Recitation Session

Sep 18, 2020

Basic commands

```
x <- c(1,3,2,5) # combine
x
```

```
## [1] 1 3 2 5
```

```
x = c(1,6,2)
x
```

```
## [1] 1 6 2
```

```
y = c(1,4,3)
length(x)
```

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

```
length(y)
```

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

```
x+y
```

```
## [1]  2 10  5
```

```
ls() # list: find out all variables
```

```
## [1] "x" "y"
```

```
rm(x,y) # delete variables
ls()
```

```
## character(0)
```

```
?matrix
x=matrix(data=c(1,2,3,4), nrow=2, ncol=2)
x
```

```
##      [,1] [,2]
## [1,]    1    3
## [2,]    2    4
```

```r
matrix(c(1,2,3,4),2,2,byrow=TRUE) # the matrix should be filled by rows
```

```
##      [,1] [,2]
## [1,]    1    2
## [2,]    3    4
```

```r
sqrt(x)
```

```
##          [,1]     [,2]
## [1,] 1.000000 1.732051
## [2,] 1.414214 2.000000
```

```r
x^2
```

```
##      [,1] [,2]
## [1,]    1    9
## [2,]    4   16
```

```r
x=rnorm(50) # normal distributions for 50 numbers
y=x+rnorm(50,mean=50,sd=.1)
cor(x,y) # cor() computes the correlation coefficient
```

```
## [1] 0.9959471
```

```r
set.seed(1303) # Set the seed of R's random number generator, which is useful for creating simulations
rnorm(50) # generates a vector of 50 pseudo-random normals with mean 0 and variance 1.
```

```
##  [1] -1.1439763145  1.3421293656  2.1853904757  0.5363925179  0.0631929665
##  [6]  0.5022344825 -0.0004167247  0.5658198405 -0.5725226890 -1.1102250073
## [11] -0.0486871234 -0.6956562176  0.8289174803  0.2066528551 -0.2356745091
## [16] -0.5563104914 -0.3647543571  0.8623550343 -0.6307715354  0.3136021252
## [21] -0.9314953177  0.8238676185  0.5233707021  0.7069214120  0.4202043256
## [26] -0.2690521547 -1.5103172999 -0.6902124766 -0.1434719524 -1.0135274099
## [31]  1.5732737361  0.0127465055  0.8726470499  0.4220661905 -0.0188157917
## [36]  2.6157489689 -0.6931401748 -0.2663217810 -0.7206364412  1.3677342065
## [41]  0.2640073322  0.6321868074 -1.3306509858  0.0268888182  1.0406363208
## [46]  1.3120237985 -0.0300020767 -0.2500257125  0.0234144857  1.6598706557
```

```r
set.seed(3)
y=rnorm(100)
mean(y)
```

```
## [1] 0.01103557
```

```r
var(y)
```

```
## [1] 0.7328675
```

```r
sqrt(var(y))
```

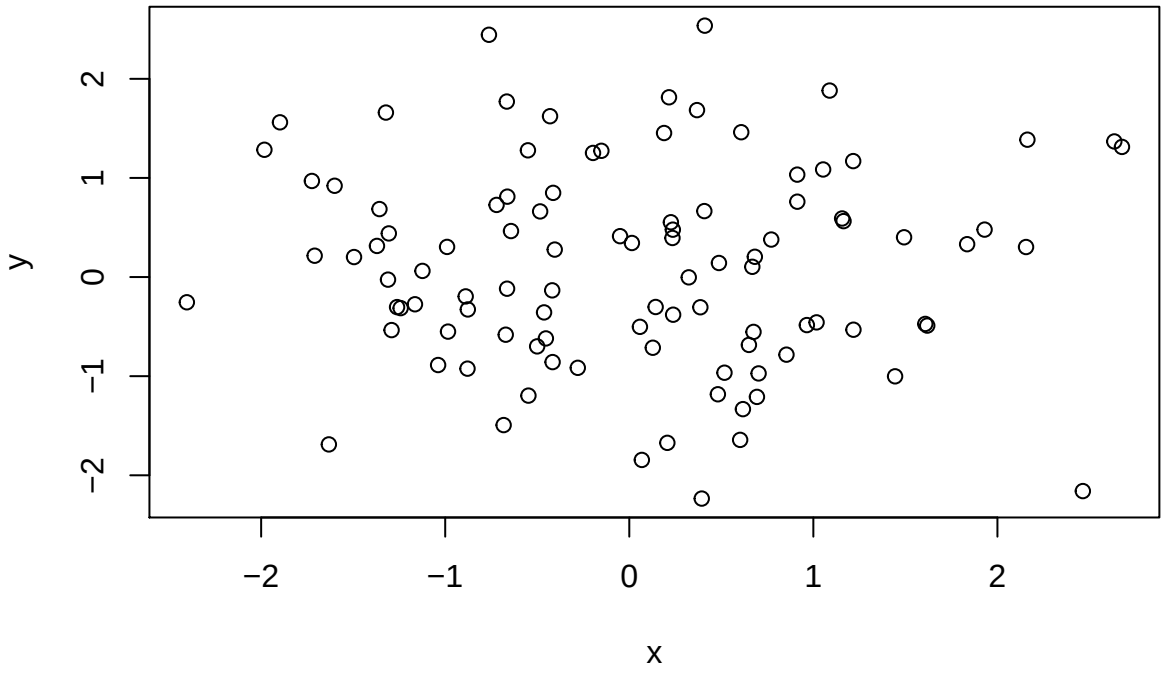
```
## [1] 0.8560768
```

```r
sd(y)
```

```
## [1] 0.8560768
```

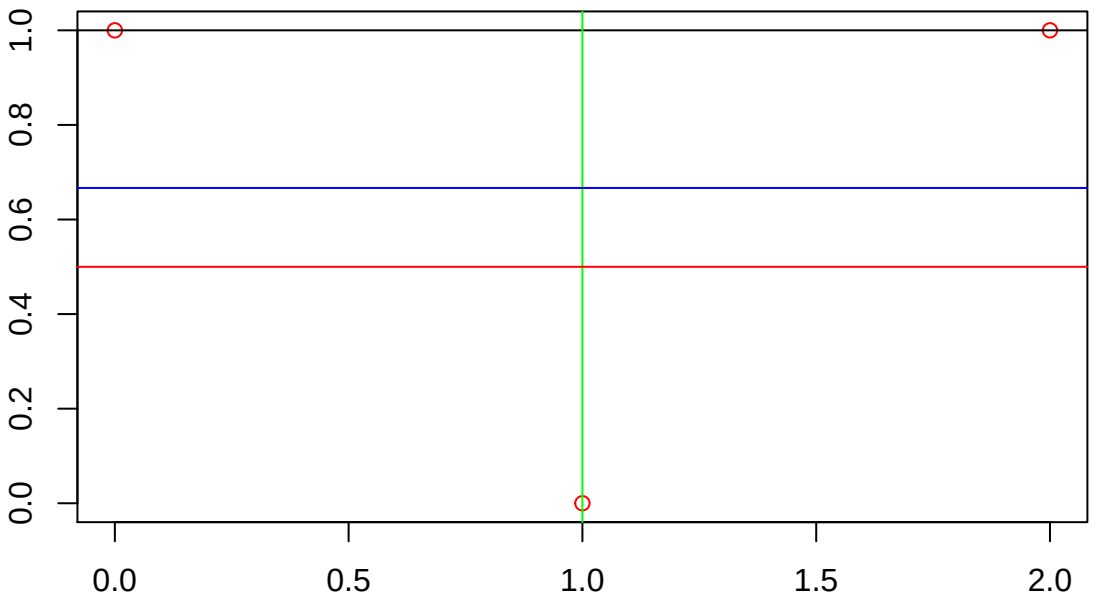
##Graphics

```r
x=rnorm(100)
y=rnorm(100)
plot(x,y)
```




```r
plot(x,y,xlab="this is the x-axis",ylab="this is the y-axis",main="Plot of X vs Y")
```


```
plot(c(0,1,2),c(1,0,1),xlab="",ylab="",col="red") # plot 3 dots
abline(1,0,col="black") # add straight line (horizontal)
abline(v=1,col="green") # add straight line (vertical)
abline(2/3,0,col="blue")
abline(0.5,0,col="red")
```



Sequence

```
x=seq(1,10)
x
```

```
##  [1]  1  2  3  4  5  6  7  8  9 10
```

```
x=1:10
x
```

```
##  [1]  1  2  3  4  5  6  7  8  9 10
```

```
x=seq(-pi,pi,length=50)
x
```

```
##  [1] -3.14159265 -3.01336438 -2.88513611 -2.75690784 -2.62867957
##  [6] -2.50045130 -2.37222302 -2.24399475 -2.11576648 -1.98753821
## [11] -1.85930994 -1.73108167 -1.60285339 -1.47462512 -1.34639685
## [16] -1.21816858 -1.08994031 -0.96171204 -0.83348377 -0.70525549
## [21] -0.57702722 -0.44879895 -0.32057068 -0.19234241 -0.06411414
## [26]  0.06411414  0.19234241  0.32057068  0.44879895  0.57702722
## [31]  0.70525549  0.83348377  0.96171204  1.08994031  1.21816858
## [36]  1.34639685  1.47462512  1.60285339  1.73108167  1.85930994
## [41]  1.98753821  2.11576648  2.24399475  2.37222302  2.50045130
## [46]  2.62867957  2.75690784  2.88513611  3.01336438  3.14159265
```

Indexing Data

```
A=matrix(1:16,4,4)
A
```

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

```
A[2,3]
```

```
## [1] 10
```

```
A[c(1,3),c(2,4)]
```

```
##      [,1] [,2]
## [1,]    5   13
## [2,]    7   15
```

```r
A[1:3,2:4]
```

```
##      [,1] [,2] [,3]
## [1,]    5    9   13
## [2,]    6   10   14
## [3,]    7   11   15
```

```r
A[1:2,]
```

```
##      [,1] [,2] [,3] [,4]
## [1,]    1    5    9   13
## [2,]    2    6   10   14
```

```r
A[,1:2]
```

```
##      [,1] [,2]
## [1,]    1    5
## [2,]    2    6
## [3,]    3    7
## [4,]    4    8
```

```r
A[1,]
```

```
## [1]  1  5  9 13
```

```r
A
```

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

```r
A[-c(1,3),] # delete row 1&3
```

```
##      [,1] [,2] [,3] [,4]
## [1,]    2    6   10   14
## [2,]    4    8   12   16
```

```r
A[-c(1,3),-c(1,3,4)] # delete row 1&3 and column 1&3&4
```

```
## [1] 6 8
```

```r
dim(A)
```

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

Loading Data

```
Auto=read.csv("Auto.csv",header=T,na.strings="?")
dim(Auto)
```

```
## [1] 397   9
```

```
Auto[1:4,]
```

```
##   mpg cylinders displacement horsepower weight acceleration year origin
## 1  18         8          307        130   3504         12.0   70      1
## 2  15         8          350        165   3693         11.5   70      1
## 3  18         8          318        150   3436         11.0   70      1
## 4  16         8          304        150   3433         12.0   70      1
##                       name
## 1 chevrolet chevelle malibu
## 2         buick skylark 320
## 3        plymouth satellite
## 4             amc rebel sst
```

```
#na.omit returns the object with incomplete cases removed.
Auto=na.omit(Auto)
dim(Auto)
```

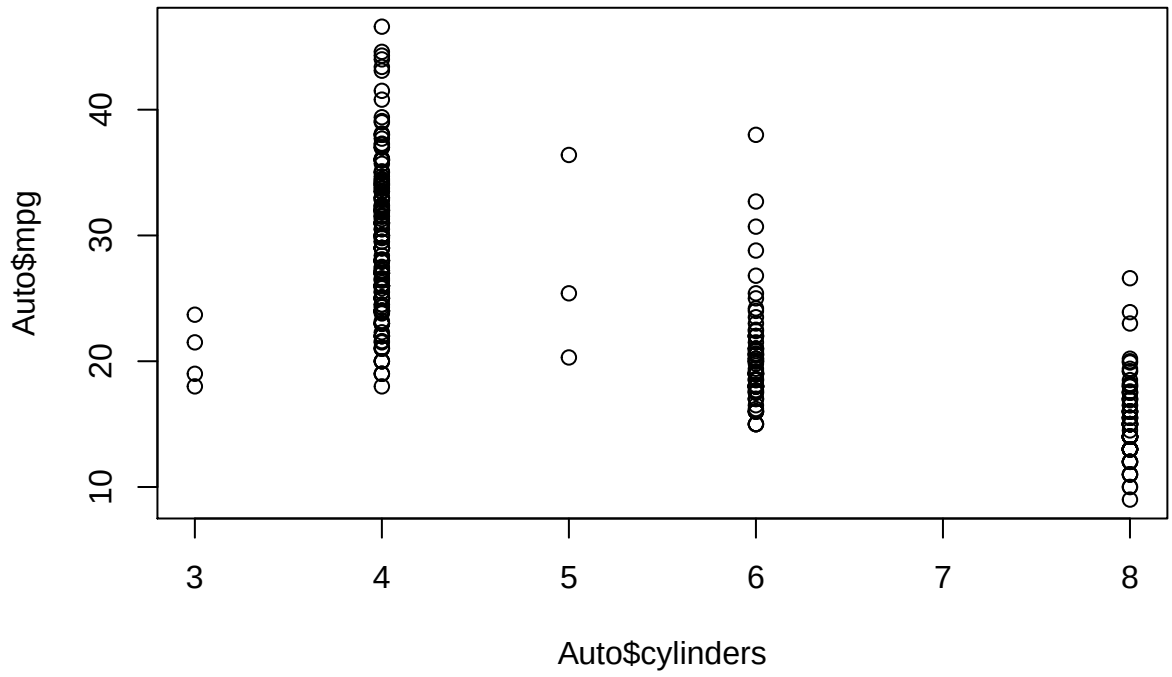
```
## [1] 392   9
```

```
names(Auto)
```

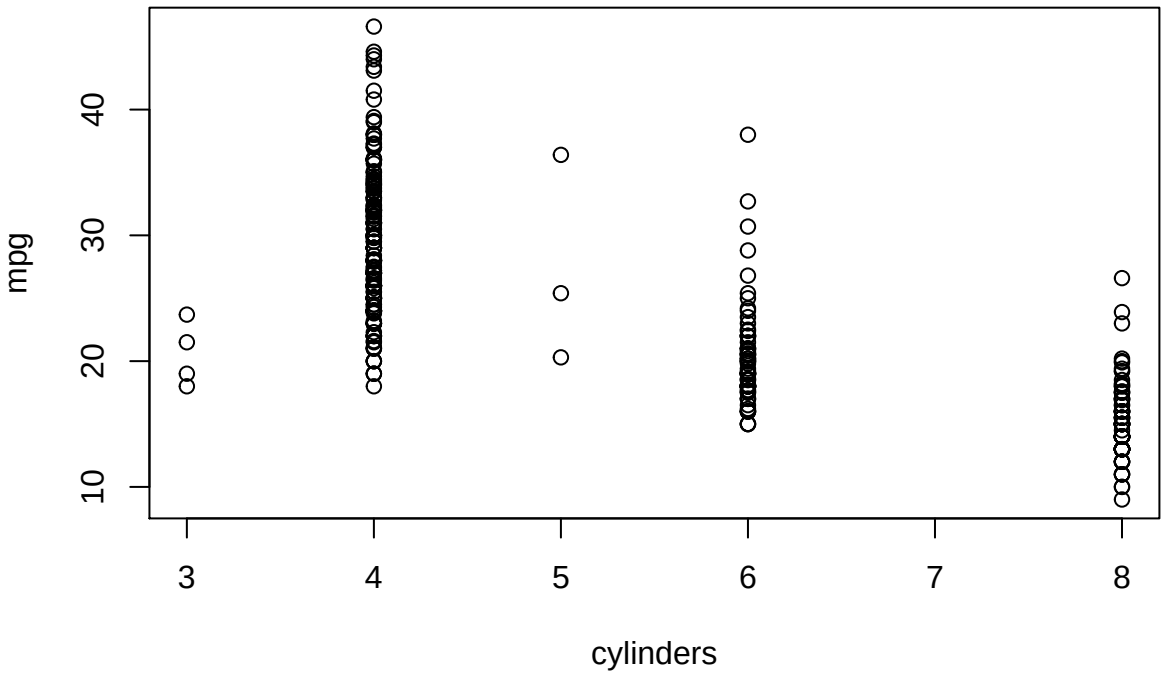
```
## [1] "mpg"          "cylinders"    "displacement" "horsepower"
## [5] "weight"       "acceleration" "year"         "origin"
## [9] "name"
```

#Additional Graphical and Numerical Summaries

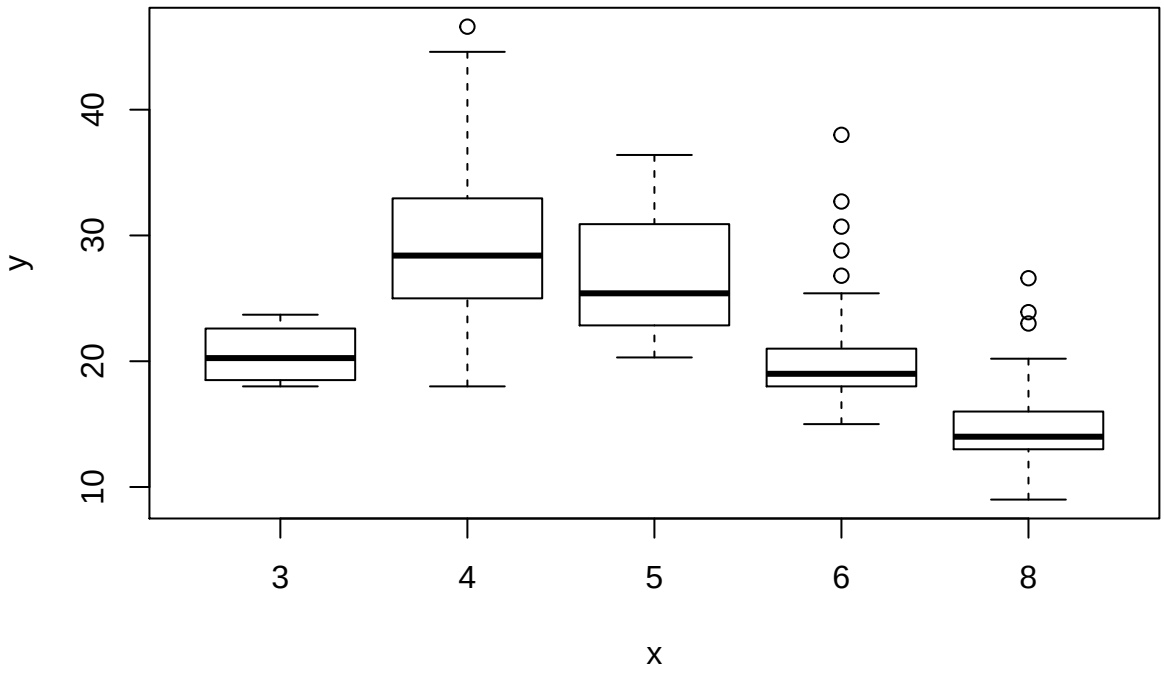
```
plot(Auto$cylinders, Auto$mpg)
```



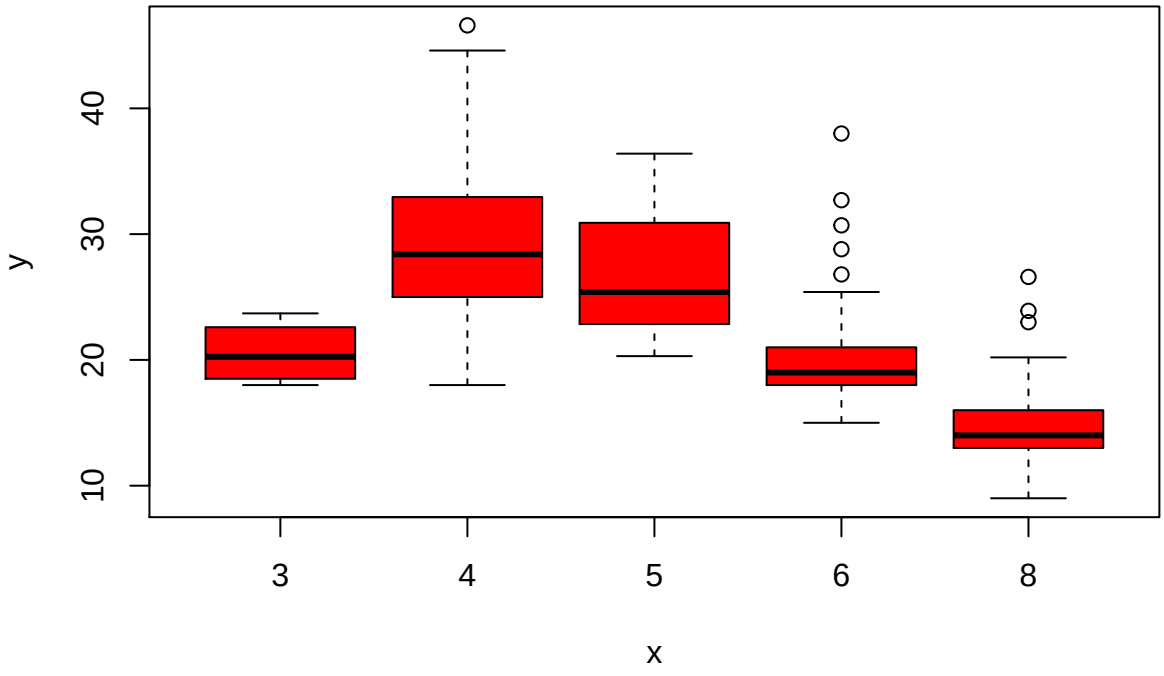

```
attach(Auto)#After that, objects can be accessed by simply their names.
plot(cylinders, mpg)
```



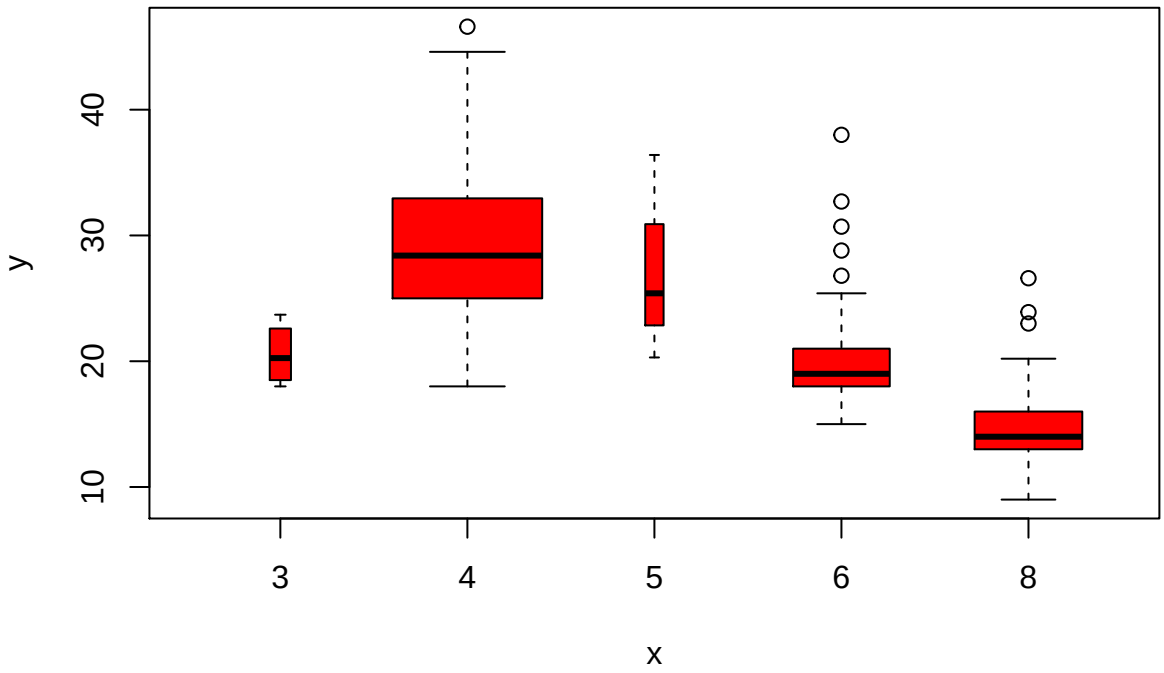

```
cylinders=as.factor(cylinders) # Convert a column into a factor column.
plot(cylinders, mpg)
```



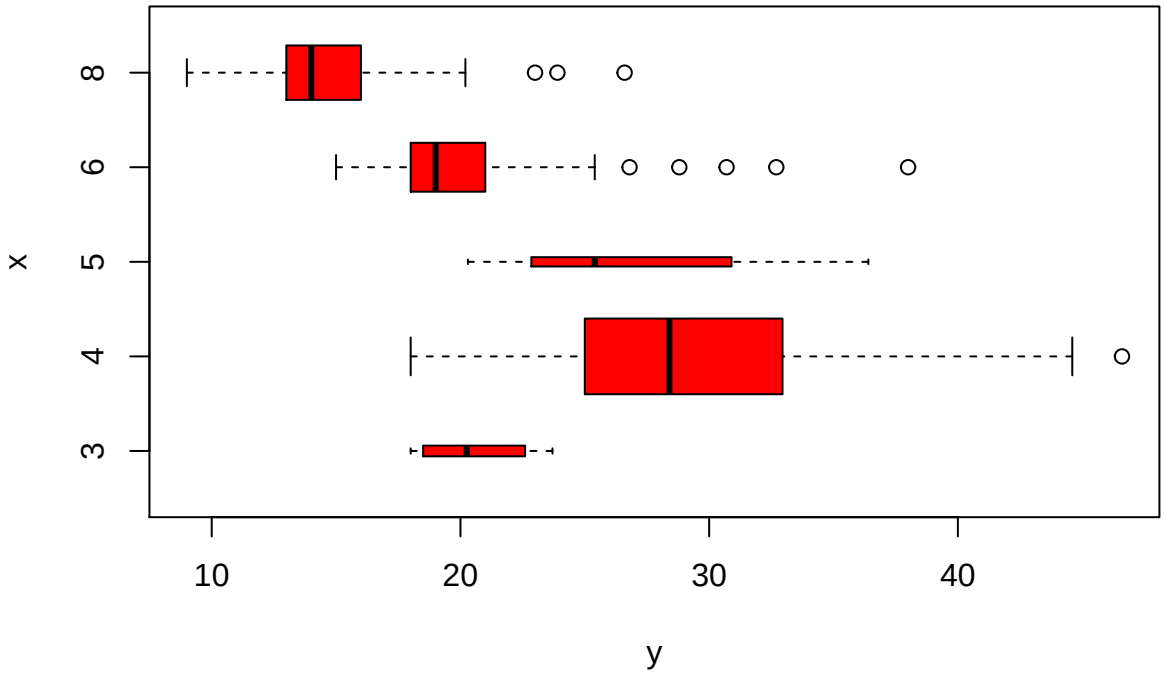

```
plot(cylinders, mpg, col="red")
```



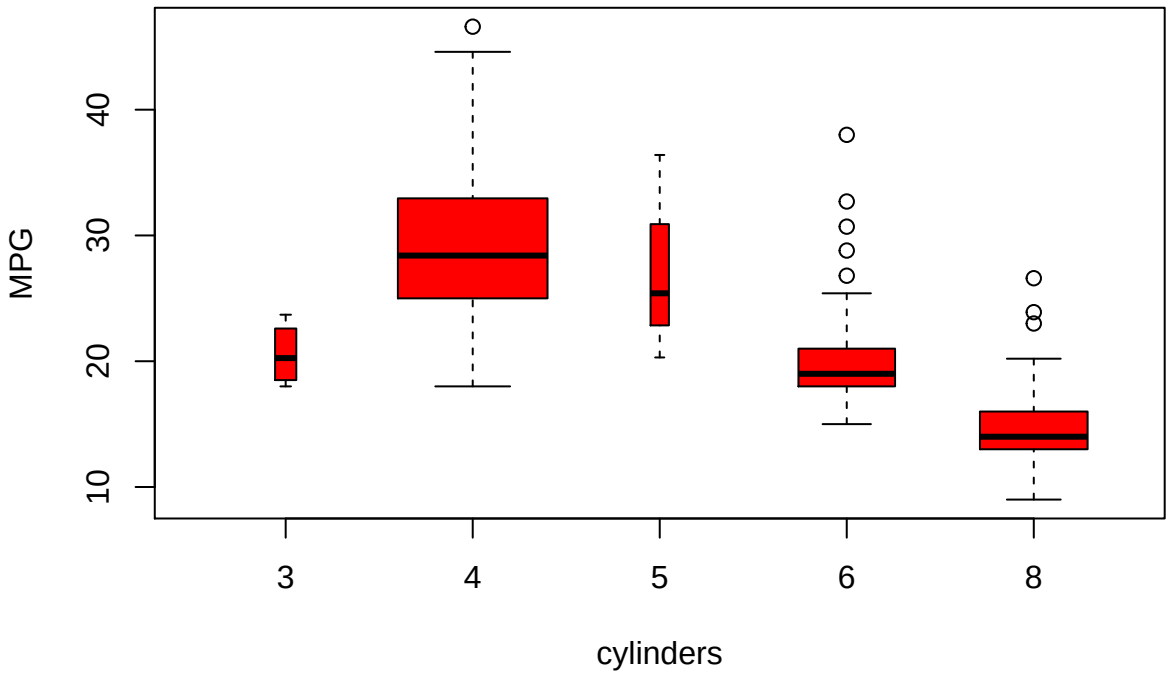

```
# Set varwidth=TRUE when you want to make widths of box proportional to the square root of the sample s
plot(cylinders, mpg, col="red", varwidth=T)
```



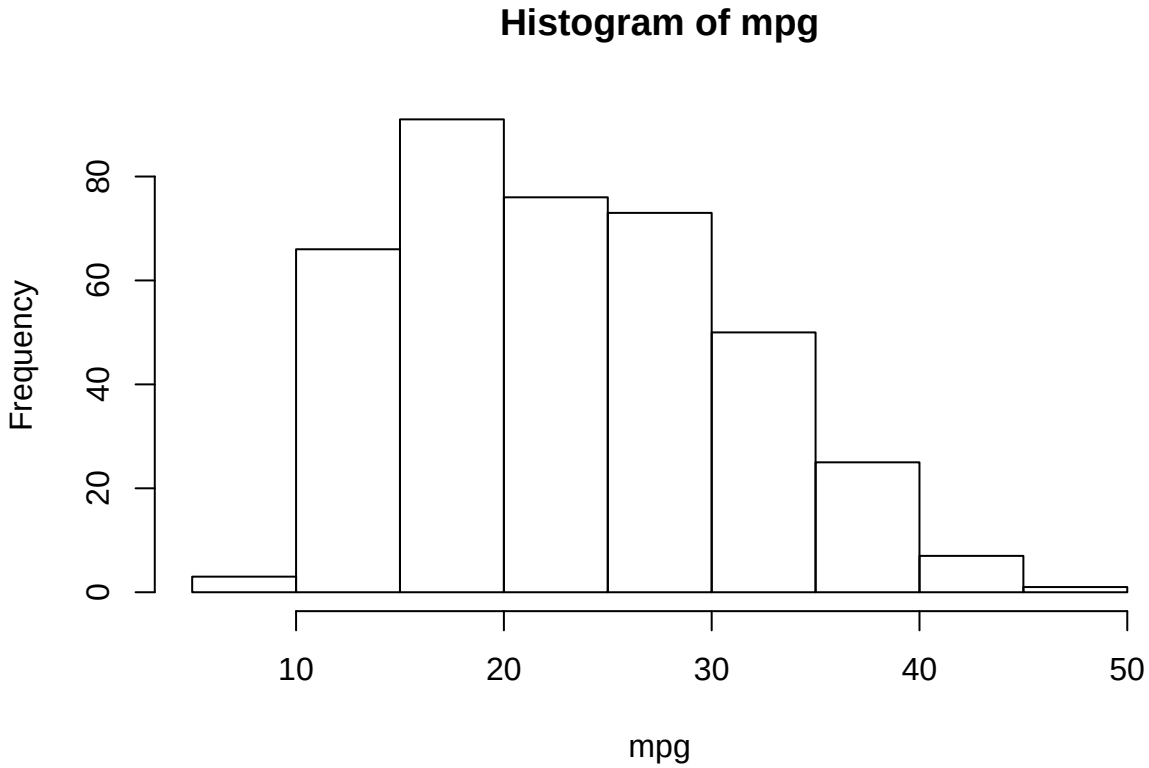

```
plot(cylinders, mpg, col="red", varwidth=T,horizontal=T)
```



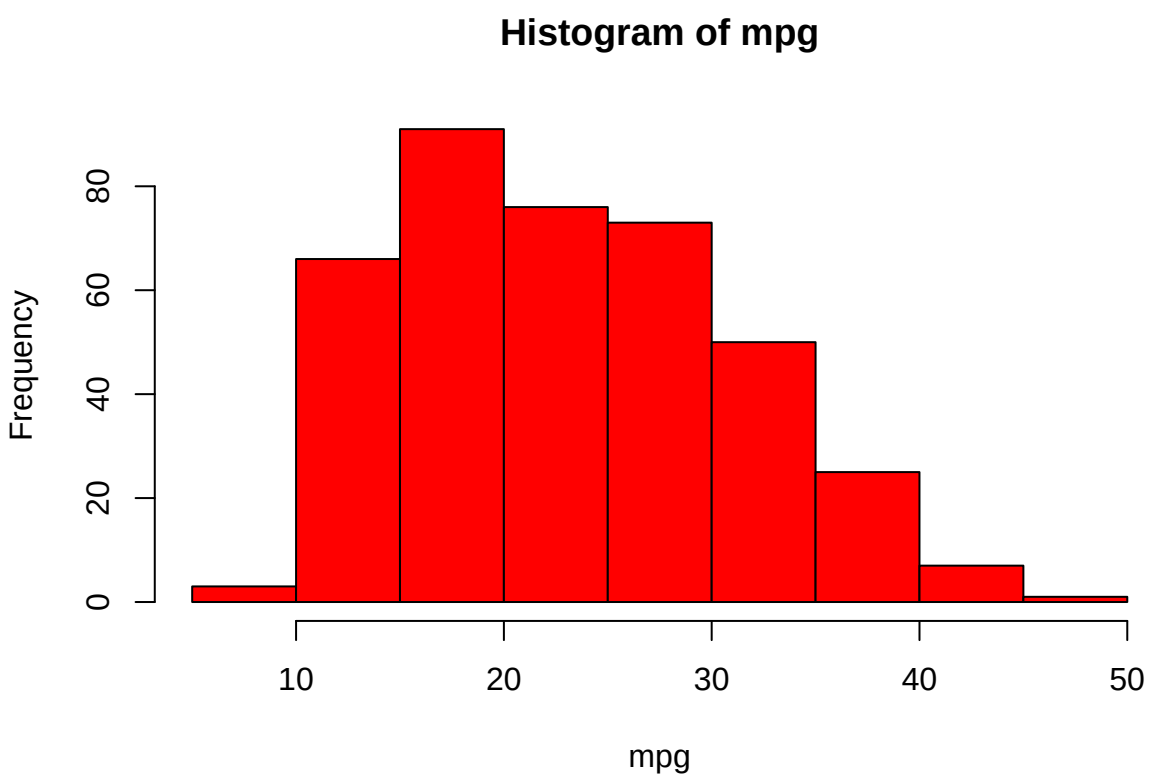

```
plot(cylinders, mpg, col="red", varwidth=T, xlab="cylinders", ylab="MPG")
```


MPG
10
20
30
40
3
4
5
6
8
cylinders

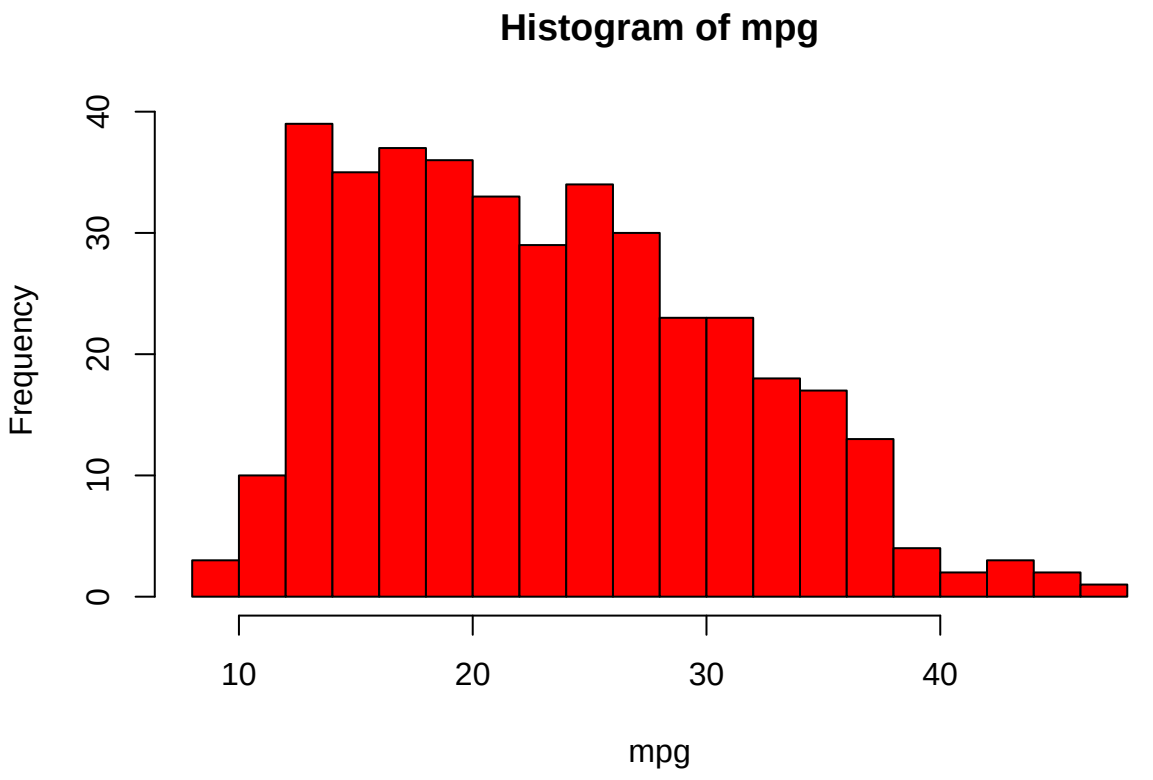
```
hist(mpg)
```


Histogram of mpg
Frequency
0
20
40
60
80
10
20
30
40
50
mpg

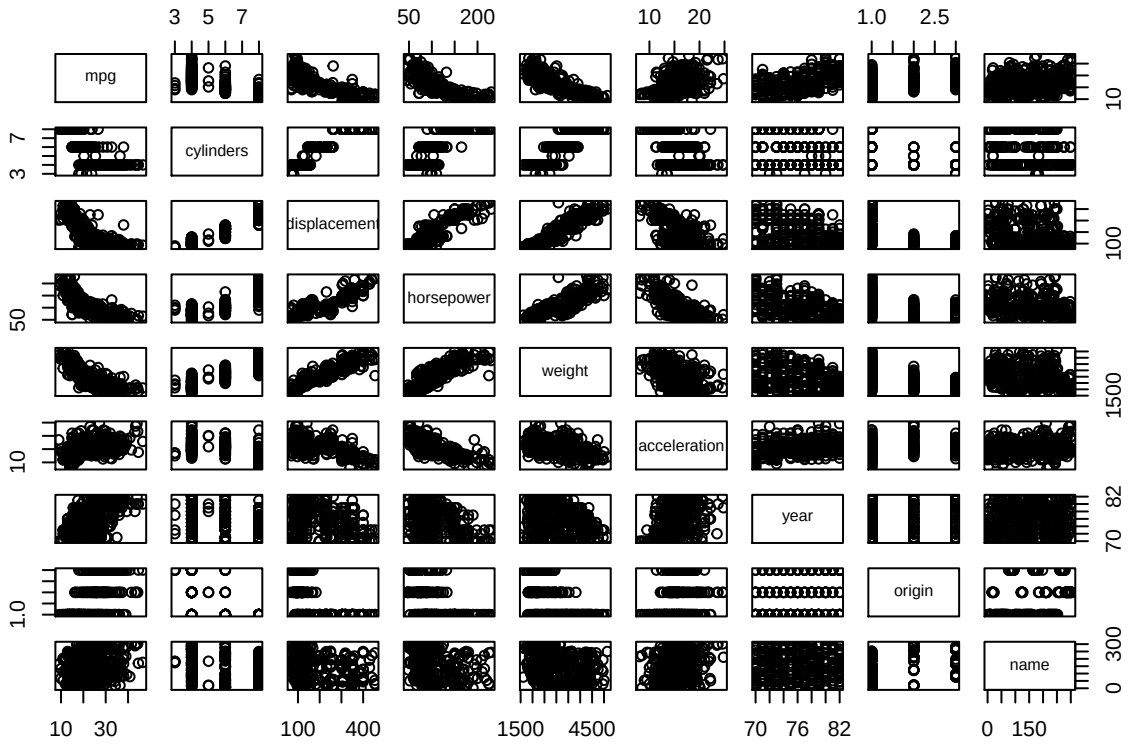
```r
hist(mpg,col=2)
```



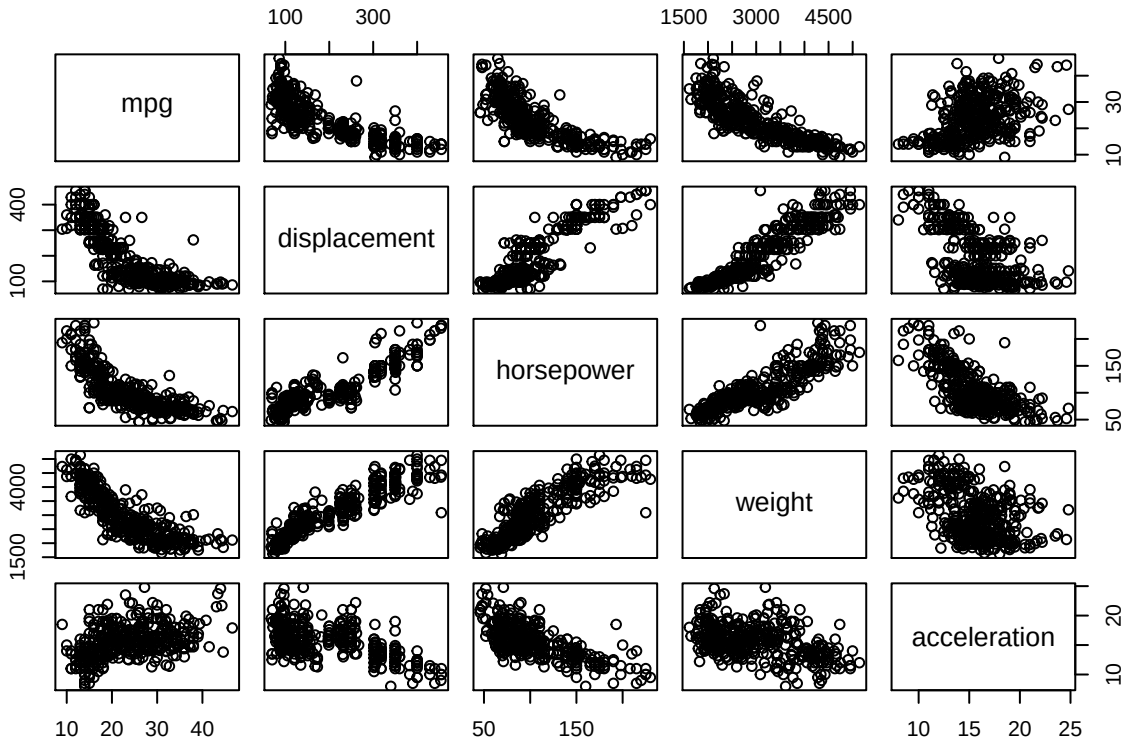

```r
hist(mpg,col=2,breaks=15)
```



```
pairs(Auto)
```




```
# only several variable selected
pairs(~ mpg + displacement + horsepower + weight + acceleration, Auto)
```



```
summary(Auto)
```

```
##       mpg          cylinders      displacement     horsepower
##  Min.   : 9.00   Min.   :3.000   Min.   : 68.0   Min.   : 46.0
##  1st Qu.:17.00   1st Qu.:4.000   1st Qu.:105.0   1st Qu.: 75.0
##  Median :22.75   Median :4.000   Median :151.0   Median : 93.5
##  Mean   :23.45   Mean   :5.472   Mean   :194.4   Mean   :104.5
##  3rd Qu.:29.00   3rd Qu.:8.000   3rd Qu.:275.8   3rd Qu.:126.0
##  Max.   :46.60   Max.   :8.000   Max.   :455.0   Max.   :230.0
##
##      weight      acceleration        year           origin
##  Min.   :1613   Min.   : 8.00   Min.   :70.00   Min.   :1.000
##  1st Qu.:2225   1st Qu.:13.78   1st Qu.:73.00   1st Qu.:1.000
##  Median :2804   Median :15.50   Median :76.00   Median :1.000
##  Mean   :2978   Mean   :15.54   Mean   :75.98   Mean   :1.577
##  3rd Qu.:3615   3rd Qu.:17.02   3rd Qu.:79.00   3rd Qu.:2.000
##  Max.   :5140   Max.   :24.80   Max.   :82.00   Max.   :3.000
##
##                  name
##  amc matador       :  5
##  ford pinto        :  5
##  toyota corolla    :  5
##  amc gremlin       :  4
##  amc hornet        :  4
##  chevrolet chevette:  4
##  (Other)           :365
```

```
summary(mpg)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    9.00   17.00   22.75   23.45   29.00   46.60
```

Writing Functions

```
my_func<-function(x,y){
 return(x+y)
 }
```

```
my_func(1,2)
```

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

library

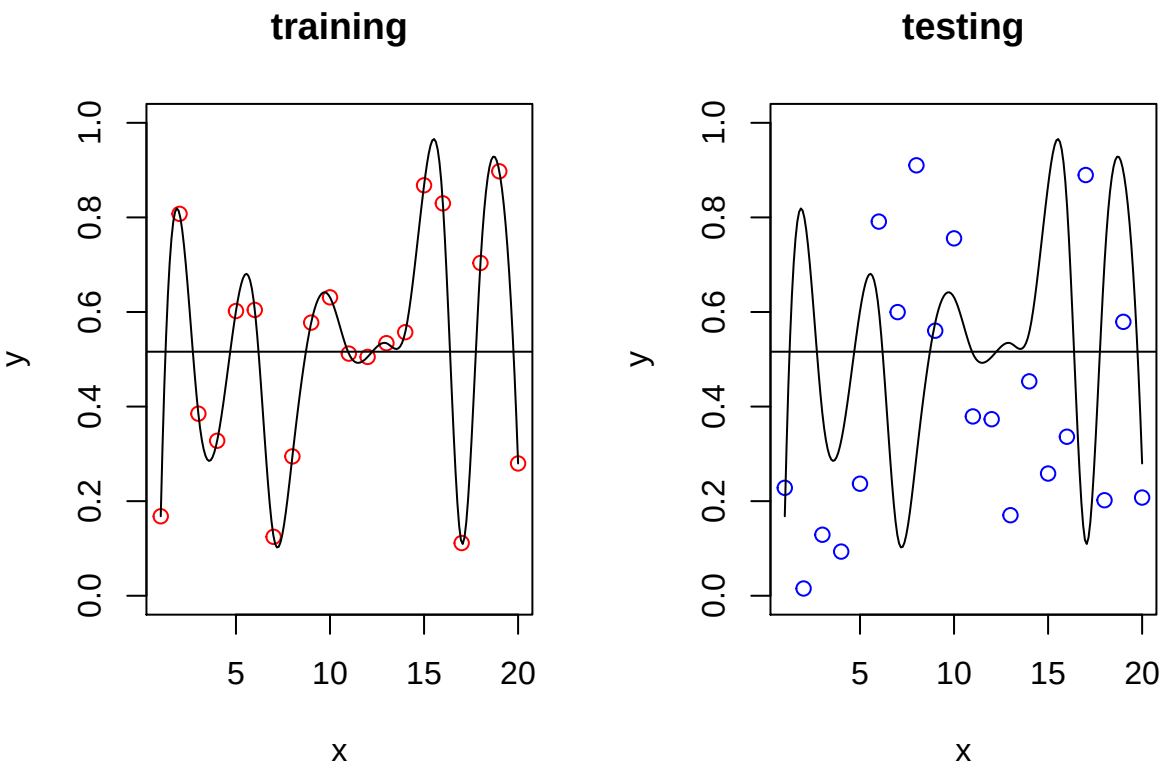
```
library(MASS)
library(ISLR)
```

```
##
## Attaching package: 'ISLR'

## The following object is masked _by_ '.GlobalEnv':
##
##     Auto
```

#Example Code (Lecture 1)

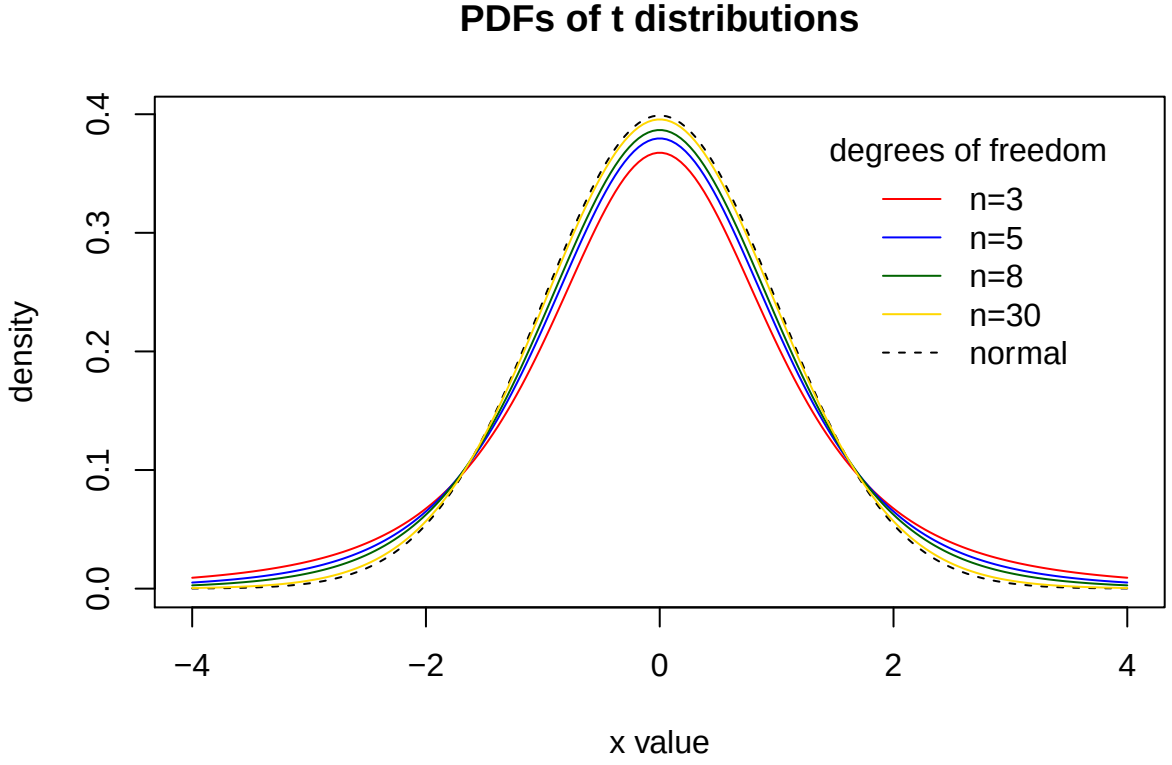
#Overfitting Problem

```
set.seed(3)
x<-c(1:20)
y<-runif(max(x)) # generates 20 uniform random numbers between 0 and 1.
z<-runif(max(x))
par(mfrow=c(1,2)) # create a matrix of 1 x 2 plots that are filled in by row.

#training dataset
plot(x,y,col='red',ylim=c(0,1),xlab='x',ylab='y',main='training')
abline(mean(y),0)
lines(spline(x, y, n = 201))

#testing dataset
plot(x,z,col='blue',ylim=c(0,1),xlab='x',ylab='y',main='testing')
abline(mean(y),0)
lines(spline(x, y, n = 201))
```

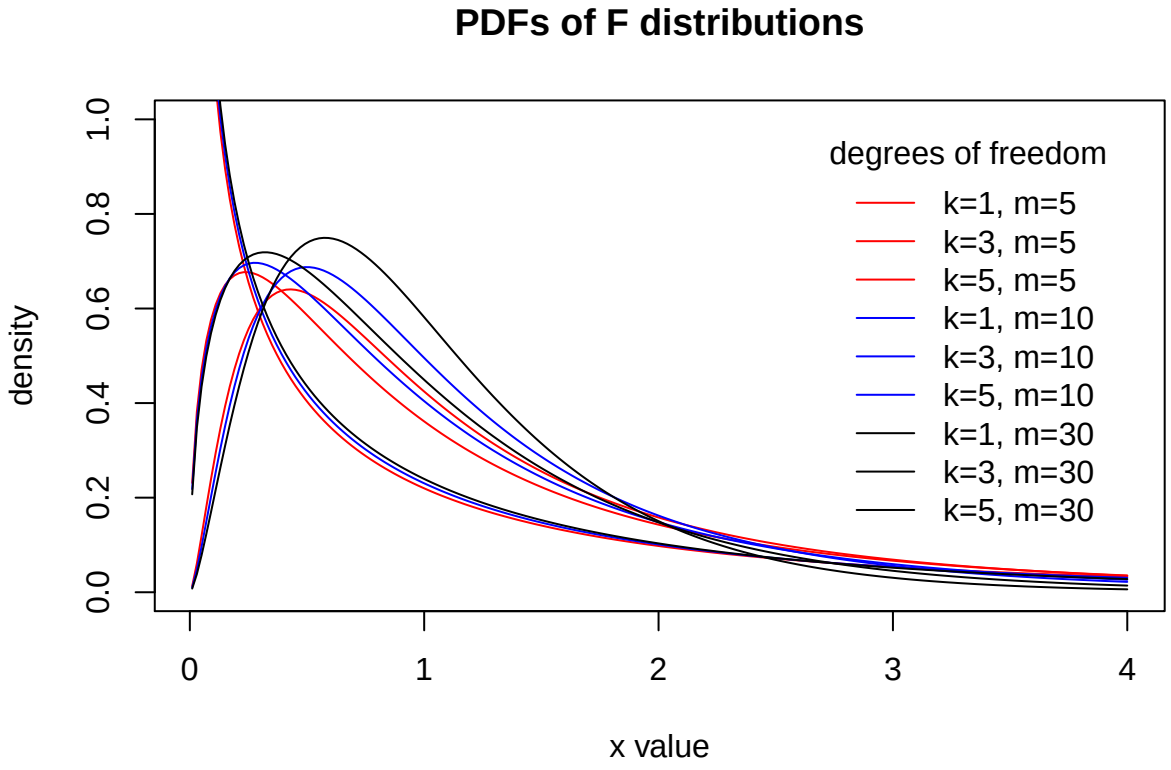


#Basic Distributions

```r
x <- seq(-4, 4, length=200)

#normal distrubution
hx <- dnorm(x)
degf <- c(3, 5, 8, 30)
colors <- c("red", "blue", "darkgreen", "gold", "black")
labels <- c("n=3", "n=5", "n=8", "n=30", "normal")

#t distribution
plot(x, hx, type="l", lty=2, xlab="x value",ylab="density", main="PDFs of t distributions")
for (i in 1:4){
  lines(x, dt(x,degf[i]), lwd=1, col=colors[i])
}
legend("topright", inset=.05, title="degrees of freedom",labels, lwd=1, lty=c(1, 1, 1, 1, 2), col=colors
```

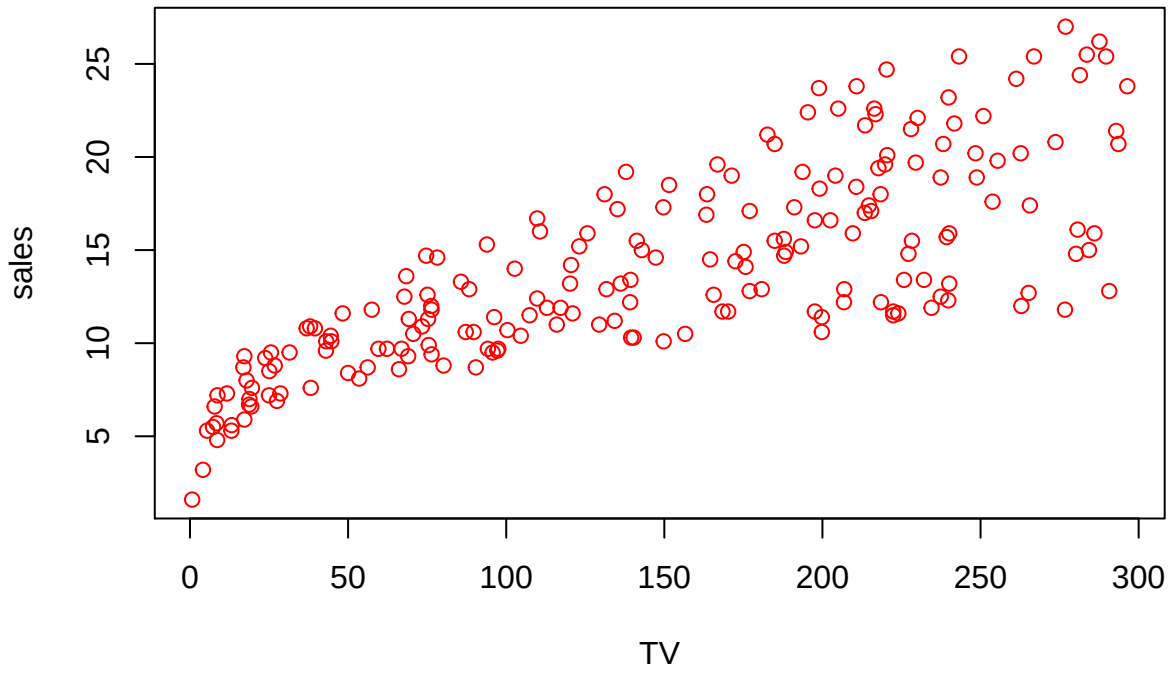



```r
x <- seq(0.01, 4, length=200)

#degree of freedom
degf1 <- c(1,3,5,1,3,5,1,3,5)
degf2 <- c(5,5,5,10,10,10,50,50,50)

colors <- c("red","red","red","blue","blue","blue","black","black","black")
labels <- c("k=1, m=5", "k=3, m=5", "k=5, m=5", "k=1, m=10", "k=3, m=10", "k=5, m=10", "k=1, m=30", "k=
plot(x, df(x,degf1[1],degf2[1]), col=colors[1],type="l", lwd=1,xlab="x value",ylab="density", main="PDFs
for (i in 2:9){
  lines(x, df(x,degf1[i],degf2[i]), lwd=1, col=colors[i])
}
legend("topright", inset=.05, title="degrees of freedom",labels, lwd=1, lty=c(1, 1, 1, 1, 1, 1, 1, 1, 1
```



#Example: Advertising (Basic Analysis)

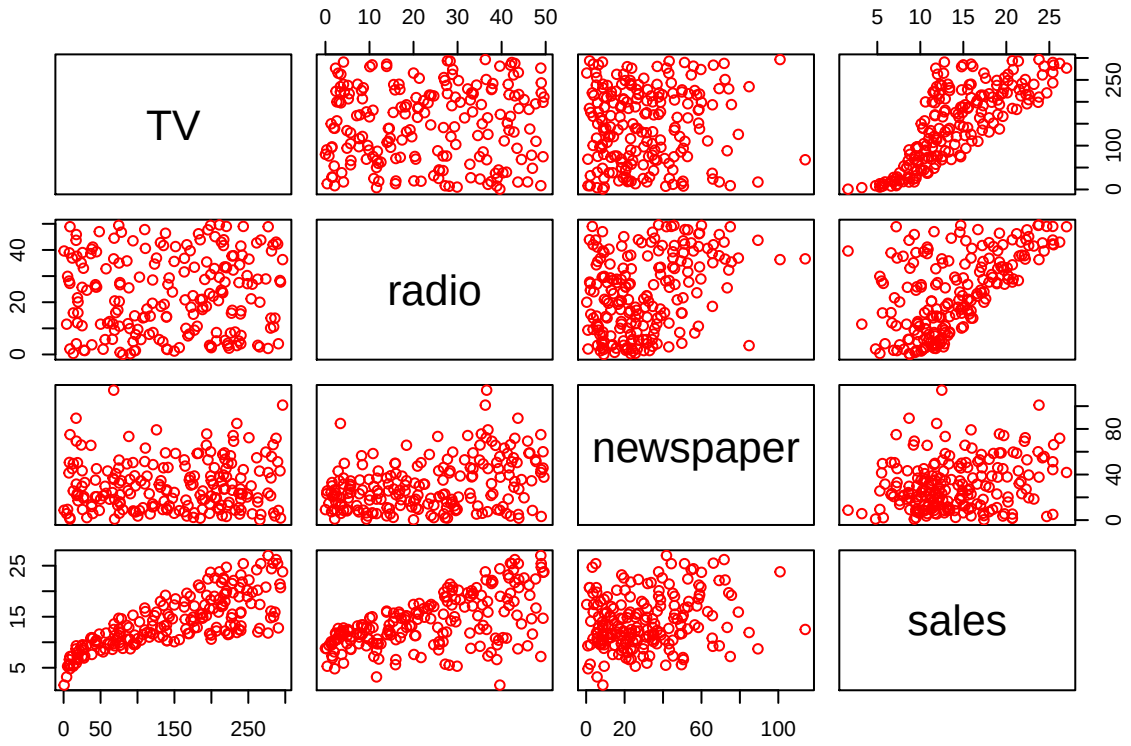
```
#read Advertising dataset
library(ISLR)
#download from http://faculty.marshall.usc.edu/gareth-james/ISL/data.html/
adv = read.csv("Advertising.csv",header=T,na.string=",")

#single scatterplot TV vs. sales
plot(adv$TV,adv$sales,xlab="TV",ylab="sales",col="red")
```



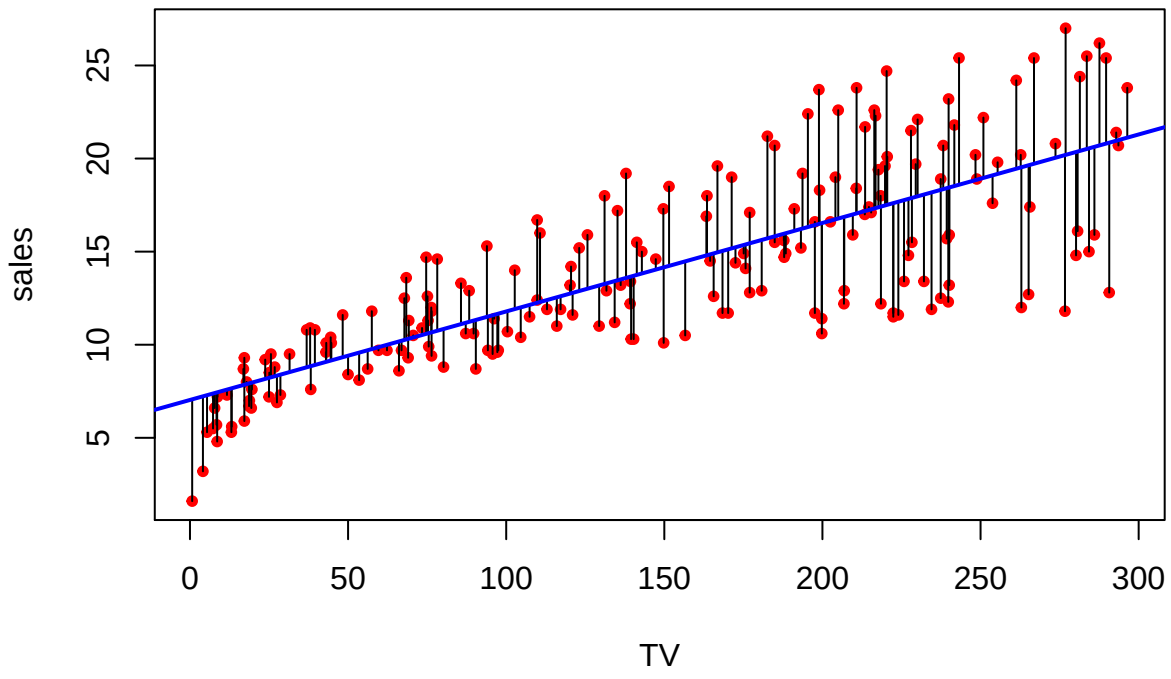

```
#correlation between variables
cor(adv[,2:4])
```

```
##                    TV      radio  newspaper
## TV         1.00000000 0.05480866 0.05664787
## radio      0.05480866 1.00000000 0.35410375
## newspaper 0.05664787 0.35410375 1.00000000
```

```
#scatter matrix of paired variables
plot(adv[c(2,3,4,5)], col="red")
```




```
#simple linear function
lm1<-lm(adv$sales~adv$TV) # fit linear models
sales_Predict<-predict(lm1)
#scatterplot TV vs. sales
plot(adv$TV,adv$sales,xlab="TV",ylab="sales",col="red",pch=20)
segments(adv$TV, adv$sales, adv$TV, sales_Predict)
abline(lm1,col="blue",lwd=2)
```



```r
#multiple linear regression
lm2<-lm(adv$sales~adv$TV+adv$radio+adv$newspaper)
summary(lm2)
```

```
##
## Call:
## lm(formula = adv$sales ~ adv$TV + adv$radio + adv$newspaper)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -8.8277 -0.8908  0.2418  1.1893  2.8292
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)    2.938889   0.311908   9.422   <2e-16 ***
## adv$TV         0.045765   0.001395  32.809   <2e-16 ***
## adv$radio      0.188530   0.008611  21.893   <2e-16 ***
## adv$newspaper -0.001037   0.005871  -0.177     0.86
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.686 on 196 degrees of freedom
## Multiple R-squared:  0.8972, Adjusted R-squared:  0.8956
## F-statistic: 570.3 on 3 and 196 DF,  p-value: < 2.2e-16
```

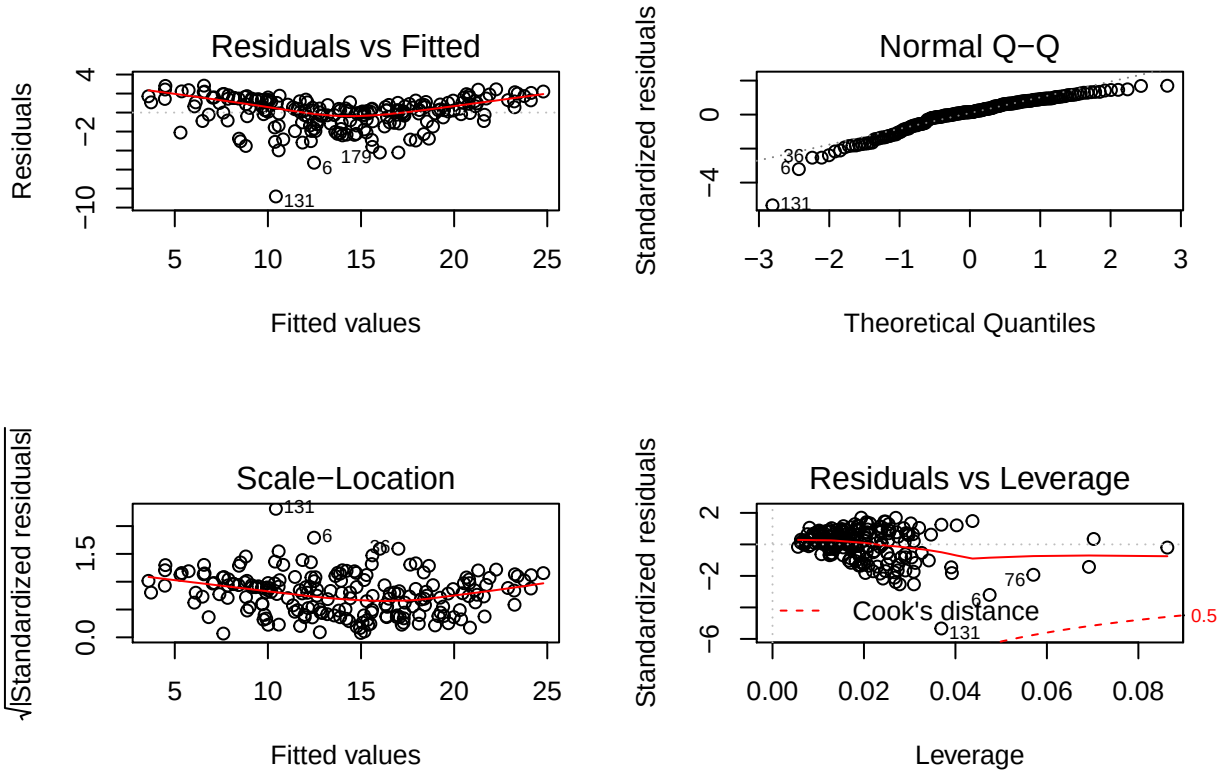
```r
coef(lm2) # a generic function which extracts model coefficients from objects returned by modeling func
```

```
##   (Intercept)        adv$TV     adv$radio adv$newspaper
##   2.938889369   0.045764645   0.188530017  -0.001037493
```

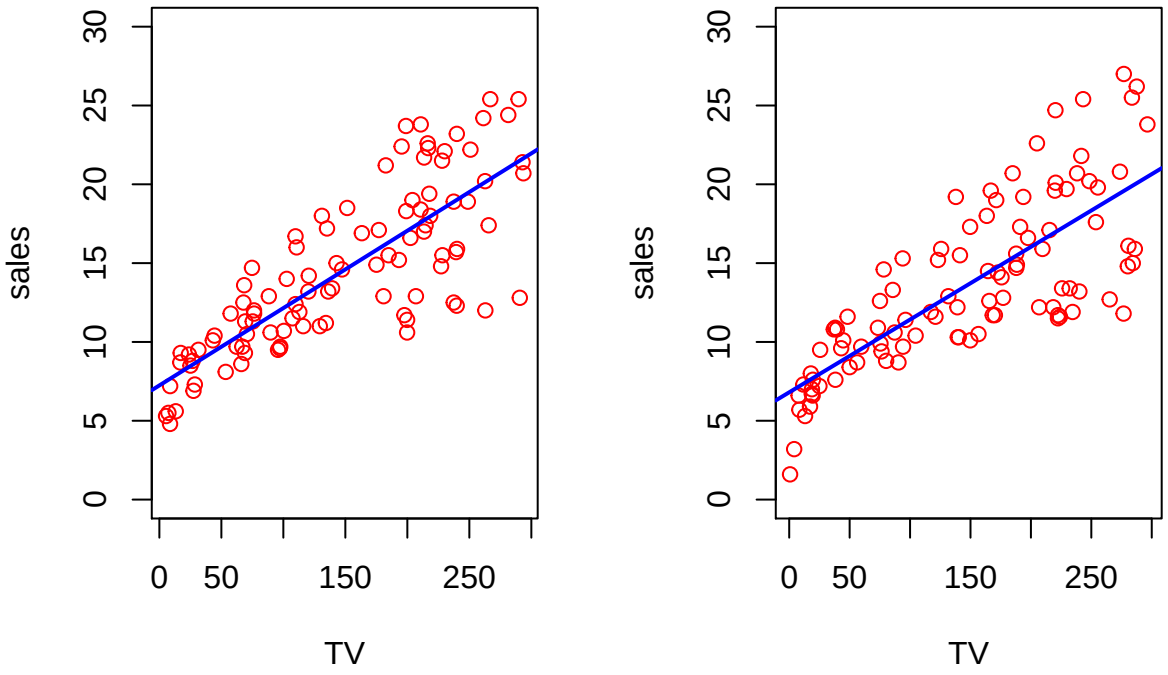
```r
confint(lm2) # confidence Intervals For Model Parameters
```

```
##                     2.5 %     97.5 %
## (Intercept)    2.32376228 3.55401646
## adv$TV         0.04301371 0.04851558
## adv$radio      0.17154745 0.20551259
## adv$newspaper -0.01261595 0.01054097
```

```r
par(mfrow=c(2,2))
plot(lm2)
```

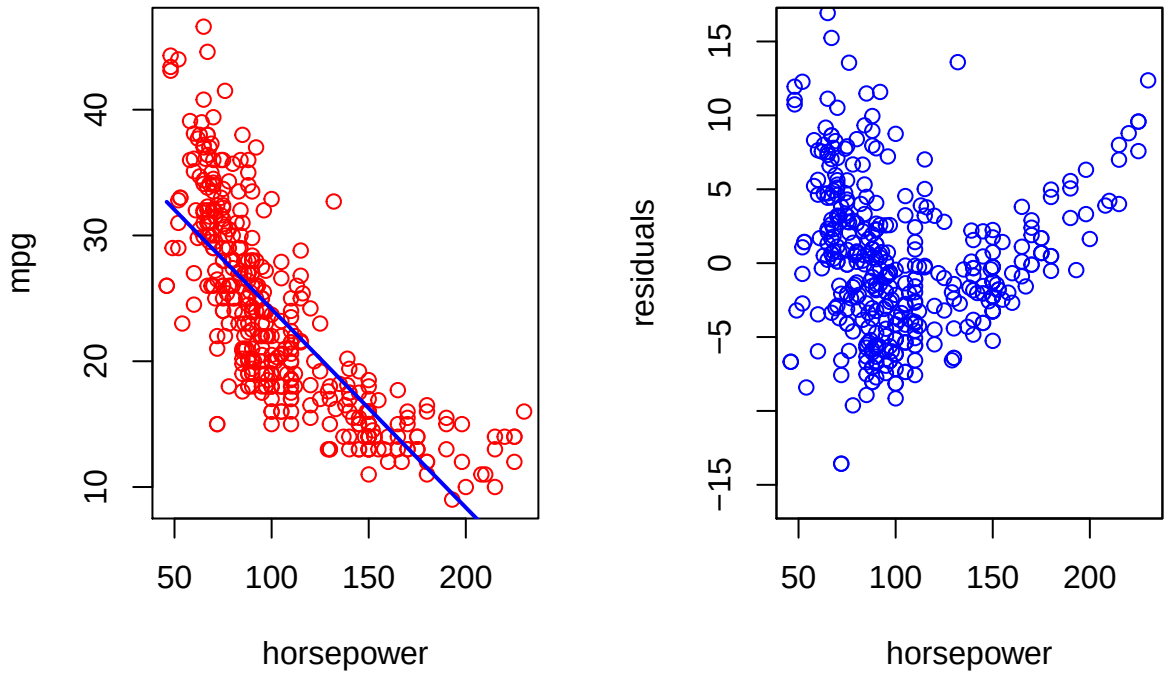



```
#the first 100 data
lm1a<-lm(adv$sales[1:100]~adv$TV[1:100])
#summary(lm1a)

#the following 100 data
lm1b<-lm(adv$sales[101:200]~adv$TV[101:200])
#summary(lm1b)
par(mfrow=c(1,2))
plot(adv$TV[1:100],adv$sales[1:100],xlab="TV",ylab="sales",col="red",ylim=c(0,30))
abline(lm(adv$sales[1:100]~adv$TV[1:100]),col="blue",lwd=2)
plot(adv$TV[101:200],adv$sales[101:200],xlab="TV",ylab="sales",col="red",ylim=c(0,30))
abline(lm(adv$sales[101:200]~adv$TV[101:200]),col="blue",lwd=2)
```

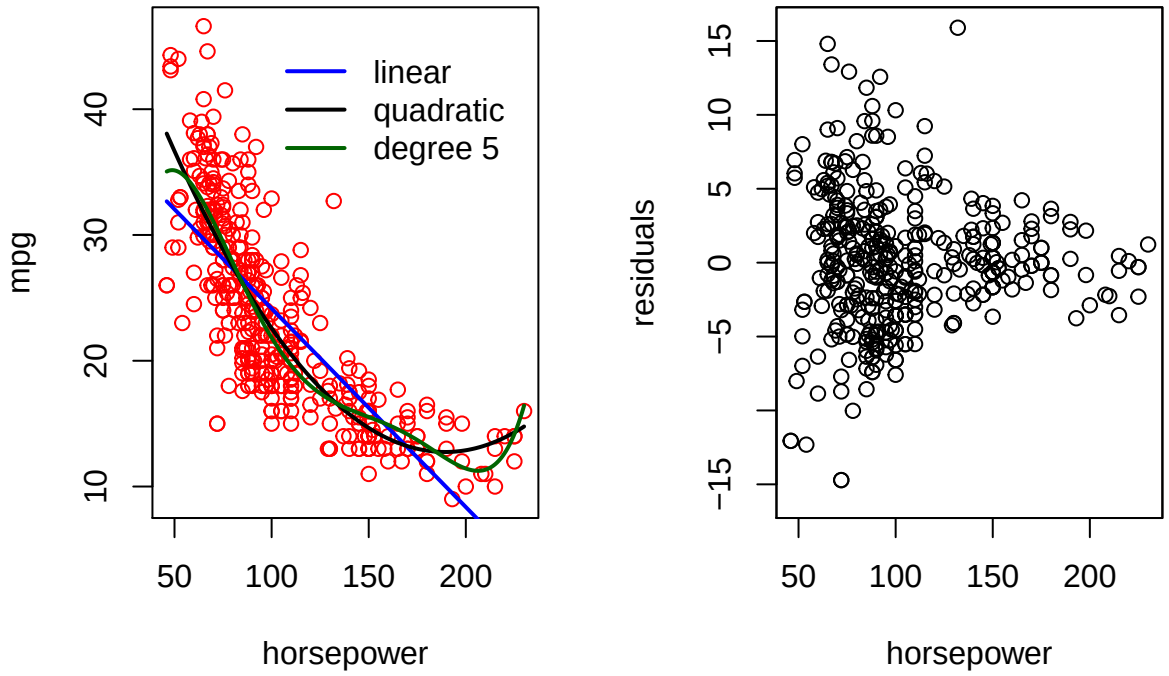


#Example: Auto (Basic Analysis)

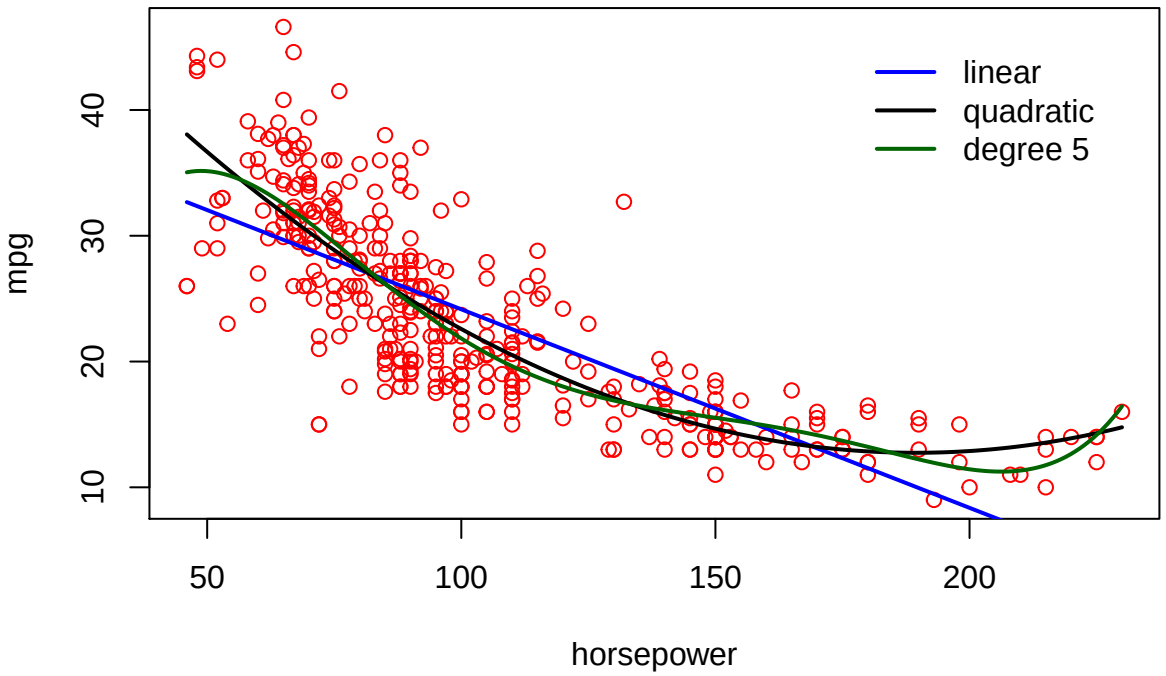
```
auto = Auto
auto$horsepower = as.integer(auto$horsepower)
lm7a <- lm(mpg~horsepower,data=auto)
lm7b <- lm(mpg~horsepower+I(horsepower^2),data=auto)
lm7c <- lm(mpg~horsepower+I(horsepower^2)+I(horsepower^3)+I(horsepower^4)+I(horsepower^5),data=auto)
par(mfrow=c(1,2))
plot(auto$horsepower,auto$mpg,col="red",xlab = "horsepower",ylab = "mpg")
new <- data.frame(horsepower = seq(min(auto$horsepower,na.rm=TRUE),max(auto$horsepower,na.rm=TRUE),leng
lines(new$horsepower,predict(lm7a,newdata=new),col="blue",lwd=2)
plot(na.omit(auto$horsepower),lm7a$residuals,col="blue",xlab = "horsepower", ylab = "residuals",ylim=c(
box()
```




```
par(mfrow=c(1,2))
plot(auto$horsepower,auto$mpg,col="red",xlab = "horsepower",ylab = "mpg")
lines(new$horsepower,predict(lm7a,newdata=new),col="blue",lwd=2)
lines(new$horsepower,predict(lm7b,newdata=new),col="black",lwd=2)
lines(new$horsepower,predict(lm7c,newdata=new),col="darkgreen",lwd=2)
legend("topright", inset=.05,c("linear","quadratic","degree 5"), lwd=2, lty=c(1, 1, 1), col=c("blue","bl
plot(na.omit(auto$horsepower),lm7b$residuals,col="black",xlab = "horsepower", ylab = "residuals",ylim=c
box()
```

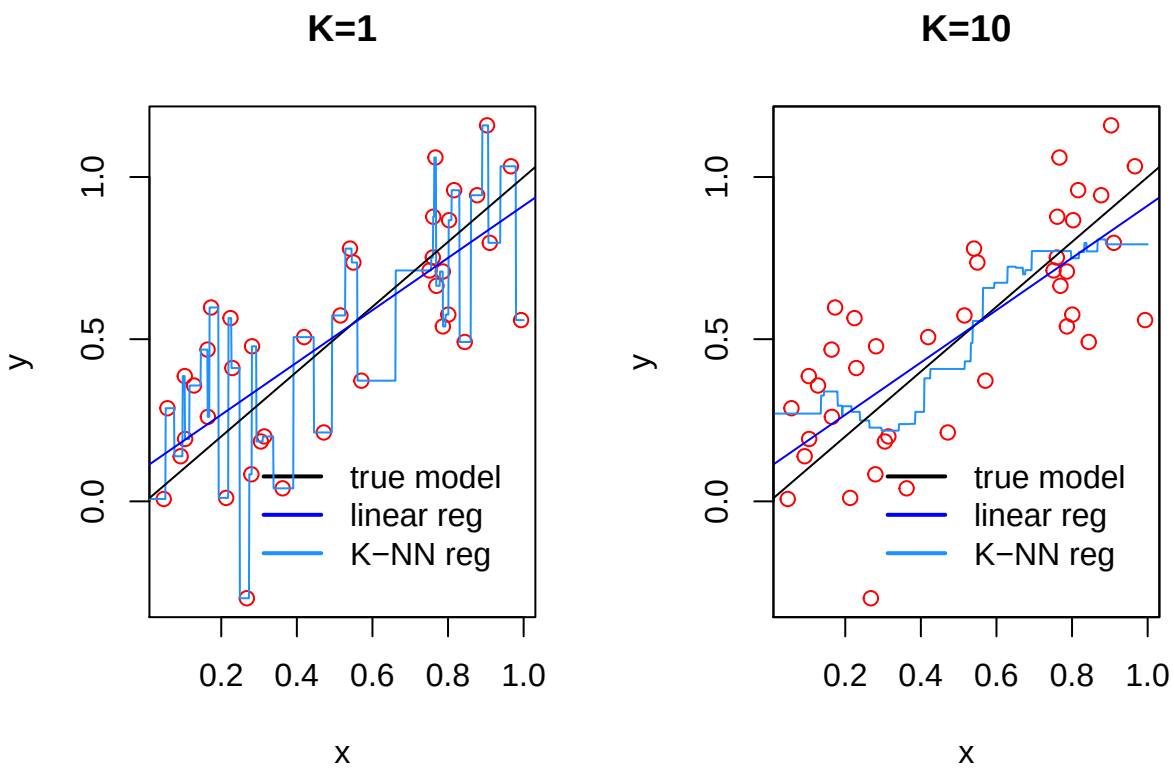


```r
plot(auto$horsepower,auto$mpg,col="red",xlab = "horsepower",ylab = "mpg")
lines(new$horsepower,predict(lm7a,newdata=new),col="blue",lwd=2)
lines(new$horsepower,predict(lm7b,newdata=new),col="black",lwd=2)
lines(new$horsepower,predict(lm7c,newdata=new),col="darkgreen",lwd=2)
legend("topright", inset=.05,c("linear","quadratic","degree 5"), lwd=2, lty=c(1, 1, 1), col=c("blue","bl
```



KNN

```r
# install.packages('FNN')
library('FNN')
par(mfrow=c(1,2))
x<-seq(0,1,length=1000)
t2.data<-data.frame(x)
x<-runif(40)
y<-x+rnorm(40,0,0.25)
t1.data<-data.frame(x,y)
knn11<-knn.reg(train=t1.data$x,test=t2.data,t1.data$y,k=1)
knn110<-knn.reg(train=t1.data$x,test=t2.data,t1.data$y,k=10)
plot(x,y,col="red",main = "K=1")
legend("bottomright", inset=.05,c("true model","linear reg","K-NN reg"), lwd=2, lty=c(1, 1, 1), col=c("
lines(t2.data$x,knn11$pred,col="dodgerblue")
abline(0,1)
abline(lm(y~x),col="blue")
plot(x,y,col="red",main = "K=10")
legend("bottomright", inset=.05,c("true model","linear reg","K-NN reg"), lwd=2, lty=c(1, 1, 1), col=c("
lines(t2.data$x,knn110$pred,col="dodgerblue")
abline(0,1)
abline(lm(y~x),col="blue")
box()
```


K=1
K=10
y
x
0.0
0.5
1.0
0.2
0.4
0.6
0.8
1.0
true model
linear reg
K−NN reg