# final_datamining

```
#sum of error function
se=function(x){
  v=var(x)
  n=length(x)
  return(sqrt(v/n))
}
x=c(2,3,4,5,6,3,2,2)
se(x)
```

```
## [1] 0.5324304
```

```
#basic stat function
basic.stats=function(x,more=F){
  stats=list()
  clean.x=x[!is.na(x)]
  stats$n=length(x)
  stats$nNAs=stats$n-length(clean.x)
  stats$mean=mean(clean.x)
  stats$std=sd(clean.x)
  stats$med=median(clean.x)
  if (more) {
    stats$skew=sum(((clean.x-stats$mean)/stats$std)^3)/length(clean.x)
    stats$kurt=sum(((clean.x-stats$mean)/stats$std)^4)/length(clean.x)-3
  }
  stats
}

basic.stats(c(1,2,4,NA,32,55,-23),more=T)
```

```
## $n
## [1] 7
##
## $nNAs
## [1] 1
##
## $mean
## [1] 11.83333
##
## $std
## [1] 27.41836
##
## $med
## [1] 3
##
## $skew
## [1] 0.3530952
##
## $kurt
```

```
## [1] -1.485175
```

```
#for
f=function(x){
  for(i in 1:10){
    res=x*i
    cat(x,"*",i,'=',res,'\n')
  }
}

f(5)
```

```
## 5 * 1 = 5
## 5 * 2 = 10
## 5 * 3 = 15
## 5 * 4 = 20
## 5 * 5 = 25
## 5 * 6 = 30
## 5 * 7 = 35
## 5 * 8 = 40
## 5 * 9 = 45
## 5 * 10 = 50
```

```
algae <- read.table('https://home.ewha.ac.kr/~josong/dm/Analysis.txt',
             header=F,
             dec='.',
             col.names=c('season','size','speed','mxPH','mnO2','Cl','NO3',
             'NH4','oPO4','PO4','Chla','a1','a2','a3','a4','a5','a6','a7'),
             na.strings=c('XXXXXXX'),
             stringsAsFactors=T)
#algae fill NA
fillPO4=function(oP){
  if(is.na(oP)) return(NA)
  else return((oP+15.6142)/0.6446)
}

algae[is.na(algae$PO4),'PO4']=sapply(algae[is.na(algae$PO4),'oPO4'],fillPO4)

#fillPO4
```

```
#central value function
central.value=function(x){
  if(is.numeric(x)) median(x.na.rm=T)
  else if (is.factor(x)) level(x)[which.max(table(x))]
  else{
    f=as.factor(x)
    levels(f)[which.max(table(f))]
  }

}

#central.value(c(2,3,4))
```

```
#reliable.rpart/prune steps
reliable.rpart=function(form,data,se=1,cp=0,verbose=T,...){
  tree=rpart(form,data,cp=cp,...)
```

```
  if(verbose&ncol(tree$cptable)<5)
    warning("No pruning will be carried out because no estimates were obtained.")
  rt.prune(tree,se,verbose)

}
```

```
#
rt.prune=function(tree,se=1,verbose=T,...){
  if(ncol(tree$cptable)<5) tree
  else{
    lin.min.err=which.min(tree$cptable[,4])
    if(verbose&lin.min.err==nrow(tree$cptable))
      warning("Minimal Cross Balidation Error is obtained at the largest tree.\n Further tree growth (a
    tol.err=tree$cptable[lin.min.err,4]+se*tree$cptable[lin.min.err,5]
    se.lin=which(tree$cptable[,4]<=tol.err)[1]
    prune.rpart(tree,cp=tree$cptable[se.lin,1]+1e-9)
  }
}
```

```
#K-fold CValidation for 2 models

cross.validation=function(all.data,clean.data,n.folds=10){
  n=nrow(all.data)
  idx=sample(n,n)
  all.data=all.data[idx,]
  clean.data=clean.data[idx,]

  n.each.part=as.integer(n/n.folds)
  perf.lm=vector()
  perf.rt=vector()

  for (i in 1:n.folds){
    cat('Fold',i,'\n')
    out.fold=((i-1)*n.each.part+1):(i*n.each.part)

    l.model=lm(a1~.,clean.data[-out.fold,1:12])
    l.model=step(l.model)
    l.model.preds=predict(l.model,clean.data[out.fold,1:12])
    l.model.preds=ifelse(l.model.preds<0,0,l.model.preds)

    r.model=reliable.rpart(a1~.,all.data[-out.fold,1:12])
    r.model.preds=predict(r.model,all.data[out.fold,1:12])

    perf.lm[i]=mean((l.model.preds-all.data[out.fold,'a1'])^2)/mean((mean(all.data[-out.fold,'a1'])-all
    perf.tr[i]=mean((r.model.preds-all.data[out.fold,'a1'])^2)/mean((mean(all.data[-out.fold,'a1'])-all
  }
}

#list(lm=list(avg=mean(perf.lm),std=sd(perf.lm),fold.res=perf.lm),rt=list(avg=mean(perfect.rt),fold.res
```

```
#lm.all
lm.all=function(train,test){
  results=list()
  results$models=list()
```

```
  results$preds=list()
  for(alg in 1:7){
    results$models[[alg]]=step(lm(as.formula(paste(names(train)[11+alg],'~.')),data=train[,c(1:11,11+alg
    p=predict(results$models[[alg]],test)
    results$preds[[alg]]=ifelse(p<0,0,p)
  }
  results
}

#lm.models=lm.all(clean.algae,clean.test.algae)

#summary(lm.models$models[[5]])
```

## Cv 5fold

```
#
find_na_positions <- function(data) {
  # NA
  na_positions <- c()

  #
  if (is.vector(data)) {
    for (i in 1:length(data)) {
      if (is.na(data[i])) {
        na_positions <- c(na_positions, i)  # NA
      }
    }
  }

  #
  else if (is.data.frame(data)) {
    for (col in 1:ncol(data)) {
      for (row in 1:nrow(data)) {
        if (is.na(data[row, col])) {
          na_positions <- c(na_positions, paste("Row:", row, "Col:", col))
        }
      }
    }
  }

  # NA
  return(na_positions)
}

#   1:    NA
vector_example <- c(1, 2, NA, 4, NA, 6)
find_na_positions(vector_example)
```

```
## [1] 3 5
```

```
#   2:      NA
data_frame_example <- data.frame(
  A = c(1, 2, NA, 4),
```

```
  B = c(NA, 2, 3, 4),
  C = c(5, NA, 7, 8)
)
find_na_positions(data_frame_example)
```

```
## [1] "Row: 3 Col: 1" "Row: 1 Col: 2" "Row: 2 Col: 3"
```

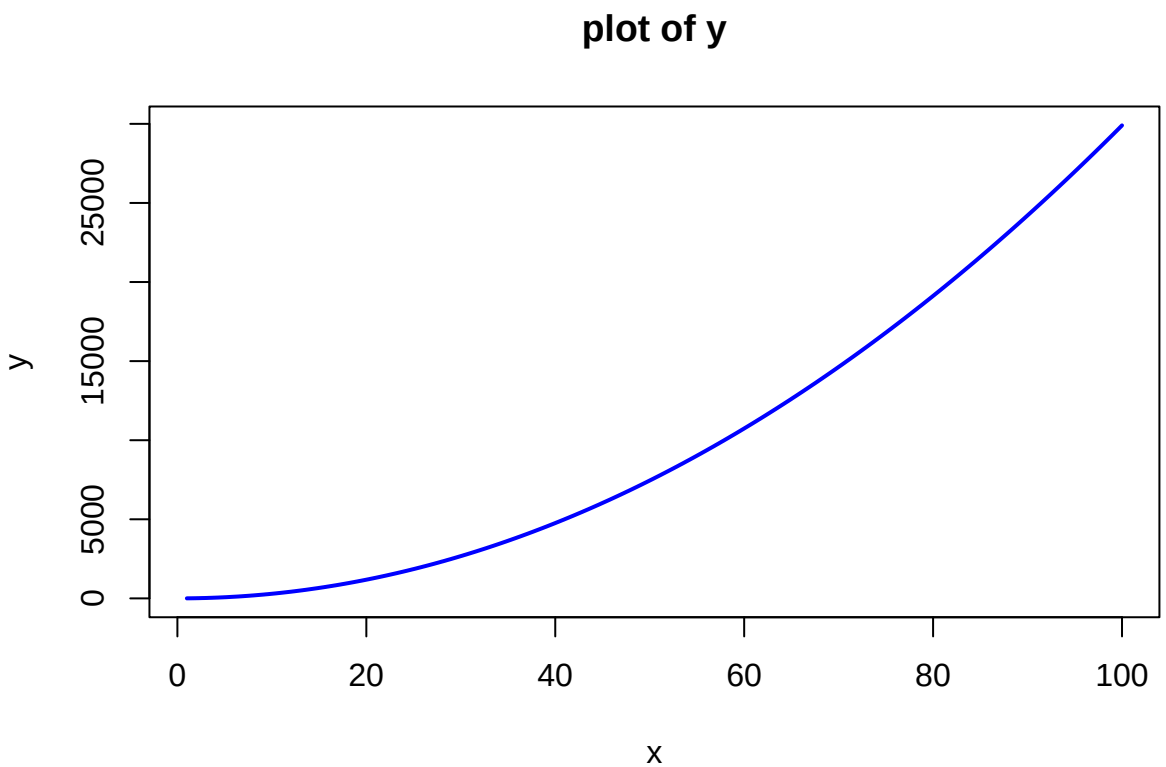
# 1

#2024 #6. (a) . my_max(x,y) , x y input return . ,
. (b) 1 100 0.1 y= 3x^2-x+2 plot .

```
#a
my_max=function(x,y){
  if(x>y){
    return(x)
  }else{
    return(y)
  }
}

my_max(3,4)
```

```
## [1] 4
```

```
#b
x=seq(1,100,by=0.1)
y=3*x^2-x+2
plot(x,y,type="l",col="blue",lwd=2,xlab="x",ylab="y",main="plot of y")
```



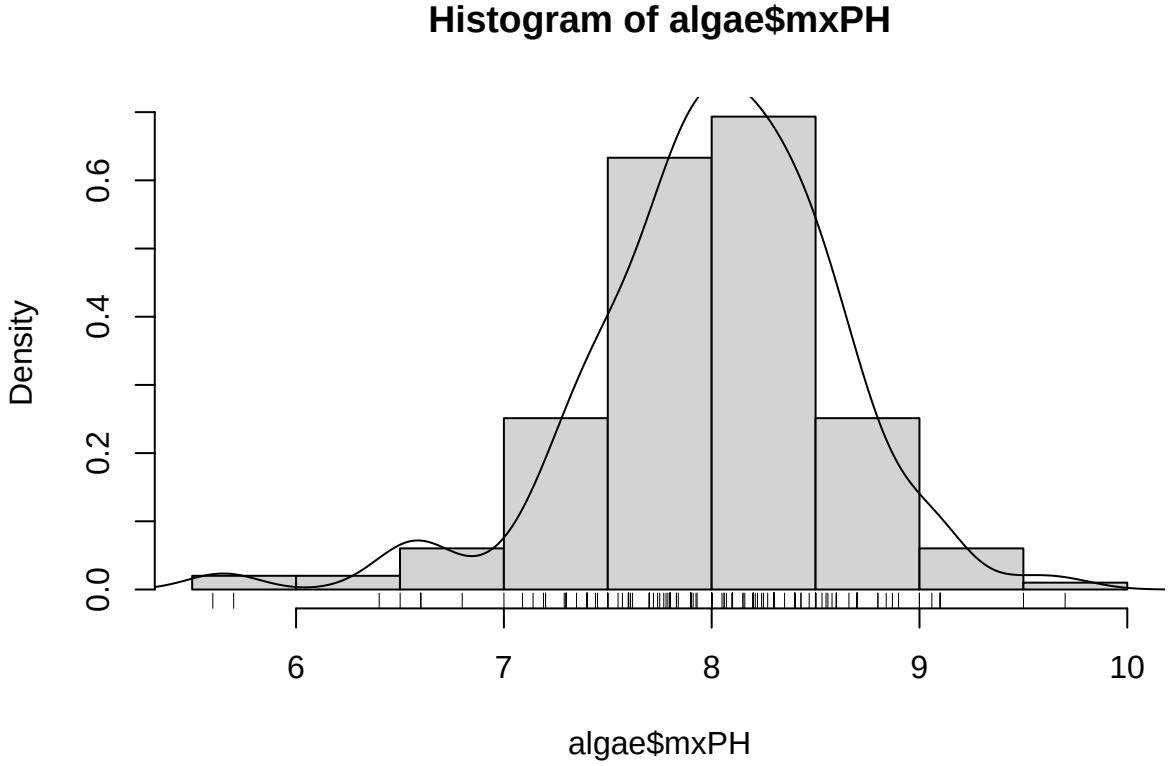

#7
:(a) list . 'myodd' . - for if (b) (a) mymodule py .

```
myodd=function(input_list){
  odd_numbers=c()
  for (i in input_list) {
    if(i %%2!=0){
      odd_numbers=c(odd_numbers,i)
    }

  }
  return(odd_numbers)
}

myodd(c(1,2,3,4,5,5,6))
```

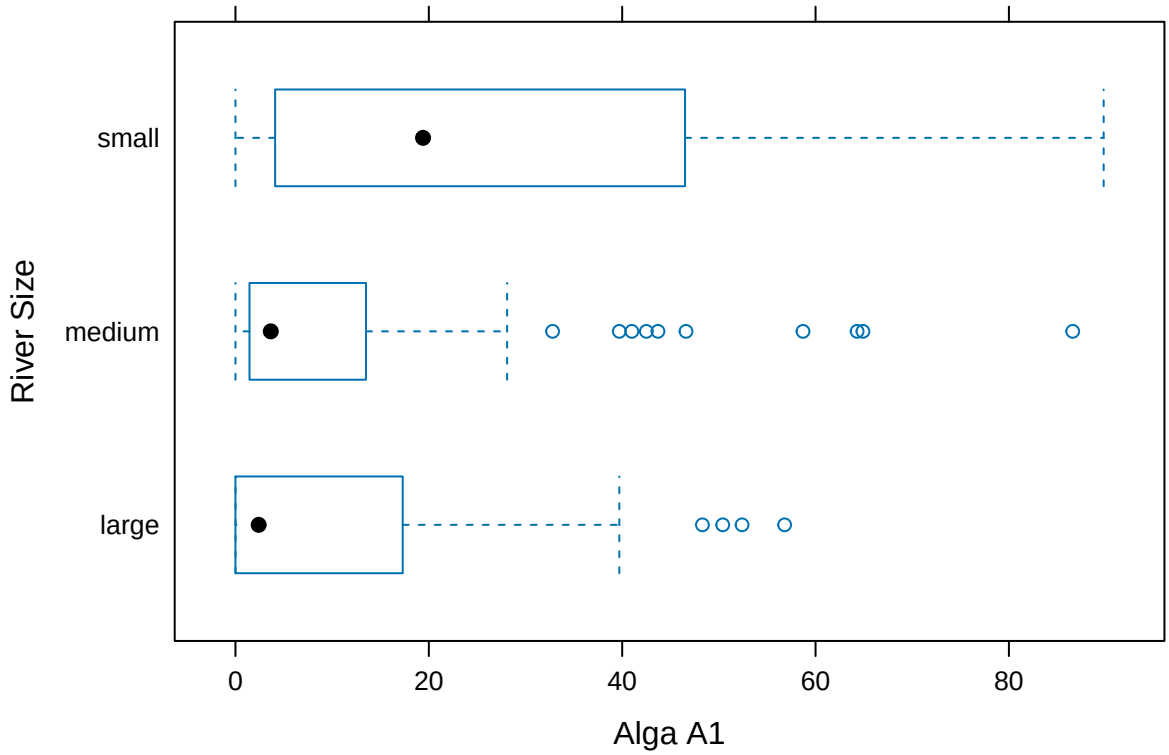
```
## [1] 1 3 5 5
```

#a fibonacci sequence a0=0,a1=1 . a(n+2)=an+a(n+1)

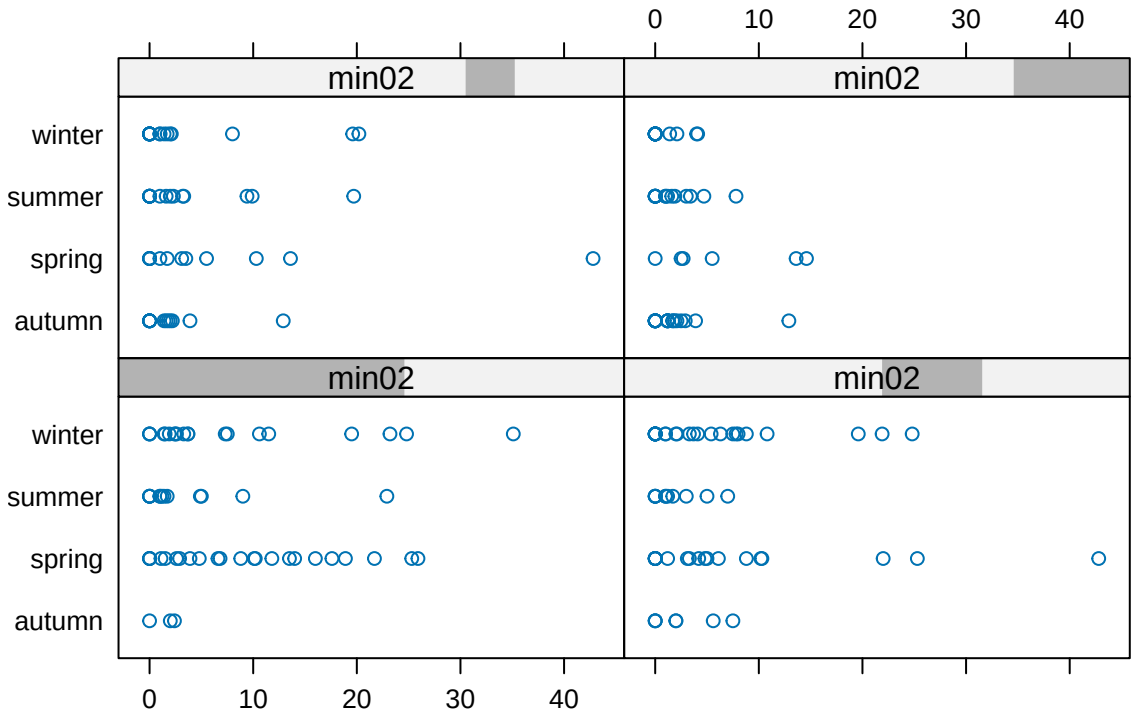
```
#
algae <- read.table('https://home.ewha.ac.kr/~josong/dm/Analysis.txt',
             header=F,
             dec='.',
             col.names=c('season','size','speed','mxPH','mnO2','Cl','NO3',
             'NH4','oPO4','PO4','Chla','a1','a2','a3','a4','a5','a6','a7'),
             na.strings=c('XXXXXXX'),
             stringsAsFactors=T)

hist(algae$mxPH, prob=T)

lines(density(algae$mxPH,na.rm=T))
rug(jitter(algae$mxPH))
```



```
library(lattice)



#
bwplot(size ~ a1, data = algae, ylab = 'River Size', xlab = 'Alga A1')
```




```
min02=equal.count(na.omit(algae$mn02),number=4,overlap=1/5)
stripplot(season ~ a3|min02,data=algae[!is.na(algae$mn02),])
```




a3

1. [30pts] .

(a) [2pts] ? , X=C(1,5,10, 4,3) 4 4 4 .

```
X=c(1,5,10, 4,3)
print(which.min(X))
```

```
## [1] 1
```

#(b) [4pts] . apply() (median) . #[,1] [,2] [,3] [,4] [,5] #[1,]112345 #[2,]1678910 #[3,] 11 12 13 14 15 #[4,] 16 17 18 19 20

```
mat=matrix(c(11,2,3,4,5,16,7,8,9,10,11,12,13,14,15,16,17,18,19,20),nrow=4,byrow=TRUE)
mat
```

```
##      [,1] [,2] [,3] [,4] [,5]
## [1,]   11    2    3    4    5
## [2,]   16    7    8    9   10
## [3,]   11   12   13   14   15
## [4,]   16   17   18   19   20
```

```
apply(mat,2,median)
```

```
## [1] 13.5  9.5 10.5 11.5 12.5
```

(c) [2pts] . sample() .

```
apply(mat,1,sample) #   1
```

```
##      [,1] [,2] [,3] [,4]
## [1,]    3    8   14   16
## [2,]    2    9   15   18
## [3,]    5   16   11   20
## [4,]    4    7   12   17
```

```
## [5,]   11   10   13   19
```

#(d) [3pts] 20 5x4 matrix . matrix 3 ( ) .

```
data=rnorm(20) #random variable

mat=matrix(data,nrow=5)
mat
```

```
##             [,1]       [,2]       [,3]        [,4]
## [1,]  1.73013829 0.5050827  0.7557696  0.21531047
## [2,]  0.09661928 0.6694740  1.8270947  0.81326674
## [3,] -1.88777028 1.1548427  0.5104353 -0.46688958
## [4,] -1.99458373 1.2454109 -1.8345908 -0.46448095
## [5,] -0.63311117 0.2539325 -1.4706928  0.04081132
```

```
mat[order(mat[,3]),]
```

```
##             [,1]       [,2]       [,3]        [,4]
## [1,] -1.99458373 1.2454109 -1.8345908 -0.46448095
## [2,] -0.63311117 0.2539325 -1.4706928  0.04081132
## [3,] -1.88777028 1.1548427  0.5104353 -0.46688958
## [4,]  1.73013829 0.5050827  0.7557696  0.21531047
## [5,]  0.09661928 0.6694740  1.8270947  0.81326674
```

#(e) [2pts] X t(2) (Student t-distribution).P(X 2) P(X q)= 0.9 q .

```
x=1-pt(2,df=2)
x
```

```
## [1] 0.09175171
```

```
qt(0.9,df=2) #quantile distn
```

```
## [1] 1.885618
```

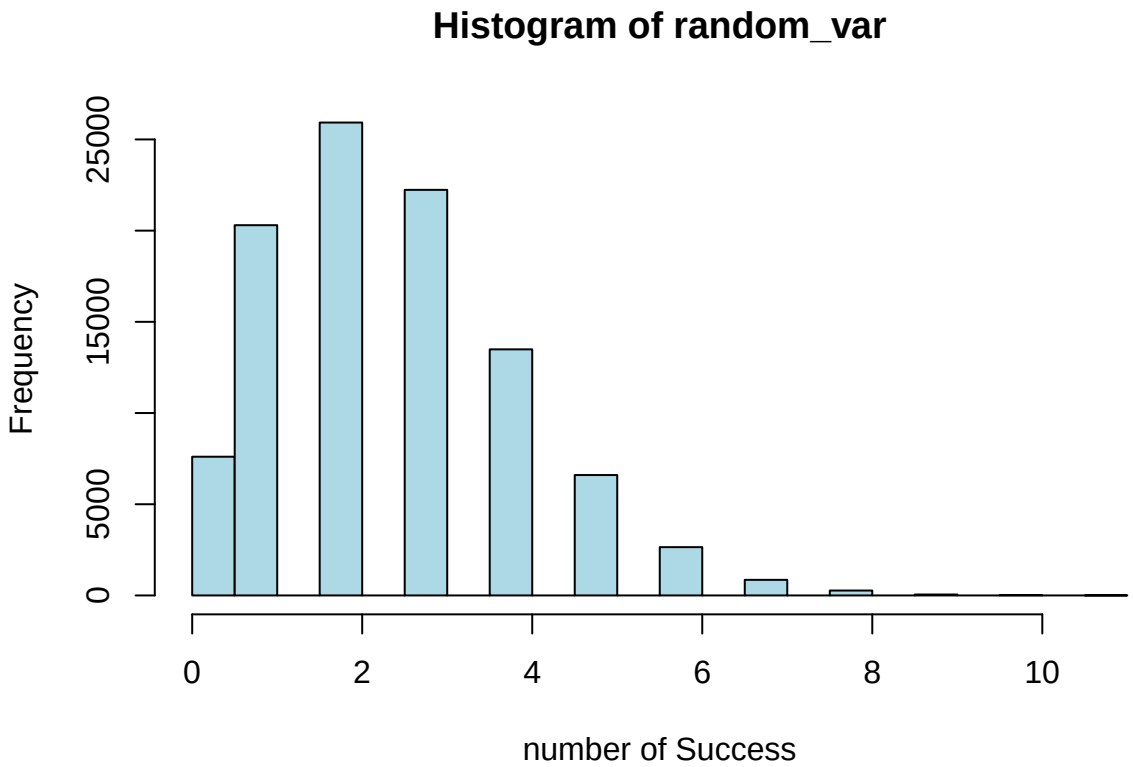
#(f) [7pts] company.csv . year, companyA, company.B 3 .CompanyA A , company.B B . .

```
library(ggplot2)
#
library(ggplot2)

# 1.       (            )
#data <- read.csv("company.csv")

#
#head(data)

# 2.     (Line plot)
#ggplot(data, aes(x = year)) +
#  geom_line(aes(y = companyA, color = "Company A")) +
#  geom_line(aes(y = companyB, color = "Company B")) +
 # labs(title = "Annual Revenue of Company A and Company B",
#       x = "Year",
 #      y = "Revenue") +
#  scale_color_manual(values = c("Company A" = "blue", "Company B" = "red")) +
 # theme_minimal()
```

```
# 3.    (Scatter plot)
#ggplot(data, aes(x = companyA, y = companyB)) +
#  geom_point(color = "darkgreen") +
 # labs(title = "Scatter Plot of Company A vs Company B Revenue",
  #     x = "Company A Revenue",
   #    y = "Company B Revenue") +
  #theme_minimal()

# 4.
#correlation <- cor(data$companyA, data$companyB)
#cat("   (Correlation between Company A and Company B):", correlation, "\n")
```

#(g) [10pts] X binomial (50,0.05) (n=50, p-0.05). #i. [3pts] 100,000 random variable histogram , histogram .

```
n=50
p=0.05
size=100000
random_var=rbinom(size,n,p)
hist(random_var,breaks=30,col="lightblue",xlab="number of Success",ylab="Frequency",border="black")
```

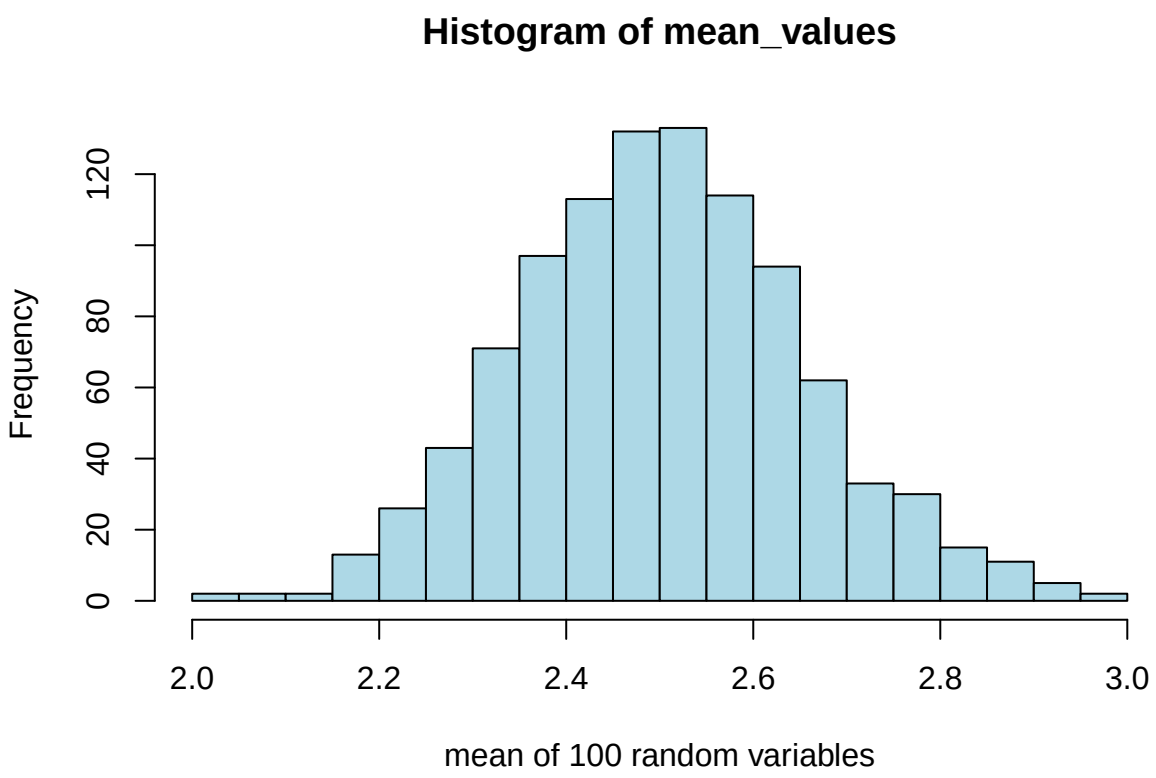



# , **(bell-shaped)** , . n p , .

#ii. [7pts] binomial(50,0.05) random variable 100 . 1000 1000 histogram . , histogram . , . ? ( : tor loop . For loop 100 B(50,0.05) , )

```
n=50;p=0.05;size=100;n_iterations=1000
mean_values=numeric(n_iterations)
for (i in 1:n_iterations){
  random_var=rbinom(size,n,p)
  mean_values[i]=mean(random_var)
```

```
}
hist(mean_values,breaks=30,col="lightblue",xlab="mean of 100 random variables",ylab="Frequency",border=
```




# :

, .

#2. [10pts] (midtermdata1). Course homepage R loading . linear regression .

#(a) [2pts] x1 x2 . . ? R-squared ?

```
#  , y  ;
```

#(b) [2pts] R-squared .

#(c) [2ptsJFit . ? R-squared ?

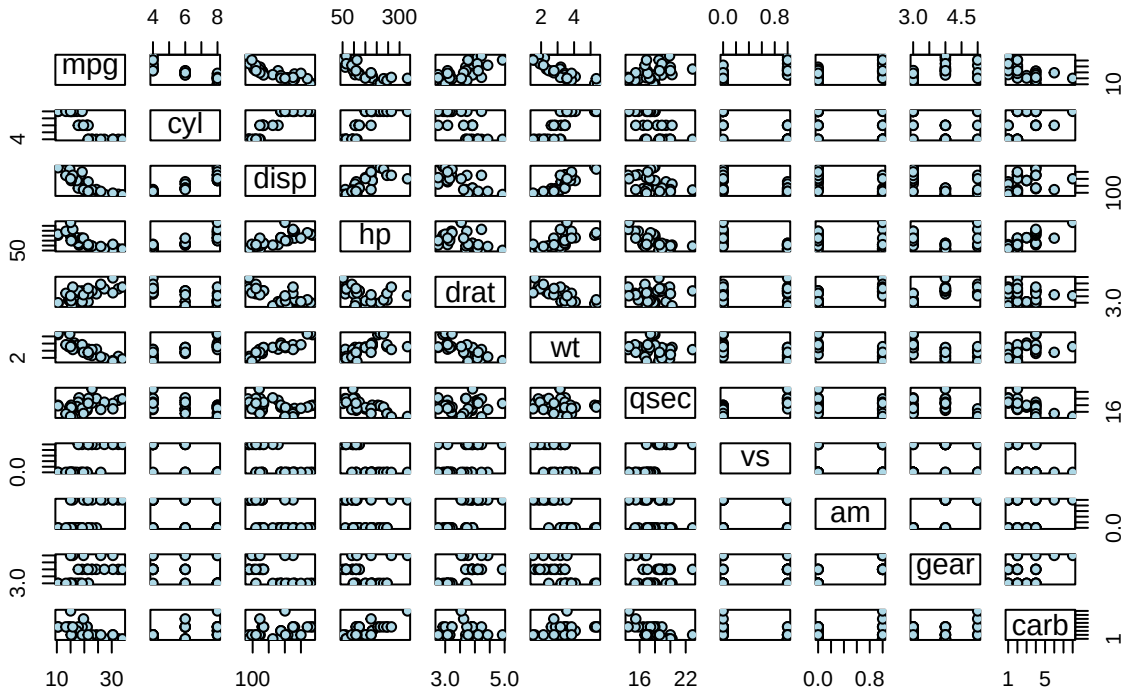
#(d) [4pts] (a) (c) ? anova test test . ,

#3. [50pts] . (mpg, miles per gallon) 9 32 . (mpg) .

```
data(mtcars)
```

#(a) [5pts] pariwise plot . ( ). patter ? correlatn ionol ? pattern ?

```r
pairs(mtcars,main="pairwise plot",pch=21,bg="lightblue")
```

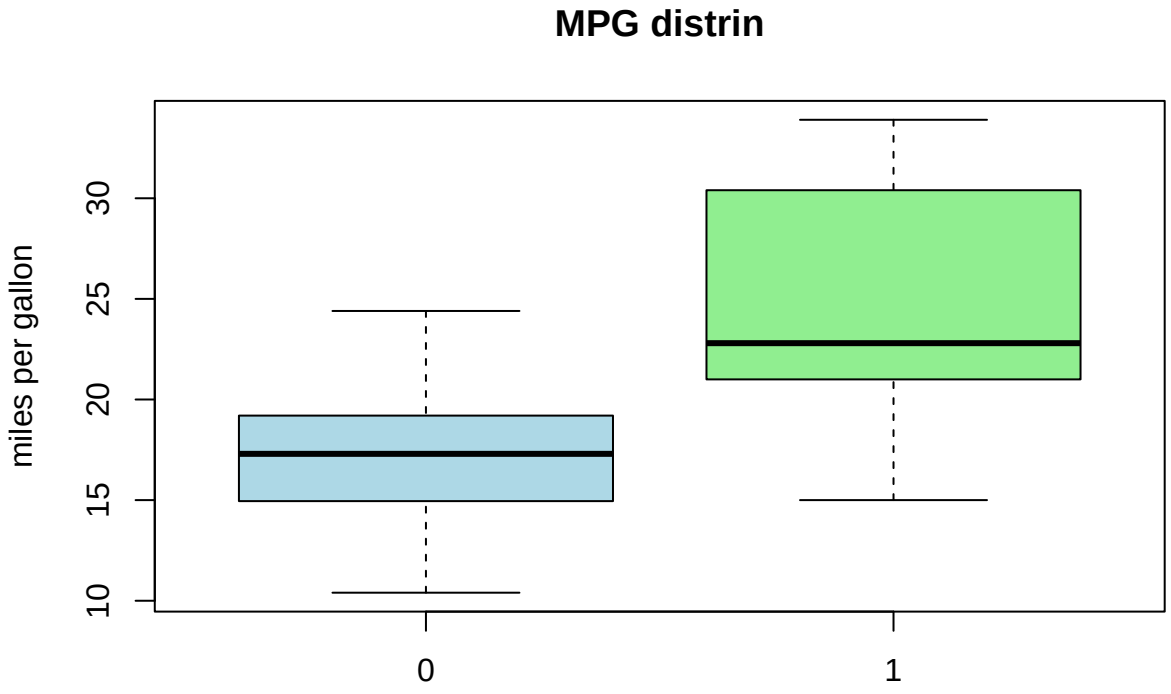



```r
cor_matrix=cor(mtcars)
cor_matrix
```

```
##             mpg        cyl       disp         hp        drat         wt
## mpg   1.0000000 -0.8521620 -0.8475514 -0.7761684  0.68117191 -0.8676594
## cyl  -0.8521620  1.0000000  0.9020329  0.8324475 -0.69993811  0.7824958
## disp -0.8475514  0.9020329  1.0000000  0.7909486 -0.71021393  0.8879799
## hp   -0.7761684  0.8324475  0.7909486  1.0000000 -0.44875912  0.6587479
## drat  0.6811719 -0.6999381 -0.7102139 -0.4487591  1.00000000 -0.7124406
## wt   -0.8676594  0.7824958  0.8879799  0.6587479 -0.71244065  1.0000000
## qsec  0.4186840 -0.5912421 -0.4336979 -0.7082234  0.09120476 -0.1747159
## vs    0.6640389 -0.8108118 -0.7104159 -0.7230967  0.44027846 -0.5549157
## am    0.5998324 -0.5226070 -0.5912270 -0.2432043  0.71271113 -0.6924953
## gear  0.4802848 -0.4926866 -0.5555692 -0.1257043  0.69961013 -0.5832870
## carb -0.5509251  0.5269883  0.3949769  0.7498125 -0.09078980  0.4276059
##             qsec         vs          am       gear        carb
## mpg   0.41868403  0.6640389  0.59983243  0.4802848 -0.55092507
## cyl  -0.59124207 -0.8108118 -0.52260705 -0.4926866  0.52698829
## disp -0.43369788 -0.7104159 -0.59122704 -0.5555692  0.39497686
## hp   -0.70822339 -0.7230967 -0.24320426 -0.1257043  0.74981247
## drat  0.09120476  0.4402785  0.71271113  0.6996101 -0.09078980
## wt   -0.17471588 -0.5549157 -0.69249526 -0.5832870  0.42760594
## qsec  1.00000000  0.7445354 -0.22986086 -0.2126822 -0.65624923
## vs    0.74453544  1.0000000  0.16834512  0.2060233 -0.56960714
## am   -0.22986086  0.1683451  1.00000000  0.7940588  0.05753435
## gear -0.21268223  0.2060233  0.79405876  1.0000000  0.27407284
## carb -0.65624923 -0.5696071  0.05753435  0.2740728  1.00000000
```

#(b) [5pts] am mpg . mpg vs am plot . ( R ) mpg am=0 am=1 ?

```
boxplot(mpg~am,data=mtcars,main="MPG distrin",xlab="transmission type(0=Automatic,1=Manual)",ylab="miles
```




# : #am = 0 ( ): mpg 17.1 , IQR ( ) 15 ~ 20 . . #am = 1 ( ): mpg 24.4 , IQR 21 ~ 28 . . # : # (am = 1) (am = 0)** . . # IQR : . . : am = 1 . . mpg , .

#(c) [10pts] missing value 1 . , Correlation-regression . , , . ( : 4, 6, 8 . .) . am factor .

```
#      (ggplot2     )
library(ggplot2)

# mpg
data(mpg)

#
missing_index <- which(is.na(mpg))
cat("          :\n")
```

```
##          :
```

```
print(missing_index)
```

```
## integer(0)
```

```
#           .
mpg[missing_index, ]
```

```
## # A tibble: 0 x 11
## # i 11 variables: manufacturer <chr>, model <chr>, displ <dbl>, year <int>,
```

```
## #   cyl <int>, trans <chr>, drv <chr>, cty <int>, hwy <int>, fl <chr>,
## #   class <chr>
```

```
#
summary(mpg)
```

```
##  manufacturer          model               displ            year
##  Length:234         Length:234         Min.   :1.600   Min.   :1999
##  Class :character   Class :character   1st Qu.:2.400   1st Qu.:1999
##  Mode  :character   Mode  :character   Median :3.300   Median :2004
##                                        Mean   :3.472   Mean   :2004
##                                        3rd Qu.:4.600   3rd Qu.:2008
##                                        Max.   :7.000   Max.   :2008
##       cyl           trans               drv                 cty
##  Min.   :4.000   Length:234         Length:234         Min.   : 9.00
##  1st Qu.:4.000   Class :character   Class :character   1st Qu.:14.00
##  Median :6.000   Mode  :character   Mode  :character   Median :17.00
##  Mean   :5.889                                         Mean   :16.86
##  3rd Qu.:8.000                                         3rd Qu.:19.00
##  Max.   :8.000                                         Max.   :35.00
##       hwy             fl                class
##  Min.   :12.00   Length:234         Length:234
##  1st Qu.:18.00   Class :character   Class :character
##  Median :24.00   Mode  :character   Mode  :character
##  Mean   :23.44
##  3rd Qu.:27.00
##  Max.   :44.00
```

```
# mpg
#     "cty"             .
# cty       (city miles per gallon)     .

#
#     , "cty"          "hwy" (     )  "displ" (    )
cor(mpg$cty, mpg$hwy, use = "complete.obs")  # cty  hwy
```

```
## [1] 0.9559159
```

```
cor(mpg$cty, mpg$displ, use = "complete.obs") # cty  displ
```

```
## [1] -0.798524
```

```
#
#    "cty"         "hwy"  "displ"
model <- lm(cty ~ hwy + displ, data = mpg)

#         "hwy"  "displ"
predicted_values <- predict(model, newdata = mpg[missing_index, ])

#
mpg$cty[missing_index] <- predicted_values

#
sum(is.na(mpg$cty))  #
```

```
## [1] 0
```

#(d) [5pts] multiple linear regression , . .

```r
model <- lm(mpg ~ cyl + disp + hp + drat + wt + qsec + vs + am + gear + carb, data = mtcars)

#
summary(model)
```

```
##
## Call:
## lm(formula = mpg ~ cyl + disp + hp + drat + wt + qsec + vs +
##     am + gear + carb, data = mtcars)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -3.4506 -1.6044 -0.1196  1.2193  4.6271
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 12.30337   18.71788   0.657   0.5181
## cyl         -0.11144    1.04502  -0.107   0.9161
## disp         0.01334    0.01786   0.747   0.4635
## hp          -0.02148    0.02177  -0.987   0.3350
## drat         0.78711    1.63537   0.481   0.6353
## wt          -3.71530    1.89441  -1.961   0.0633 .
## qsec         0.82104    0.73084   1.123   0.2739
## vs           0.31776    2.10451   0.151   0.8814
## am           2.52023    2.05665   1.225   0.2340
## gear         0.65541    1.49326   0.439   0.6652
## carb        -0.19942    0.82875  -0.241   0.8122
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.65 on 21 degrees of freedom
## Multiple R-squared:  0.869,  Adjusted R-squared:  0.8066
## F-statistic: 13.93 on 10 and 21 DF,  p-value: 3.793e-07
```

# : # (Estimate): 1 , mpg ( ) . , wt( ) -3.800894 . 1 (mpg) 3.8 . am( ) 2.938108 , (am=1) (am=0) 2.94 .

# (Std. Error): . .

#t-value p-value: t-value: , . p-value: mpg . p-value 0.05 . , wt( ) am( ) p-value 0.05 , . , cyl( ) , qsec, vs, gear, carb .

#R-squared ( ): Multiple R-squared: . 0.8935 , 89.35% . Adjusted R-squared: R-squared , R-squared . 0.8454 , . # F-statistic p-value: F-statistic: . 18.92 , p-value 6.17e-06 . .

#(e) [5pts] stepwise regression , . .

```r
step(model)
```

```
## Start:  AIC=70.9
## mpg ~ cyl + disp + hp + drat + wt + qsec + vs + am + gear + carb
##
##        Df Sum of Sq    RSS    AIC
## - cyl   1    0.0799 147.57 68.915
```

```
## - vs    1    0.1601 147.66 68.932
## - carb  1    0.4067 147.90 68.986
## - gear  1    1.3531 148.85 69.190
## - drat  1    1.6270 149.12 69.249
## - disp  1    3.9167 151.41 69.736
## - hp    1    6.8399 154.33 70.348
## - qsec  1    8.8641 156.36 70.765
## <none>              147.49 70.898
## - am    1   10.5467 158.04 71.108
## - wt    1   27.0144 174.51 74.280
##
## Step:  AIC=68.92
## mpg ~ disp + hp + drat + wt + qsec + vs + am + gear + carb
##
##        Df Sum of Sq    RSS    AIC
## - vs    1    0.2685 147.84 66.973
## - carb  1    0.5201 148.09 67.028
## - gear  1    1.8211 149.40 67.308
## - drat  1    1.9826 149.56 67.342
## - disp  1    3.9009 151.47 67.750
## - hp    1    7.3632 154.94 68.473
## <none>              147.57 68.915
## - qsec  1   10.0933 157.67 69.032
## - am    1   11.8359 159.41 69.384
## - wt    1   27.0280 174.60 72.297
##
## Step:  AIC=66.97
## mpg ~ disp + hp + drat + wt + qsec + am + gear + carb
##
##        Df Sum of Sq    RSS    AIC
## - carb  1    0.6855 148.53 65.121
## - gear  1    2.1437 149.99 65.434
## - drat  1    2.2139 150.06 65.449
## - disp  1    3.6467 151.49 65.753
## - hp    1    7.1060 154.95 66.475
## <none>              147.84 66.973
## - am    1   11.5694 159.41 67.384
## - qsec  1   15.6830 163.53 68.200
## - wt    1   27.3799 175.22 70.410
##
## Step:  AIC=65.12
## mpg ~ disp + hp + drat + wt + qsec + am + gear
##
##        Df Sum of Sq    RSS    AIC
## - gear  1     1.565 150.09 63.457
## - drat  1     1.932 150.46 63.535
## <none>              148.53 65.121
## - disp  1    10.110 158.64 65.229
## - am    1    12.323 160.85 65.672
## - hp    1    14.826 163.35 66.166
## - qsec  1    26.408 174.94 68.358
## - wt    1    69.127 217.66 75.350
##
## Step:  AIC=63.46
```

```
## mpg ~ disp + hp + drat + wt + qsec + am
##
##        Df Sum of Sq    RSS    AIC
## - drat  1     3.345 153.44 62.162
## - disp  1     8.545 158.64 63.229
## <none>              150.09 63.457
## - hp    1    13.285 163.38 64.171
## - am    1    20.036 170.13 65.466
## - qsec  1    25.574 175.67 66.491
## - wt    1    67.572 217.66 73.351
##
## Step:  AIC=62.16
## mpg ~ disp + hp + wt + qsec + am
##
##        Df Sum of Sq    RSS    AIC
## - disp  1     6.629 160.07 61.515
## <none>              153.44 62.162
## - hp    1    12.572 166.01 62.682
## - qsec  1    26.470 179.91 65.255
## - am    1    32.198 185.63 66.258
## - wt    1    69.043 222.48 72.051
##
## Step:  AIC=61.52
## mpg ~ hp + wt + qsec + am
##
##        Df Sum of Sq    RSS    AIC
## - hp    1     9.219 169.29 61.307
## <none>              160.07 61.515
## - qsec  1    20.225 180.29 63.323
## - am    1    25.993 186.06 64.331
## - wt    1    78.494 238.56 72.284
##
## Step:  AIC=61.31
## mpg ~ wt + qsec + am
##
##        Df Sum of Sq    RSS    AIC
## <none>              169.29 61.307
## - am    1    26.178 195.46 63.908
## - qsec  1   109.034 278.32 75.217
## - wt    1   183.347 352.63 82.790
##
## Call:
## lm(formula = mpg ~ wt + qsec + am, data = mtcars)
##
## Coefficients:
## (Intercept)           wt         qsec           am
##       9.618       -3.917        1.226        2.936
```

```
summary(step(model))
```

```
## Start:  AIC=70.9
## mpg ~ cyl + disp + hp + drat + wt + qsec + vs + am + gear + carb
##
##        Df Sum of Sq    RSS    AIC
```

```
## - cyl   1    0.0799 147.57 68.915
## - vs    1    0.1601 147.66 68.932
## - carb  1    0.4067 147.90 68.986
## - gear  1    1.3531 148.85 69.190
## - drat  1    1.6270 149.12 69.249
## - disp  1    3.9167 151.41 69.736
## - hp    1    6.8399 154.33 70.348
## - qsec  1    8.8641 156.36 70.765
## <none>              147.49 70.898
## - am    1   10.5467 158.04 71.108
## - wt    1   27.0144 174.51 74.280
##
## Step:  AIC=68.92
## mpg ~ disp + hp + drat + wt + qsec + vs + am + gear + carb
##
##        Df Sum of Sq    RSS    AIC
## - vs    1    0.2685 147.84 66.973
## - carb  1    0.5201 148.09 67.028
## - gear  1    1.8211 149.40 67.308
## - drat  1    1.9826 149.56 67.342
## - disp  1    3.9009 151.47 67.750
## - hp    1    7.3632 154.94 68.473
## <none>              147.57 68.915
## - qsec  1   10.0933 157.67 69.032
## - am    1   11.8359 159.41 69.384
## - wt    1   27.0280 174.60 72.297
##
## Step:  AIC=66.97
## mpg ~ disp + hp + drat + wt + qsec + am + gear + carb
##
##        Df Sum of Sq    RSS    AIC
## - carb  1    0.6855 148.53 65.121
## - gear  1    2.1437 149.99 65.434
## - drat  1    2.2139 150.06 65.449
## - disp  1    3.6467 151.49 65.753
## - hp    1    7.1060 154.95 66.475
## <none>              147.84 66.973
## - am    1   11.5694 159.41 67.384
## - qsec  1   15.6830 163.53 68.200
## - wt    1   27.3799 175.22 70.410
##
## Step:  AIC=65.12
## mpg ~ disp + hp + drat + wt + qsec + am + gear
##
##        Df Sum of Sq    RSS    AIC
## - gear  1     1.565 150.09 63.457
## - drat  1     1.932 150.46 63.535
## <none>              148.53 65.121
## - disp  1    10.110 158.64 65.229
## - am    1    12.323 160.85 65.672
## - hp    1    14.826 163.35 66.166
## - qsec  1    26.408 174.94 68.358
## - wt    1    69.127 217.66 75.350
##
```

```
## Step:  AIC=63.46
## mpg ~ disp + hp + drat + wt + qsec + am
##
##        Df Sum of Sq    RSS    AIC
## - drat  1     3.345 153.44 62.162
## - disp  1     8.545 158.64 63.229
## <none>              150.09 63.457
## - hp    1    13.285 163.38 64.171
## - am    1    20.036 170.13 65.466
## - qsec  1    25.574 175.67 66.491
## - wt    1    67.572 217.66 73.351
##
## Step:  AIC=62.16
## mpg ~ disp + hp + wt + qsec + am
##
##        Df Sum of Sq    RSS    AIC
## - disp  1     6.629 160.07 61.515
## <none>              153.44 62.162
## - hp    1    12.572 166.01 62.682
## - qsec  1    26.470 179.91 65.255
## - am    1    32.198 185.63 66.258
## - wt    1    69.043 222.48 72.051
##
## Step:  AIC=61.52
## mpg ~ hp + wt + qsec + am
##
##        Df Sum of Sq    RSS    AIC
## - hp    1     9.219 169.29 61.307
## <none>              160.07 61.515
## - qsec  1    20.225 180.29 63.323
## - am    1    25.993 186.06 64.331
## - wt    1    78.494 238.56 72.284
##
## Step:  AIC=61.31
## mpg ~ wt + qsec + am
##
##        Df Sum of Sq    RSS    AIC
## <none>              169.29 61.307
## - am    1    26.178 195.46 63.908
## - qsec  1   109.034 278.32 75.217
## - wt    1   183.347 352.63 82.790
##
## Call:
## lm(formula = mpg ~ wt + qsec + am, data = mtcars)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -3.4811 -1.5555 -0.7257  1.4110  4.6610
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   9.6178     6.9596   1.382 0.177915
## wt           -3.9165     0.7112  -5.507 6.95e-06 ***
```

```
## qsec          1.2259     0.2887   4.247 0.000216 ***
## am            2.9358     1.4109   2.081 0.046716 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.459 on 28 degrees of freedom
## Multiple R-squared:  0.8497, Adjusted R-squared:  0.8336
## F-statistic: 52.75 on 3 and 28 DF,  p-value: 1.21e-11
```

#AIC

#(f) [15pts] regression tree . Cross validation deviance optimal tree . optimal tree .

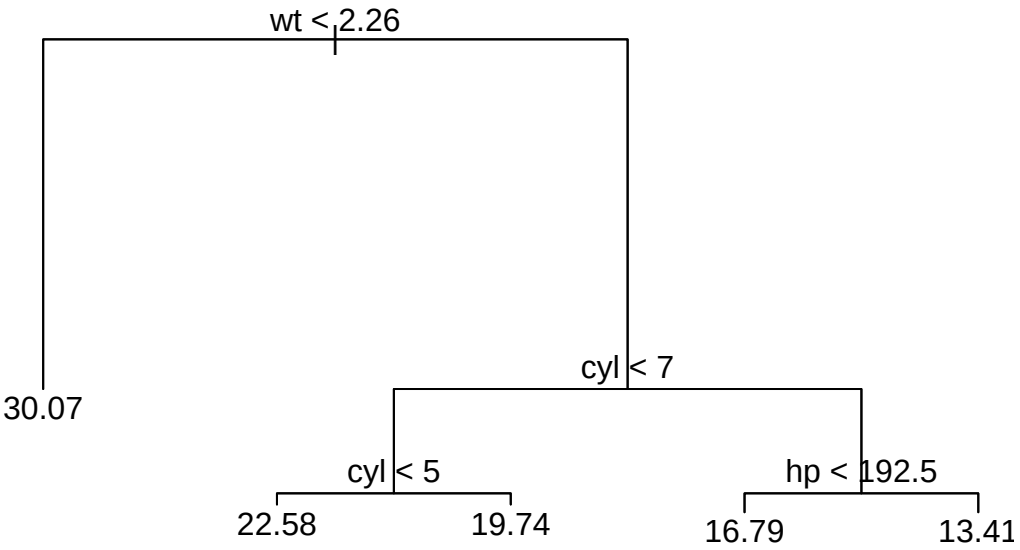
```
library(tree)
data(mtcars)
X=mtcars[,-1]
Y=mtcars$mpg
str(mtcars)
```

```
## 'data.frame':    32 obs. of  11 variables:
##  $ mpg : num  21 21 22.8 21.4 18.7 18.1 14.3 24.4 22.8 19.2 ...
##  $ cyl : num  6 6 4 6 8 6 8 4 4 6 ...
##  $ disp: num  160 160 108 258 360 ...
##  $ hp  : num  110 110 93 110 175 105 245 62 95 123 ...
##  $ drat: num  3.9 3.9 3.85 3.08 3.15 2.76 3.21 3.69 3.92 3.92 ...
##  $ wt  : num  2.62 2.88 2.32 3.21 3.44 ...
##  $ qsec: num  16.5 17 18.6 19.4 17 ...
##  $ vs  : num  0 0 1 1 0 1 0 1 1 1 ...
##  $ am  : num  1 1 1 0 0 0 0 0 0 0 ...
##  $ gear: num  4 4 4 3 3 3 3 4 4 4 ...
##  $ carb: num  4 4 1 1 2 1 4 2 2 4 ...
```

```
head(mtcars)
```

```
##                   mpg cyl disp  hp drat    wt  qsec vs am gear carb
## Mazda RX4         21.0   6  160 110 3.90 2.620 16.46  0  1    4    4
## Mazda RX4 Wag     21.0   6  160 110 3.90 2.875 17.02  0  1    4    4
## Datsun 710        22.8   4  108  93 3.85 2.320 18.61  1  1    4    1
## Hornet 4 Drive    21.4   6  258 110 3.08 3.215 19.44  1  0    3    1
## Hornet Sportabout 18.7   8  360 175 3.15 3.440 17.02  0  0    3    2
## Valiant           18.1   6  225 105 2.76 3.460 20.22  1  0    3    1
```

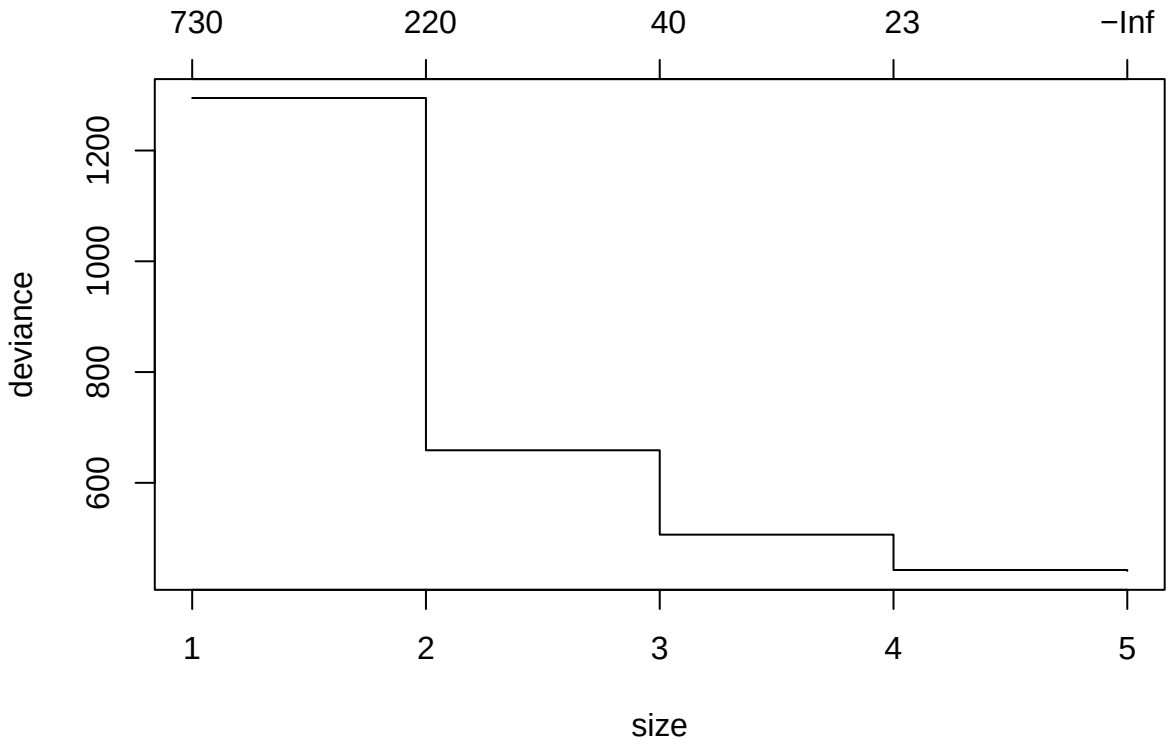
```
tr1=tree(mpg~.,data=mtcars)

plot(tr1);text(tr1)
```



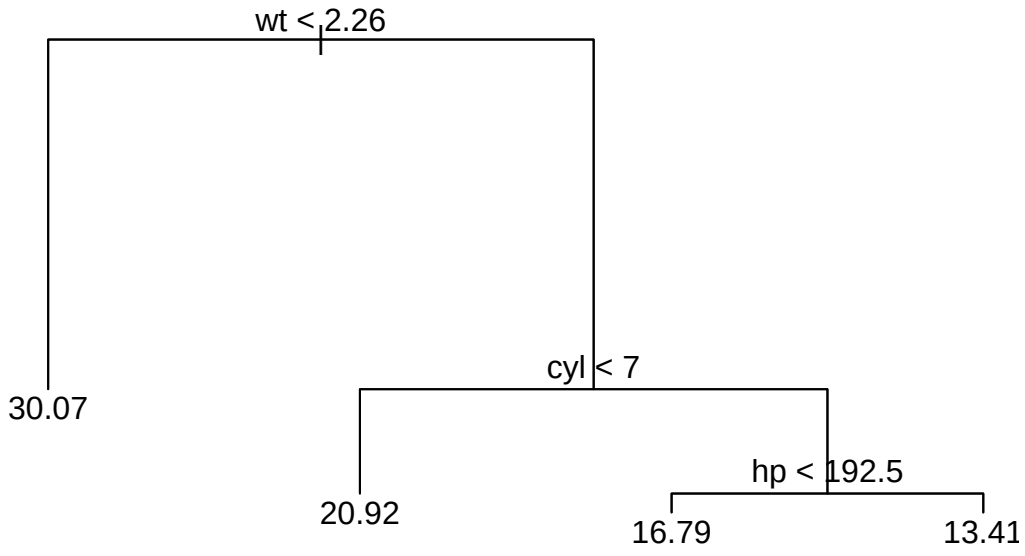

```
#cv_model=train(mpg~.,data=mtcars,method="")

#optimum tree using minimum CV error
tr1.cv=cv.tree(tr1) #cross validation
tr1.cv
```

```
## $size
## [1] 5 4 3 2 1
##
## $dev
## [1]  440.9999  442.6667  506.4521  658.7918 1294.7802
##
## $k
## [1]      -Inf  23.47736  39.78286 219.24404 734.92732
##
## $method
## [1] "deviance"
##
## attr(,"class")
## [1] "prune"         "tree.sequence"
```

```
plot(tr1.cv) #4
```




```
final.tr=prune.tree(tr1,best=4)
plot(final.tr);text(final.tr)
```




```
rt.prediction.cyl=predict(final.tr,mtcars)
rt.prediction.cyl
```

```
##           Mazda RX4       Mazda RX4 Wag          Datsun 710      Hornet 4 Drive
##            20.92500            20.92500            20.92500            20.92500
##   Hornet Sportabout             Valiant          Duster 360           Merc 240D
##            16.78571            20.92500            13.41429            20.92500
##            Merc 230            Merc 280           Merc 280C          Merc 450SE
##            20.92500            20.92500            20.92500            16.78571
##          Merc 450SL         Merc 450SLC  Cadillac Fleetwood Lincoln Continental
```

```
##            16.78571            16.78571            13.41429            13.41429
##   Chrysler Imperial             Fiat 128          Honda Civic       Toyota Corolla
##            13.41429            30.06667            30.06667            30.06667
##       Toyota Corona     Dodge Challenger          AMC Javelin           Camaro Z28
##            20.92500            16.78571            16.78571            13.41429
##    Pontiac Firebird            Fiat X1-9       Porsche 914-2        Lotus Europa
##            16.78571            30.06667            30.06667            30.06667
##      Ford Pantera L         Ferrari Dino        Maserati Bora           Volvo 142E
##            13.41429            20.92500            13.41429            20.92500
```

```
mean((mtcars$cyl-predict(final.tr,mtcars))^2)
```

```
## [1] 244.1849
```

#1[2pts] regression tree fitted value ? # . , fitted value .

#( [3pts] ?

# [3pt] terminal node ? terminal node fitted value ?

```
nrow(subset(mtcars,wt<2.26))
```

```
## [1] 6
```

```
#6    ,       30.7
```

# [7pts] .

#(g) [5pts] (stepwise regression optimal tree) normalized mean squared error (NMSE) . ?

```
##NMSE=(sum(yi-yihat)^2)/sum(yi-yimean)^2
set.seed(123)
library(MASS)
library(rpart)
library(caret)
##train data
trainIndex=createDataPartition(mtcars$mpg,p=0.7,list=FALSE)
train_data=mtcars[trainIndex,]
test_data=mtcars[-trainIndex,]

#lm model
stepwise_model=lm(mpg~.,data=train_data)
stepwise_model=stepAIC(stepwise_model,direction="both")
```

```
## Start:  AIC=53.9
## mpg ~ cyl + disp + hp + drat + wt + qsec + vs + am + gear + carb
##
##        Df Sum of Sq     RSS    AIC
## - vs    1    0.1106  90.793 51.933
## - disp  1    2.9053  93.587 52.660
## - am    1    3.0851  93.767 52.706
## - hp    1    3.2106  93.893 52.738
## - cyl   1    6.1379  96.820 53.475
## - carb  1    6.9552  97.637 53.677
## - drat  1    7.8619  98.544 53.899
## <none>               90.682 53.903
## - qsec  1    9.8768 100.559 54.385
```

```
## - gear  1   10.8537 101.536 54.617
## - wt    1   16.4558 107.138 55.906
##
## Step:  AIC=51.93
## mpg ~ cyl + disp + hp + drat + wt + qsec + am + gear + carb
##
##        Df Sum of Sq     RSS    AIC
## - disp  1    3.1643  93.957 50.755
## - hp    1    3.3181  94.111 50.794
## - am    1    4.2199  95.013 51.023
## - carb  1    6.8556  97.648 51.680
## <none>               90.793 51.933
## - drat  1    7.9658  98.759 51.951
## - cyl   1    8.7438  99.537 52.139
## - qsec  1    9.8923 100.685 52.415
## - gear  1   11.0543 101.847 52.690
## + vs    1    0.1106  90.682 53.903
## - wt    1   16.8282 107.621 54.013
##
## Step:  AIC=50.75
## mpg ~ cyl + hp + drat + wt + qsec + am + gear + carb
##
##        Df Sum of Sq     RSS    AIC
## - hp    1    1.4041  95.361 49.111
## - am    1    4.9109  98.868 49.978
## - qsec  1    8.1007 102.058 50.740
## <none>               93.957 50.755
## - drat  1   10.7149 104.672 51.347
## - gear  1   11.6278 105.585 51.555
## + disp  1    3.1643  90.793 51.933
## - cyl   1   14.5789 108.536 52.217
## - carb  1   14.6370 108.594 52.230
## + vs    1    0.3695  93.587 52.660
## - wt    1   17.4668 111.424 52.847
##
## Step:  AIC=49.11
## mpg ~ cyl + drat + wt + qsec + am + gear + carb
##
##        Df Sum of Sq     RSS    AIC
## - am    1    4.2329  99.594 48.153
## <none>               95.361 49.111
## - qsec  1    9.4538 104.815 49.379
## - gear  1   12.8044 108.166 50.135
## - cyl   1   13.6330 108.994 50.318
## - drat  1   14.4737 109.835 50.502
## + hp    1    1.4041  93.957 50.755
## + disp  1    1.2502  94.111 50.794
## + vs    1    0.3819  94.979 51.014
## - wt    1   17.9702 113.331 51.254
## - carb  1   24.0645 119.426 52.511
##
## Step:  AIC=48.15
## mpg ~ cyl + drat + wt + qsec + gear + carb
##
```

```
##        Df Sum of Sq     RSS    AIC
## - qsec  1     5.287 104.881 47.395
## <none>               99.594 48.153
## - cyl   1    11.056 110.650 48.680
## + am    1     4.233  95.361 49.111
## - drat  1    14.198 113.792 49.352
## + disp  1     1.967  97.627 49.675
## + vs    1     1.956  97.638 49.677
## + hp    1     0.726  98.868 49.978
## - wt    1    18.323 117.917 50.206
## - gear  1    27.625 127.219 52.029
## - carb  1    34.474 134.068 53.287
##
## Step:  AIC=47.39
## mpg ~ cyl + drat + wt + gear + carb
##
##        Df Sum of Sq     RSS    AIC
## - cyl   1     5.932 110.813 46.715
## <none>              104.881 47.395
## - drat  1    11.346 116.227 47.860
## + qsec  1     5.287  99.594 48.153
## - wt    1    13.263 118.144 48.252
## + hp    1     2.271 102.610 48.869
## + vs    1     0.360 104.521 49.312
## + disp  1     0.151 104.730 49.360
## + am    1     0.066 104.815 49.379
## - gear  1    28.453 133.334 51.155
## - carb  1    44.559 149.441 53.892
##
## Step:  AIC=46.71
## mpg ~ drat + wt + gear + carb
##
##        Df Sum of Sq    RSS    AIC
## - drat  1     6.062 116.88 45.993
## <none>              110.81 46.715
## + cyl   1     5.932 104.88 47.395
## + vs    1     3.943 106.87 47.845
## + disp  1     3.001 107.81 48.056
## + am    1     0.614 110.20 48.582
## + hp    1     0.397 110.42 48.629
## + qsec  1     0.163 110.65 48.680
## - wt    1    21.160 131.97 48.909
## - gear  1    26.168 136.98 49.803
## - carb  1    65.994 176.81 55.928
##
## Step:  AIC=45.99
## mpg ~ wt + gear + carb
##
##        Df Sum of Sq    RSS    AIC
## <none>              116.88 45.993
## + drat  1     6.062 110.81 46.715
## + hp    1     2.594 114.28 47.455
## + vs    1     1.749 115.13 47.631
## + disp  1     1.407 115.47 47.702
```

```
## + cyl   1     0.648 116.23 47.860
## + qsec  1     0.632 116.24 47.863
## + am    1     0.593 116.28 47.871
## - gear  1    41.086 157.96 51.223
## - wt    1    48.498 165.37 52.324
## - carb  1    62.682 179.56 54.299
```

```
#tree model
tree_model=rpart(mpg~.,data=train_data)

#make predictions on the test set
pred_stepwise=predict(stepwise_model,newdata=test_data)
pred_tree=predict(tree_model,newdata=test_data)

#Calculate NMSE for both models

nmse=function(y_true,y_pred){
  mse=mean((y_true-y_pred)^2)
  var_y=var(y_true)
  return(mse/var_y)
}
#actual y value for the test set
y_true=test_data$mpg

#calculate NMSE for the two models
nmse_stepwise=nmse(y_true,pred_stepwise)
nmse_tree=nmse(y_true,pred_tree)
print(nmse_stepwise)
```

```
## [1] 0.4618337
```

```
print(nmse_tree)
```

```
## [1] 0.7874121
```

```
if (nmse_stepwise < nmse_tree) {
  cat("Stepwise Regression model performs better.\n")
} else {
  cat("Optimal Tree model performs better.\n")
}
```

```
## Stepwise Regression model performs better.
```

#4. [10pts|Answer the following question.

#A. 4pts 2 ? 2 .

# ( ) ( ) .

**1. (Accuracy / Predictive Performance):**

**- . , ( : MSE, RMSE, MAE ) ( , , ) .**

**- . (overfitting) .**

**2. (Model Complexity / Simplicity):**

#- , , . # - , **(overfitting)** . # - (generalization) .

---

**2**

# , . **Bias-Variance Trade-off**( - ) .

**- (Bias-Variance Trade-off):**

- **(Bias)**: . , . , .
- **(Variance)**: . , **(overfitting)** . , .

**:**

- ( ) , .
- ( ) , ( ).

**?**

**, .**

#- , .

# , . (regularization), (cross-validation), ( : AIC, BIC, cross-validation ) , .

#B. [3pts] knearest neighborhood , observation . , standardize . standardize .

**#K-Nearest Neighbors (KNN)** . , , .

**:**

1. : # - KNN , . # - , , **(km)** , ( ) . ( ) . # - , , , .

#2. : # - ( : 0~1 , 0~1000 ), . # - , A 0 1 , B 0 1000 , B **A** .

**Standardization ( )**

# , . **(standardization)** . .

#1. : 0 1 , . #2. : KNN , . #3. : , .

**:**

#KNN . (standardization) , . .

#C. [3pts] cv-errorL test set prediction error . training set

# **training set** **(overfitting)** . **(biased evaluation)** , :

**1. (Overfitting) :**

#- **Training set** . , **training set** . # training set , . , **(overfitting)** . #- ( , **test set** ) , .

**2. :**

- , . ( , **training error**) .
- **Training error** , . , .

**3. Test set :**

#- **Test set** . . . #- **Test error** , .

**4. (Cross-Validation) :**

#- **(cross-validation, CV)** , . # - , . , .

**:**

#**Training set** . , . **test set** .

```
#
set.seed(123)
x=matrix(sample(20),nrow=4)
x[x[,4]>15,]
```

```
##      [,1] [,2] [,3] [,4] [,5]
## [1,]   15   10    5   20   16
## [2,]   19    2    4   17    1
```

```
x[order(x[,4]),]
```

```
##      [,1] [,2] [,3] [,4] [,5]
## [1,]    3   11    9    7   13
## [2,]   14    6   18   12    8
## [3,]   19    2    4   17    1
## [4,]   15   10    5   20   16
```

```r
x[sample(4),]
```

```
##      [,1] [,2] [,3] [,4] [,5]
## [1,]   15   10    5   20   16
## [2,]    3   11    9    7   13
## [3,]   14    6   18   12    8
## [4,]   19    2    4   17    1
```

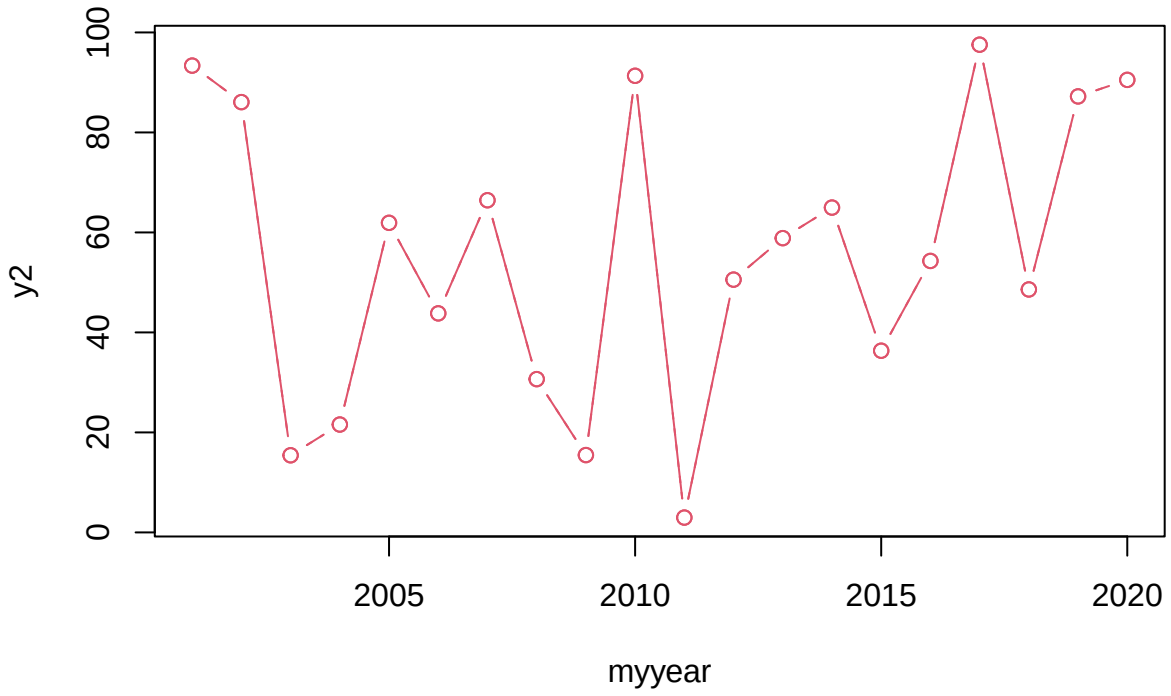
```r
# ,
#qnorm
#dnorm
#pnorm
#rnorm(n,mena=0,)
#apply(row.sum)

#
```

```r
re=read.table("/Users/eunsimkim/Ewha/24_2/Lifescience/Data/NIFED.DAT.rdata")
head(re)
```

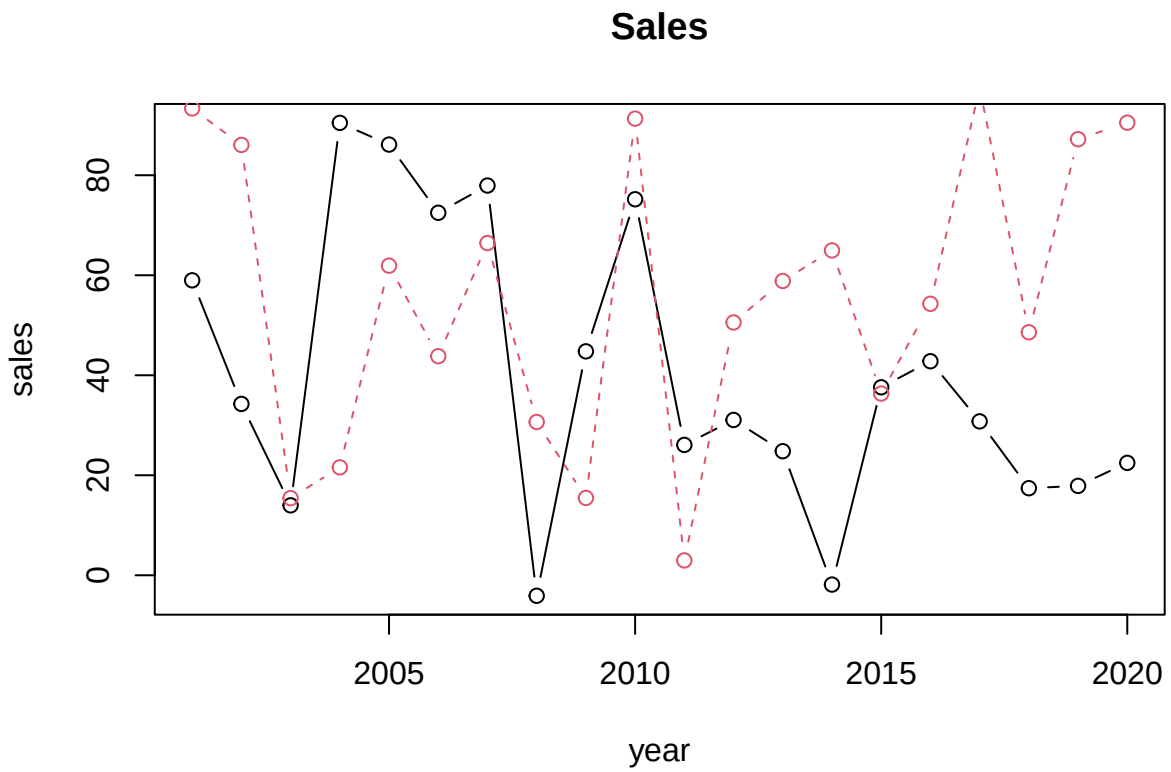
```
##       V1
## 1   RDA2
## 2      A
## 3      2
## 4 131585
## 5  66560
## 6   1026
```

```r
algae <- read.table('https://home.ewha.ac.kr/~josong/dm/Analysis.txt',
             header=F,
             dec='.',
             col.names=c('season','size','speed','mxPH','mnO2','Cl','NO3',
             'NH4','oPO4','PO4','Chla','a1','a2','a3','a4','a5','a6','a7'),
             na.strings=c('XXXXXXX'),
             stringsAsFactors=T)
```

```r
#ch2

myyear=2001:2020
y1=100*runif(20)+rnorm(20,0,5)
y2=103*runif(20)+rnorm(20,0,5)
plot(myyear,y2,type="b",lty=2,col=2)
lines(myyear,y2,type="b",col=2)
```




```
plot(myyear,y1,type="b",main="Sales",xlab="year",ylab="sales")
lines(myyear,y2,type="b",lty=2,col=2)
```




```
#legend("toplight",c('amazon','nvidia',lty=1:2,col=1:2))
#
#KNN  myinpute
#
#train,       .     ,              :
```

```
#  linear: stepwise; localsearch ; AIC   .    ,
#tree: recurrsive binary split.      : <terminal node           . observed fited value MSE, R, test data
#cross-validation    .

#

#  coeffiecient y=0.99+99x1+22X2+
```