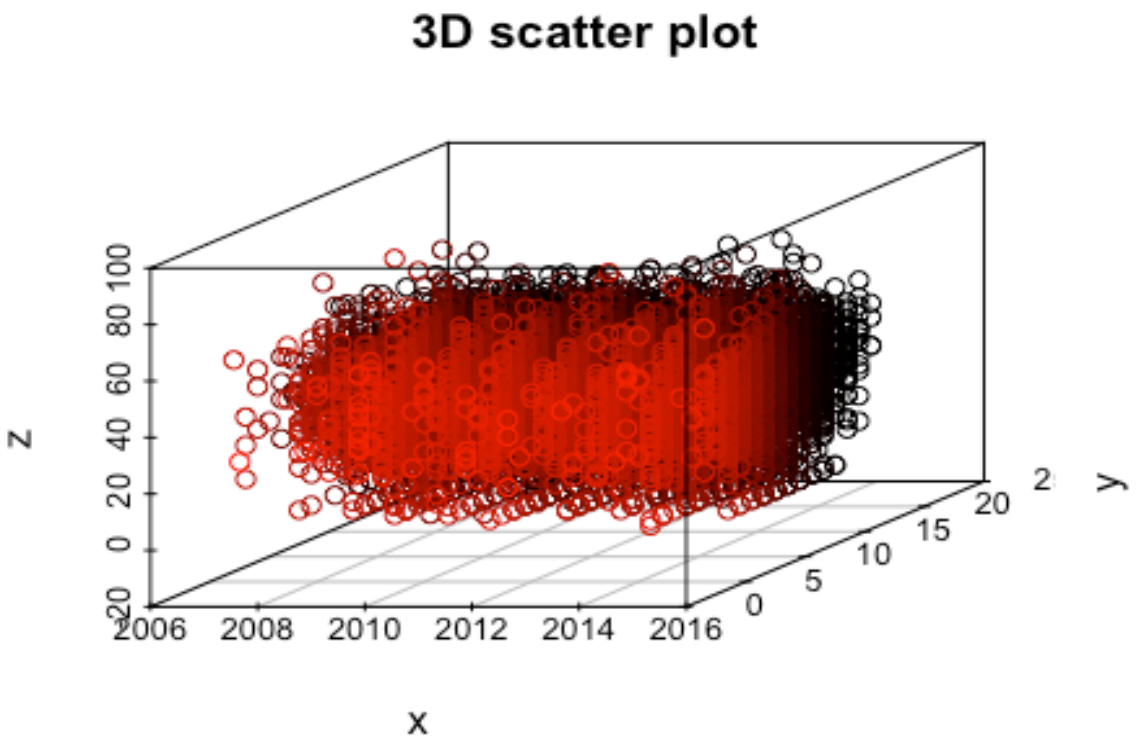
```
> ##evaluate the data quality of friends_count
> ##whether the data has NULL
> sum(is.na(hw1$friends_count))
[1] 0
>
> ##whether the data has minus valaues
> sum(hw1$friends_count < 0)
[1] 1
```

```
> sum(hw1$friends_count)
[1] 23185173
>
```

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.

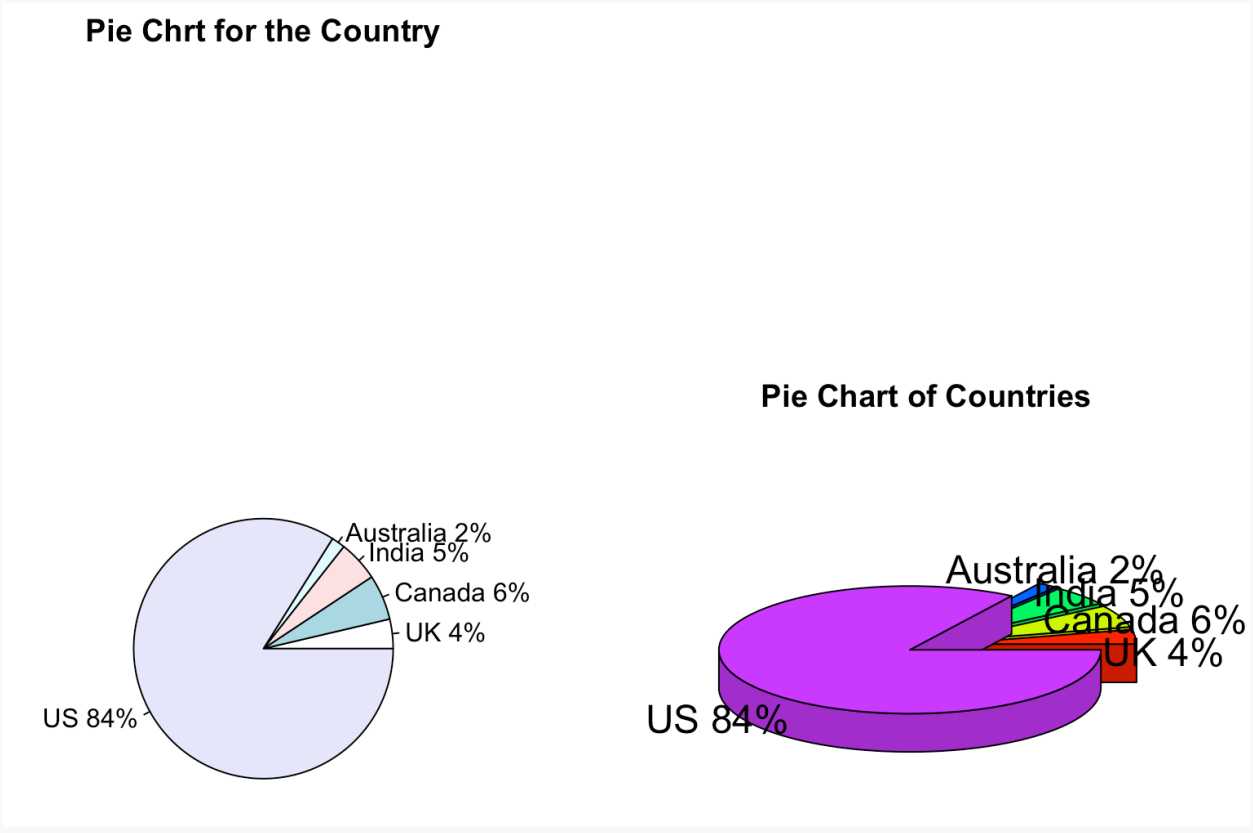
d . Produce a 3D scatter plot with highlighting

```
## 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")
```



e. Pie chart with percentage

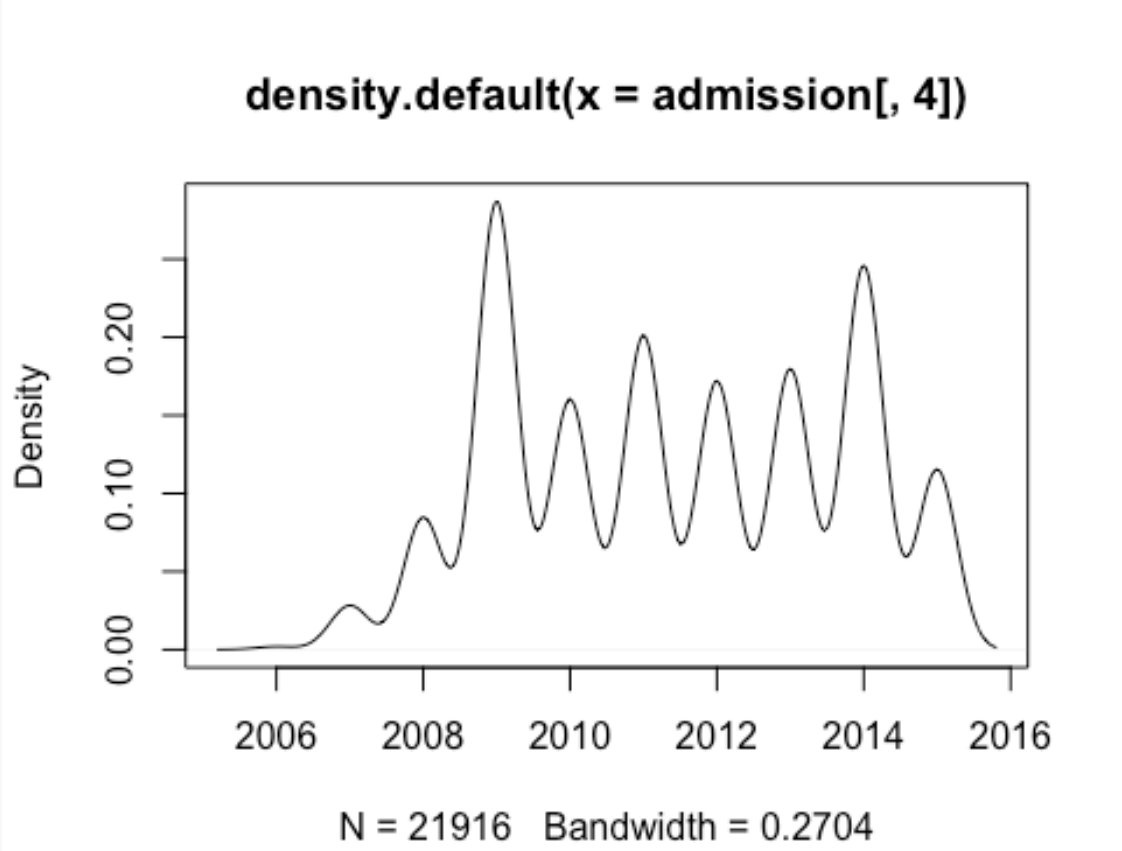
```
## e. pie chart with percentages
library(plotrix)
par(mfrow=c(1,2))
twitter_account <- c(650,1000,900,300,14900)
twitter_country <- c("UK","Canada","India","Australia","US")
percent <- round(twitter_account/sum(twitter_account)*100)
twitter_country <- paste(twitter_country, percent)# add percents 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")
```



f. 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)

```
    calories      protein          fat       sodium        fiber 
  106.883117    2.545455    1.012987  159.675325    2.151948 
       carbo       sugars       potass    vitamins        shelf 
   14.597403    6.922078   96.077922   28.246753    2.207792 
      weight         cups       rating 
    1.029610    0.821039   42.665705 
```

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

```
 calories   protein       fat    sodium     fiber     carbo 
110.00000   3.00000   1.00000 180.00000   2.00000  14.00000 
   sugars    potass  vitamins     shelf    weight      cups 
  7.00000  90.00000  25.00000   2.00000   1.00000   0.75000 
   rating 
 40.40021 
```

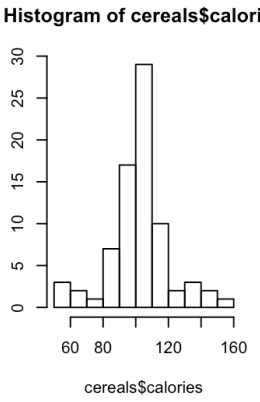
sapply(cereals[,4:16],max,na.rm=TRUE)

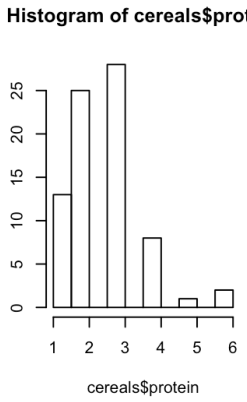
```
 calories   protein       fat    sodium     fiber     carbo 
160.00000   6.00000   5.00000 320.00000  14.00000  23.00000 
   sugars    potass  vitamins     shelf    weight      cups 
 15.00000 330.00000 100.00000   3.00000   1.50000   1.50000 
   rating 
 93.70491 
```

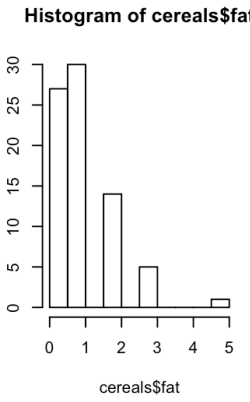
sapply(cereals[,4:16],sd,na.rm=TRUE)

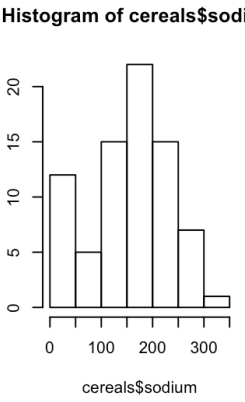
```
  calories    protein       fat     sodium      fiber 
19.4841191  1.0947897  1.0064726 83.8322952  2.3833640 
     carbo     sugars     potass   vitamins      shelf 
 4.2789563  4.4448854 71.2868125 22.3425225  0.8325241 
    weight       cups     rating 
 0.1504768  0.2327161 14.0472887 
```

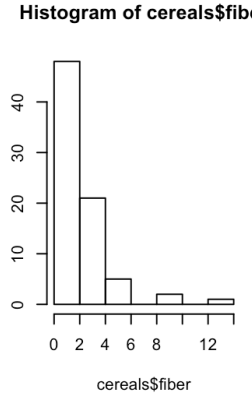
c. summary of quantities values except name, mfr shelf

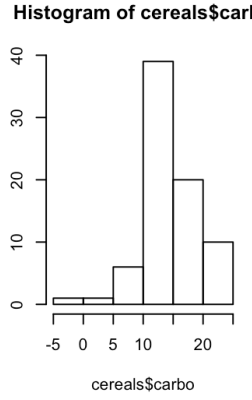


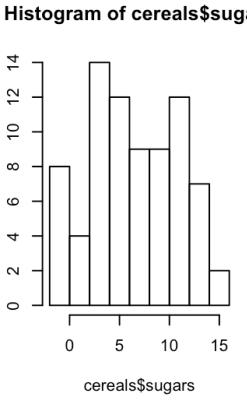


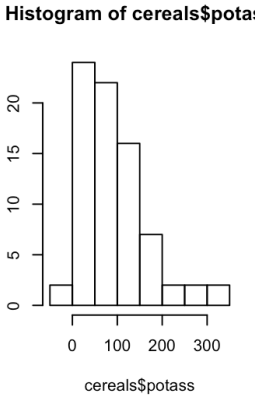


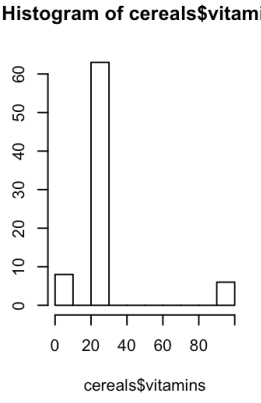


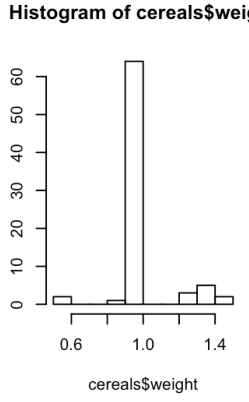


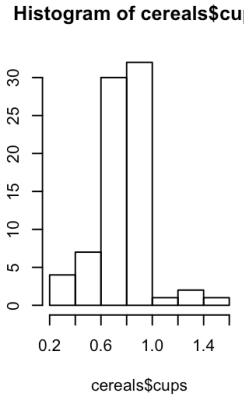


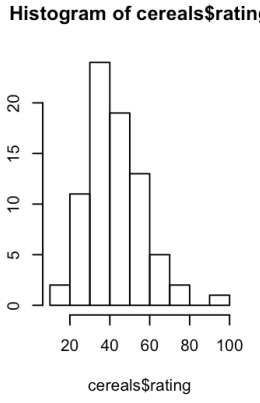












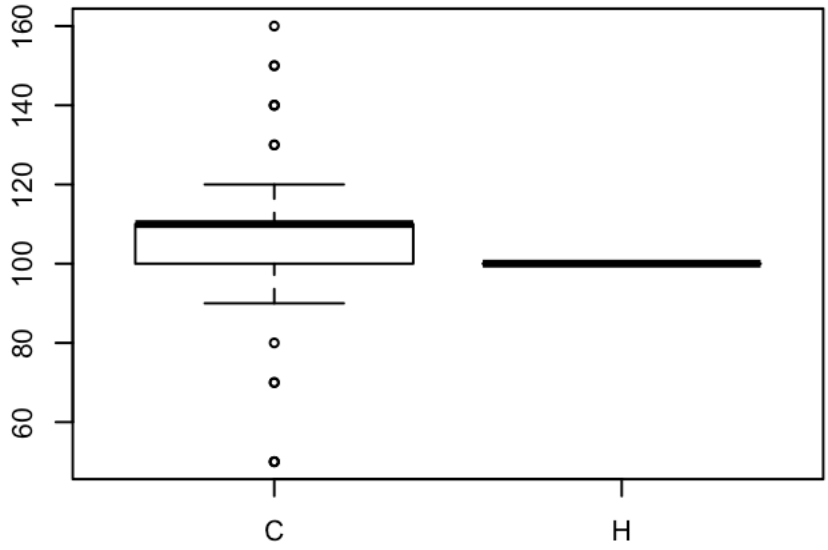
Values have largest variability?

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.

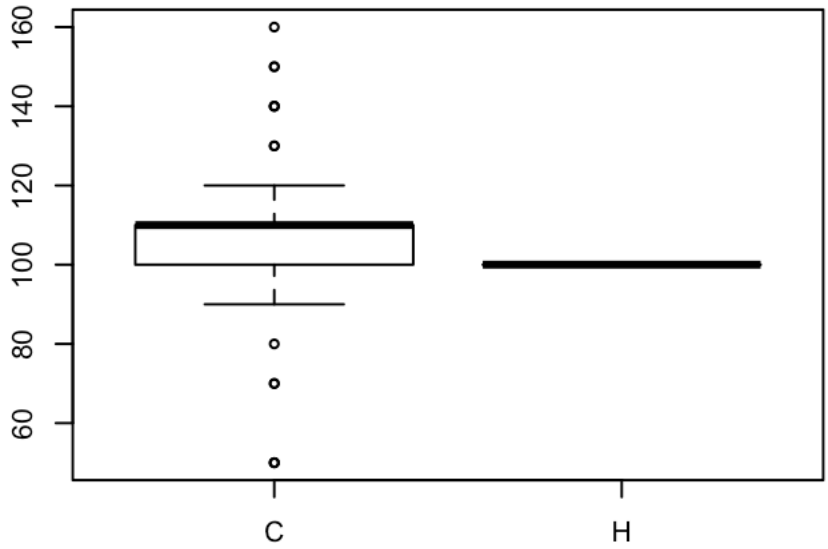
boxplot(cereals$calories~cereals$type)



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(cereals[4:16])
             calories      protein          fat       sodium
calories  1.00000000  0.019066068  0.498609814  0.300649227
protein   0.01906607  1.000000000  0.208430990 -0.054674348
fat       0.49860981  0.208430990  1.000000000 -0.005407464
sodium    0.30064923 -0.054674348 -0.005407464  1.000000000
fiber    -0.29341275  0.500330043  0.016719237 -0.070675009
carbo     0.25068091 -0.130863648 -0.318043492  0.355983473
sugars    0.56234029 -0.329141777  0.270819175  0.101451381
potass   -0.06660886  0.549407400  0.193278602 -0.032603467
vitamins  0.26535630  0.007335371 -0.031156266  0.361476688
shelf     0.09723437  0.133864789  0.263691089 -0.069719015
weight    0.69609108  0.216158486  0.214625033  0.308576451
cups      0.08719955 -0.244469158 -0.175892142  0.119664615
rating   -0.68937603  0.470618465 -0.409283660 -0.401295204
               fiber       carbo      sugars      potass
calories -0.29341275  0.25068091  0.56234029 -0.06660886
protein   0.50033004 -0.13086365 -0.32914178  0.54940740
fat       0.01671924 -0.31804349  0.27081918  0.19327860
sodium   -0.07067501  0.35598347  0.10145138 -0.03260347
fiber     1.00000000 -0.35608274 -0.14120539  0.90337367
carbo    -0.35608274  1.00000000 -0.33166538 -0.34968522
sugars   -0.14120539 -0.33166538  1.00000000  0.02169581
potass    0.90337367 -0.34968522  0.02169581  1.00000000
vitamins -0.03224268  0.25814755  0.12513726  0.02069869
shelf     0.29753906 -0.10179030  0.10043789  0.36066341
weight    0.24722563  0.13513642  0.45064760  0.41630315
cups     -0.51306093  0.36393247 -0.03235762 -0.49519495
rating    0.58416042  0.05205466 -0.75967466  0.38016537
```

```
              vitamins       shelf     weight        cups
calories  0.265356298  0.09723437  0.6960911  0.08719955
protein   0.007335371  0.13386479  0.2161585 -0.24446916
fat      -0.031156266  0.26369109  0.2146250 -0.17589214
sodium    0.361476688 -0.06971902  0.3085765  0.11966461
fiber    -0.032242679  0.29753906  0.2472256 -0.51306093
carbo     0.258147549 -0.10179030  0.1351364  0.36393247
sugars    0.125137260  0.10043789  0.4506476 -0.03235762
potass    0.020698687  0.36066341  0.4163032 -0.49519495
vitamins  1.000000000  0.29926167  0.3203241  0.12840454
shelf     0.299261665  1.00000000  0.1907620 -0.33526876
weight    0.320324059  0.19076197  1.0000000 -0.19958272
cups      0.128404543 -0.33526876 -0.1995827  1.00000000
rating   -0.240543611  0.02515882 -0.2981240 -0.20316006
               rating
calories -0.68937603
protein   0.47061846
fat      -0.40928366
sodium   -0.40129520
fiber     0.58416042
carbo     0.05205466
sugars   -0.75967466
potass    0.38016537
vitamins -0.24054361
shelf     0.02515882
weight   -0.29812398
cups     -0.20316006
rating    1.00000000
```

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)

```
> pcs.cor <- prcomp(na.omit(cereals[,-c(1:3)]))
> summary(pcs.cor)
Importance of components:
                          PC1     PC2      PC3     PC4     PC5     PC6     PC7     PC8     PC9    PC10   PC11    PC12     PC13
Standard deviation     84.8289 71.3721 22.37869 18.8655 8.62931 2.37580 2.08502 0.80551 0.69493 0.53223 0.1844 0.06684 5.26e-08
Proportion of Variance  0.5438  0.3850  0.03785  0.0269 0.00563 0.00043 0.00033 0.00005 0.00004 0.00002 0.0000 0.00000 0.00e+00
Cumulative Proportion   0.5438  0.9288  0.96661  0.9935 0.99914 0.99956 0.99989 0.99994 0.99998 1.00000 1.0000 1.00000 1.00e+00
>
>
> summary(pcs.cor)
Importance of components:
                          PC1     PC2      PC3     PC4     PC5     PC6     PC7     PC8     PC9    PC10   PC11    PC12     PC13
Standard deviation     84.8289 71.3721 22.37869 18.8655 8.62931 2.37580 2.08502 0.80551 0.69493 0.53223 0.1844 0.06684 5.26e-08
Proportion of Variance  0.5438  0.3850  0.03785  0.0269 0.00563 0.00043 0.00033 0.00005 0.00004 0.00002 0.0000 0.00000 0.00e+00
Cumulative Proportion   0.5438  0.9288  0.96661  0.9935 0.99914 0.99956 0.99989 0.99994 0.99998 1.00000 1.0000 1.00000 1.00e+00
```

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]

```
      CRIM         ZN      INDUS       CHAS        NOX
-0.1519870  0.3652962 -0.3662756  0.1086312 -0.2325018
        RM        AGE        DIS        RAD        TAX
 0.6412654 -0.1911959  0.1188865 -0.1979240 -0.2736867
   PTRATIO      LSTAT       MEDV  CAT..MEDV
-0.4434247 -0.4699108  0.7897888  1.0000000
```

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

```
> order(cor(bostonhousing)[,14],decreasing = TRUE)
 [1] 14 13  6  2  8  4  1  7  9  5 10  3 11 12
```

c.

```
bostonhousing<-read.csv("/Users/jingli/Desktop/BostonHousing.csv")
pca_bh<-prcomp(bostonhousing)
summary(pca_bh)
```

```
Importance of components:
                            PC1      PC2      PC3     PC4
Standard deviation     169.761 28.74148 16.35153 8.93818
Proportion of Variance   0.958  0.02746  0.00889 0.00266
Cumulative Proportion    0.958  0.98543  0.99431 0.99697
                            PC5     PC6     PC7    PC8     PC9
Standard deviation      6.88113 4.13097 3.68806 2.9939 1.66185
Proportion of Variance 0.00157 0.00057 0.00045 0.0003 0.00009
Cumulative Proportion  0.99854 0.99911 0.99956 0.9999 0.99995
                           PC10    PC11  PC12   PC13    PC14
Standard deviation      1.05055 0.47515 0.244 0.1959 0.05386
Proportion of Variance 0.00004 0.00001 0.000 0.0000 0.00000
Cumulative Proportion  0.99999 1.00000 1.000 1.0000 1.00000
```

```
bostonhousing<-read.csv("/Users/jingli/Desktop/BostonHousing.csv")
pca_bh<-prcomp(bostonhousing)
summary(pca_bh)
pca_bhs<-prcomp(max_min)
```

summary(pca_bhs)

```
Importance of components:
                          PC1    PC2    PC3     PC4     PC5
Standard deviation     0.6676 0.3925 0.3108 0.24363 0.20205
Proportion of Variance 0.4918 0.1699 0.1066 0.06549 0.04504
Cumulative Proportion  0.4918 0.6617 0.7683 0.83380 0.87884
                           PC6     PC7     PC8     PC9    PC10
Standard deviation     0.16712 0.14737 0.13434 0.10468 0.08886
Proportion of Variance 0.03082 0.02396 0.01991 0.01209 0.00871
Cumulative Proportion  0.90966 0.93362 0.95354 0.96563 0.97434
                          PC11    PC12    PC13    PC14
Standard deviation     0.08449 0.08208 0.07481 0.06150
Proportion of Variance 0.00788 0.00743 0.00618 0.00417
Cumulative Proportion  0.98222 0.98965 0.99583 1.00000
```

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':

##

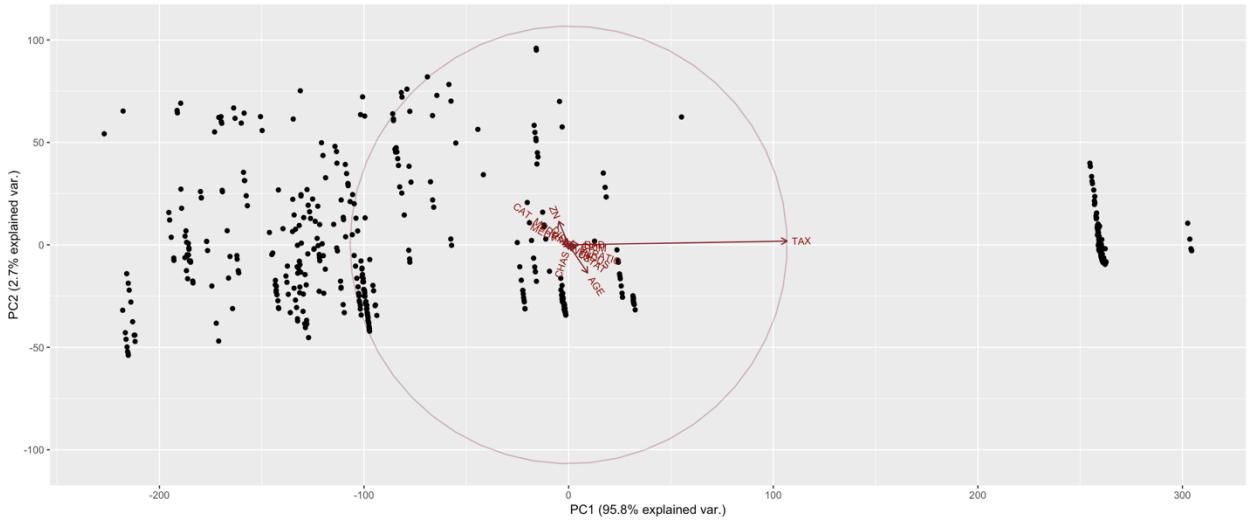
%+%, alpha

Loading required package: plyr

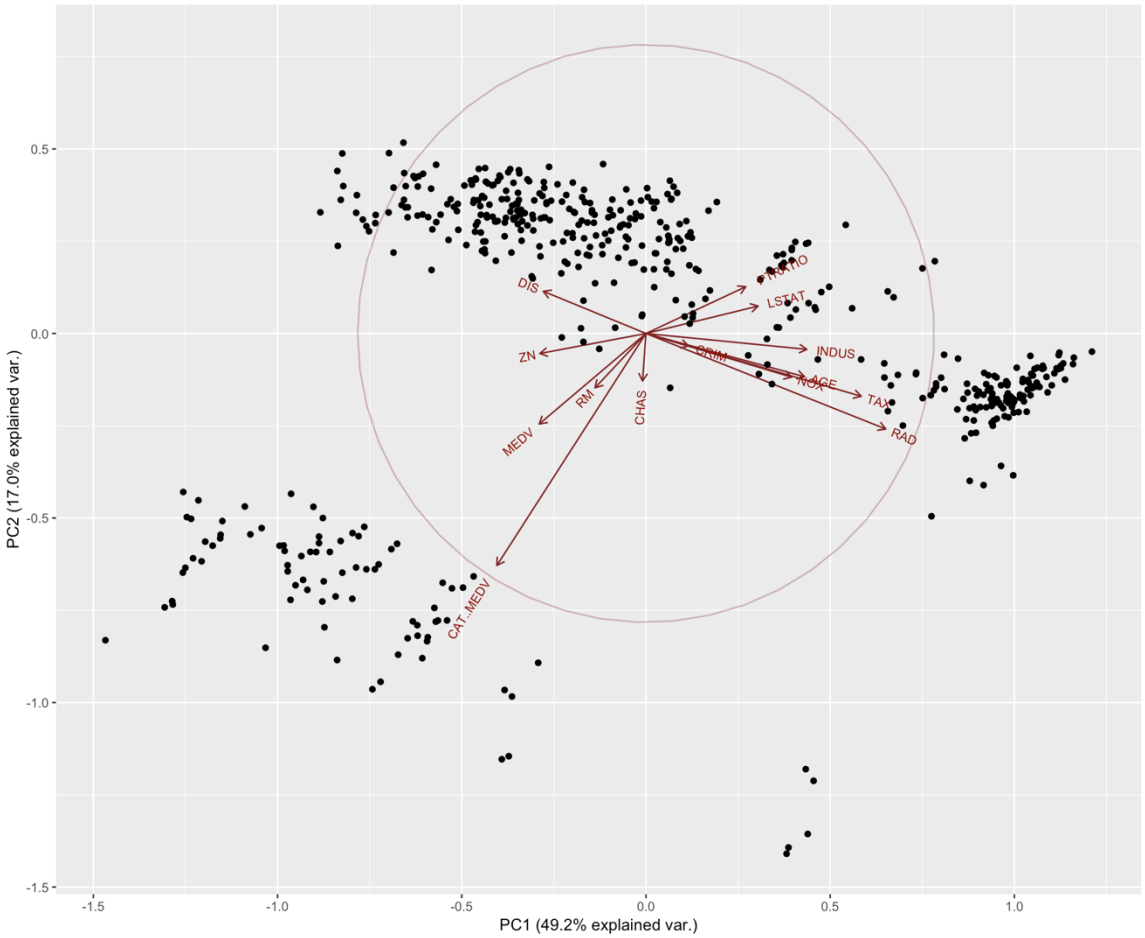
Loading required package: scales

##

```
## 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

```
> wine <-read.csv("/Users/jingli/Desktop/Wine.csv")
> View(wine)
> pcs.cor <- prcomp(wine[,-1])
> summary(pcs.cor)
Importance of components:
                          PC1      PC2     PC3     PC4     PC5     PC6    PC7    PC8    PC9   PC10   PC11   PC12    PC13
Standard deviation     314.9632 13.13527 3.07215 2.23409 1.10853 0.91710 0.5282 0.3891 0.3348 0.2678 0.1938 0.1452 0.09057
Proportion of Variance   0.9981  0.00174 0.00009 0.00005 0.00001 0.00001 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.00000
Cumulative Proportion    0.9981  0.99983 0.99992 0.99997 0.99998 0.99999 1.0000 1.0000 1.0000 1.0000 1.0000 1.0000 1.00000
> View(pcs.cor)
> pcs.cor$rotation
                                PC1           PC2           PC3           PC4           PC5           PC6           PC7           PC8
Alcohol              -0.0016592647 -1.203406e-03 -0.016873809  0.141446778  0.020336977 -0.194120104  0.923280337 -2.848207e-01
Malic_Acid            0.0006810156 -2.154982e-03 -0.122003373  0.160389543 -0.612883454 -0.742472963 -0.150109941  6.467447e-02
Ash                  -0.0001949057 -4.593693e-03 -0.051987430 -0.009772810  0.020175575 -0.041752912  0.045009549  1.493395e-01
Ash_Alcalinity        0.0046713006 -2.645039e-02 -0.938593003 -0.330965260  0.064352340  0.024065303  0.031526583 -1.515391e-02
Magnesium            -0.0178680075 -9.993442e-01  0.029780248 -0.005393756 -0.006149345  0.001923782  0.001797363  3.552212e-03
Total_Phenols        -0.0009898297 -8.779622e-04  0.040484644 -0.074584656  0.315245063 -0.278716809 -0.020185710  1.772379e-01
Flavanoids           -0.0015672883  5.185073e-05  0.085443339 -0.169086724  0.524761088 -0.433597955 -0.038868518  2.481166e-01
Nonflavanoid_Phenols  0.0001230867  1.354479e-03 -0.013510780  0.010805561 -0.029647512  0.021952834 -0.004665483 -6.497968e-03
Proanthocyanins      -0.0006006078 -5.004400e-03  0.024659382 -0.050120952  0.251182529 -0.241884488 -0.309799487 -8.704332e-01
Color_Intensity      -0.0023271432 -1.510035e-02 -0.291398464  0.878893693  0.331747051 -0.002739609 -0.112836514  8.128692e-02
Hue                  -0.0001713800  7.626731e-04  0.025977662 -0.060034945  0.051524077  0.023776167  0.030819813  2.951904e-03
OD280_OD315          -0.0007049316  3.495364e-03  0.070323969 -0.178200254  0.260639176 -0.288912753  0.101973518  1.867145e-01
Proline              -0.9998229365  1.777381e-02 -0.004528682 -0.003112916 -0.002298569  0.001212255 -0.001076189 -1.034095e-05
                              PC9          PC10          PC11          PC12          PC13
Alcohol              -8.660061e-02  2.245000e-03 -0.0149715080 -1.565141e-02  8.029245e-03
Malic_Acid           -1.566214e-02  1.850935e-02 -0.0231876506  6.729555e-02 -1.109039e-02
Ash                  -7.364985e-02  8.679965e-02  0.9540106426 -1.320630e-01 -1.736857e-01
Ash_Alcalinity       -2.044578e-03 -3.554028e-03 -0.0528216953  5.393806e-03  1.939563e-03
Magnesium             1.963668e-03  4.051542e-05 -0.0030248882  6.208885e-04  2.284536e-03
Total_Phenols        -2.556729e-01 -8.471951e-01  0.0088016070  3.882903e-03 -2.669144e-02
Flavanoids           -3.783067e-01  5.201384e-01 -0.1332046120 -3.748803e-02  6.959853e-02
Nonflavanoid_Phenols -3.675204e-02 -3.771319e-02  0.1991789841  1.475524e-01  9.664662e-01
Proanthocyanins       5.152017e-02 -9.722752e-03  0.1356214601 -1.311883e-02 -1.760357e-02
Color_Intensity       9.902908e-02  2.314712e-02 -0.0098196717  5.035557e-02 -4.632943e-03
Hue                  -3.306512e-02  3.846983e-02  0.0975106606  9.755619e-01 -1.665508e-01
OD280_OD315           8.737465e-01 -1.701708e-02  0.0284851062  1.163025e-02  4.419224e-02
Proline               7.255852e-05 -4.926638e-05 -0.0002404522 -9.999951e-05  3.626701e-05
>
```

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.