R Tutorial

August 4, 2019

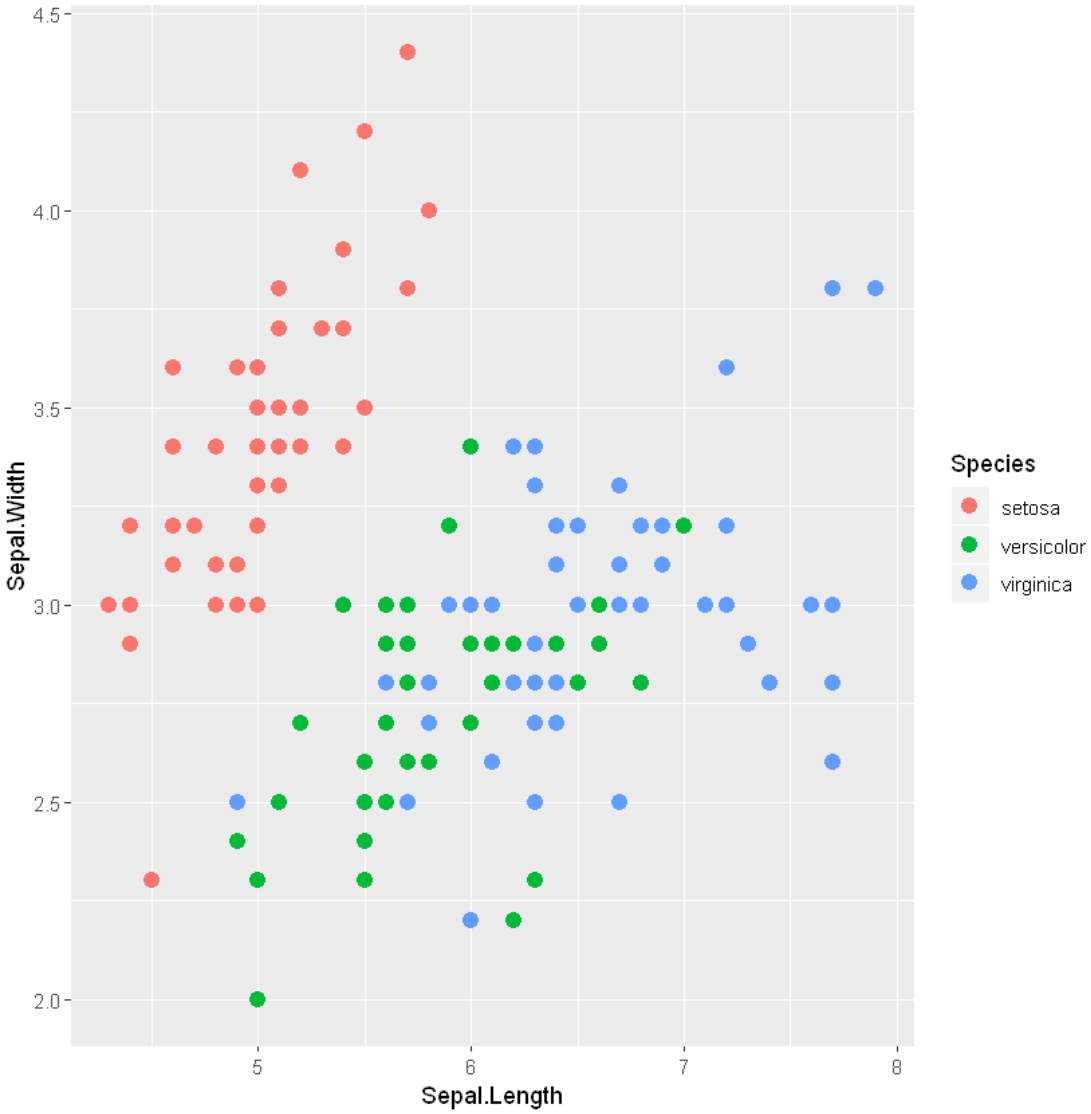
Contents of this tutorial are partly taken from the course created by Barton Poulson titled as "R Programming Tutorial - Learn the Basics of Statistical Computing". The link to the youtube video is https://www.youtube.com/watch?v=_V8eKsto3Ug. New contents are added to further improve the coverage of materials.

[3]:
```
library(ggplot2)
```

[22]:
```
head(iris, n = 10L)
```

Sepal.Length	Sepal.Width	Petal.Length	Petal.Width	Species
5.1	3.5	1.4	0.2	setosa
4.9	3.0	1.4	0.2	setosa
4.7	3.2	1.3	0.2	setosa
4.6	3.1	1.5	0.2	setosa
5.0	3.6	1.4	0.2	setosa
5.4	3.9	1.7	0.4	setosa
4.6	3.4	1.4	0.3	setosa
5.0	3.4	1.5	0.2	setosa
4.4	2.9	1.4	0.2	setosa
4.9	3.1	1.5	0.1	setosa

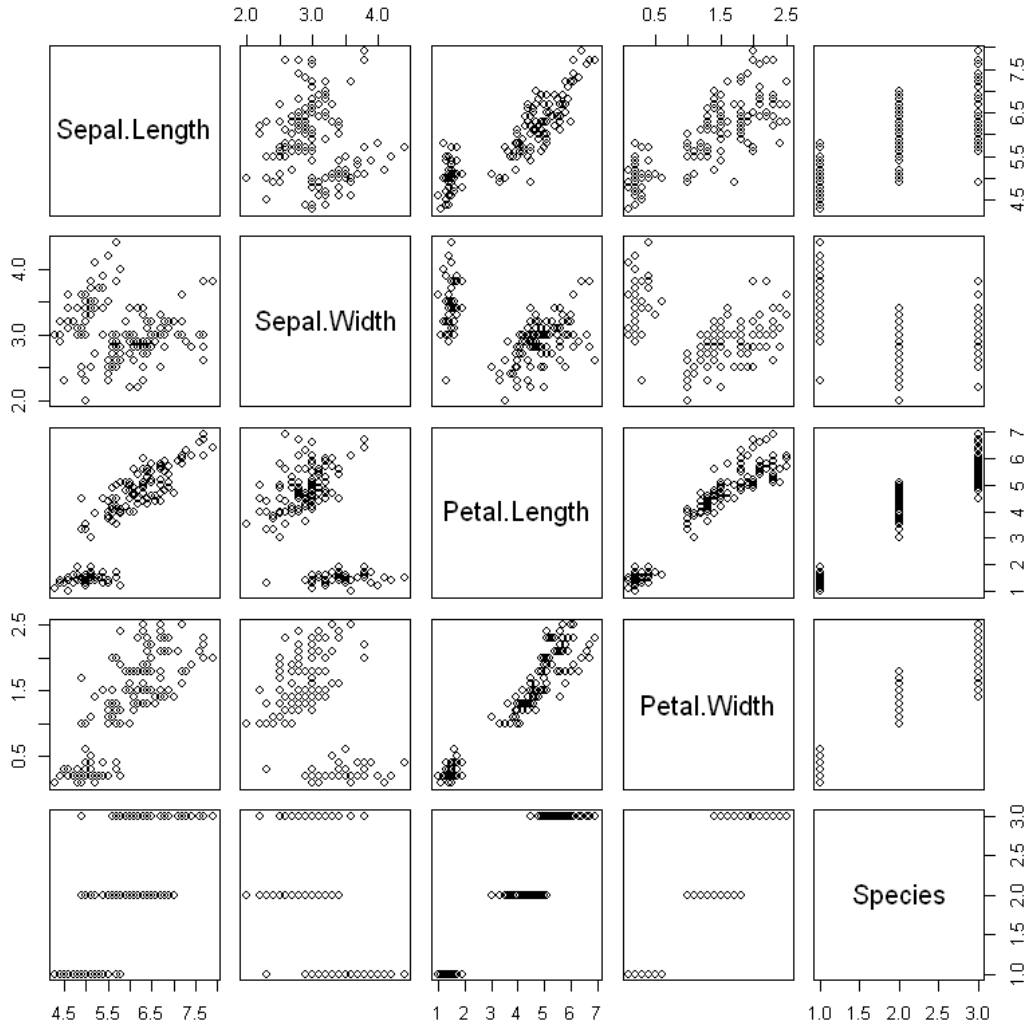
[23]:
```
ggplot(data=iris, aes(x=Sepal.Length, y=Sepal.Width, color=Species))+
geom_point(size=3)
```



[26]:
```
summary(iris)
plot(iris)
```

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

```
 virginica :50
```

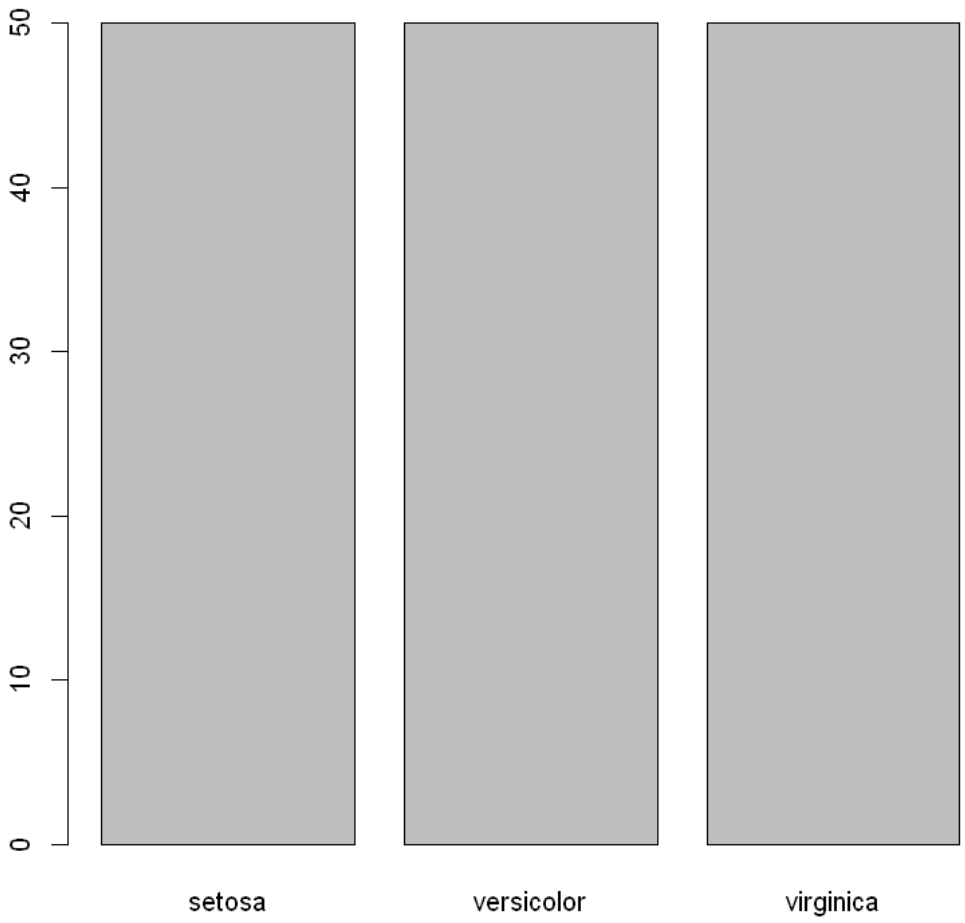


0.1 Packages

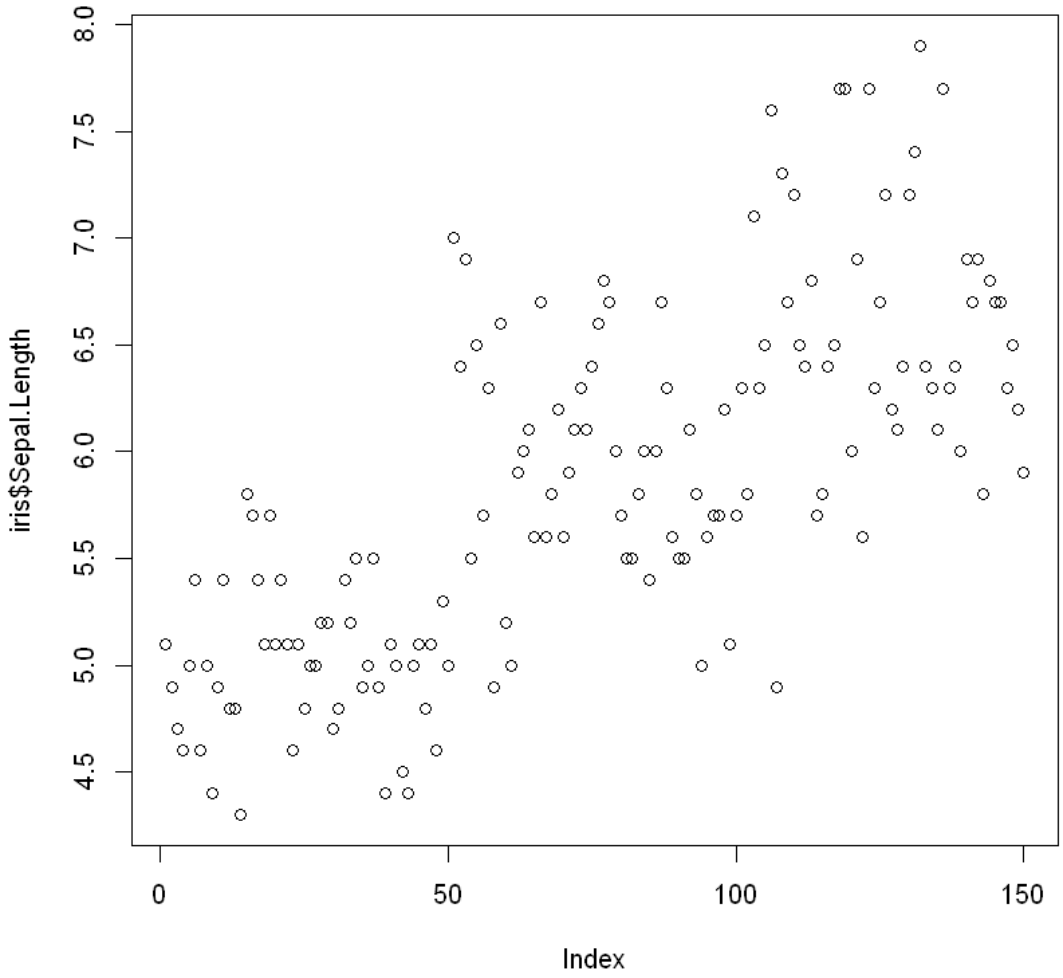
dyplyr - manipulating data frames tydyr - cleaning up information stringr - working with string lubridate - manipulating date information httr - working with website data ggvis - grammar for graphics/ interactive visualization ggplot2 - plotting data shiny - interactive web applications rio - R input output rmarkdown - interactive or rich notebooks

One package to load all the files: pacman

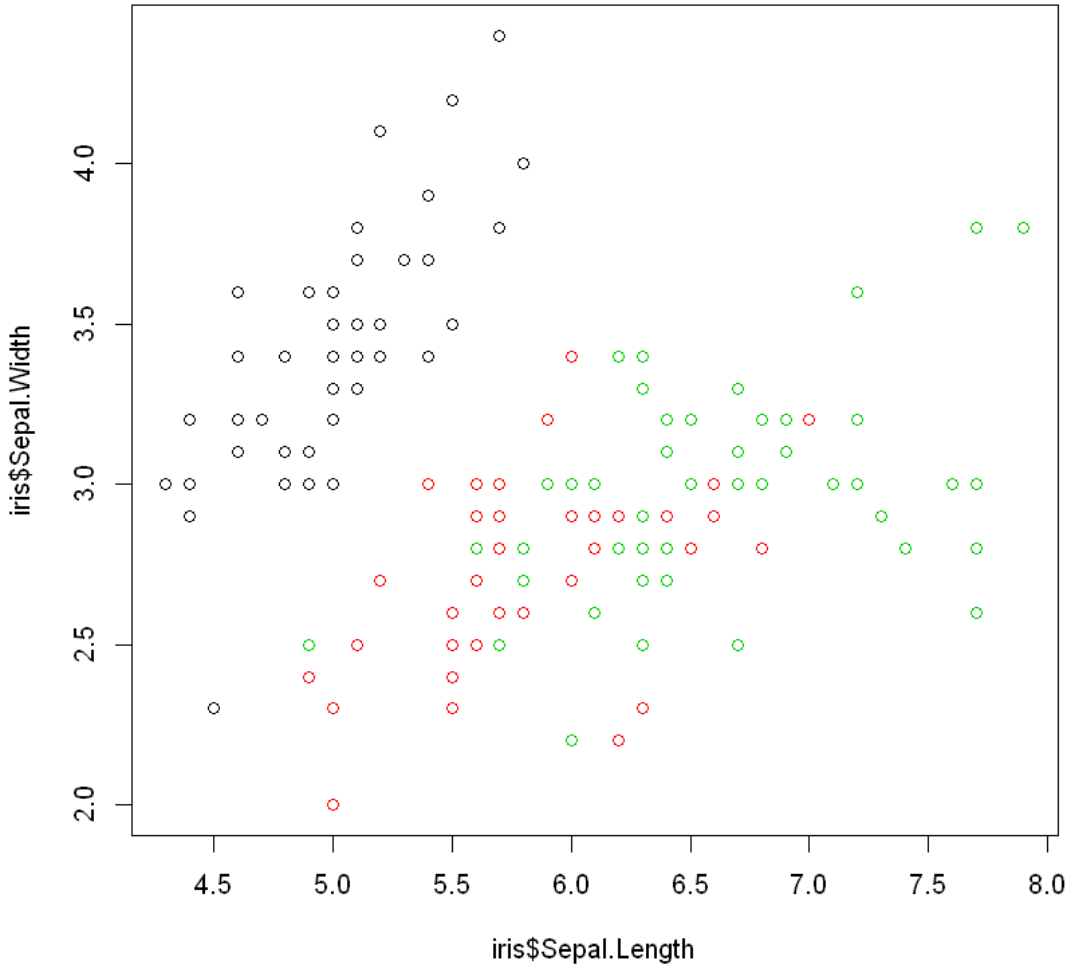
[30]:
```
library(dplyr)
plot(iris$Species)
```



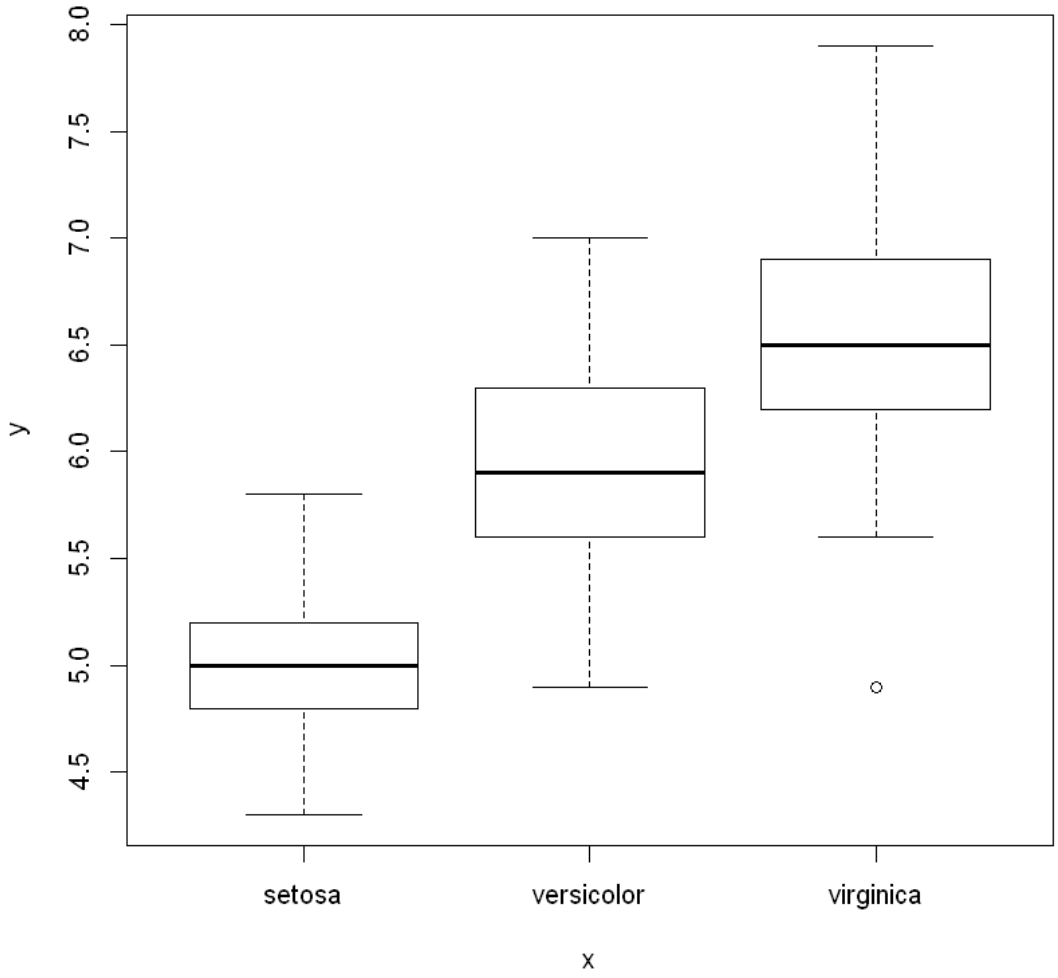
[31]:
```
plot(iris$Sepal.Length)
```



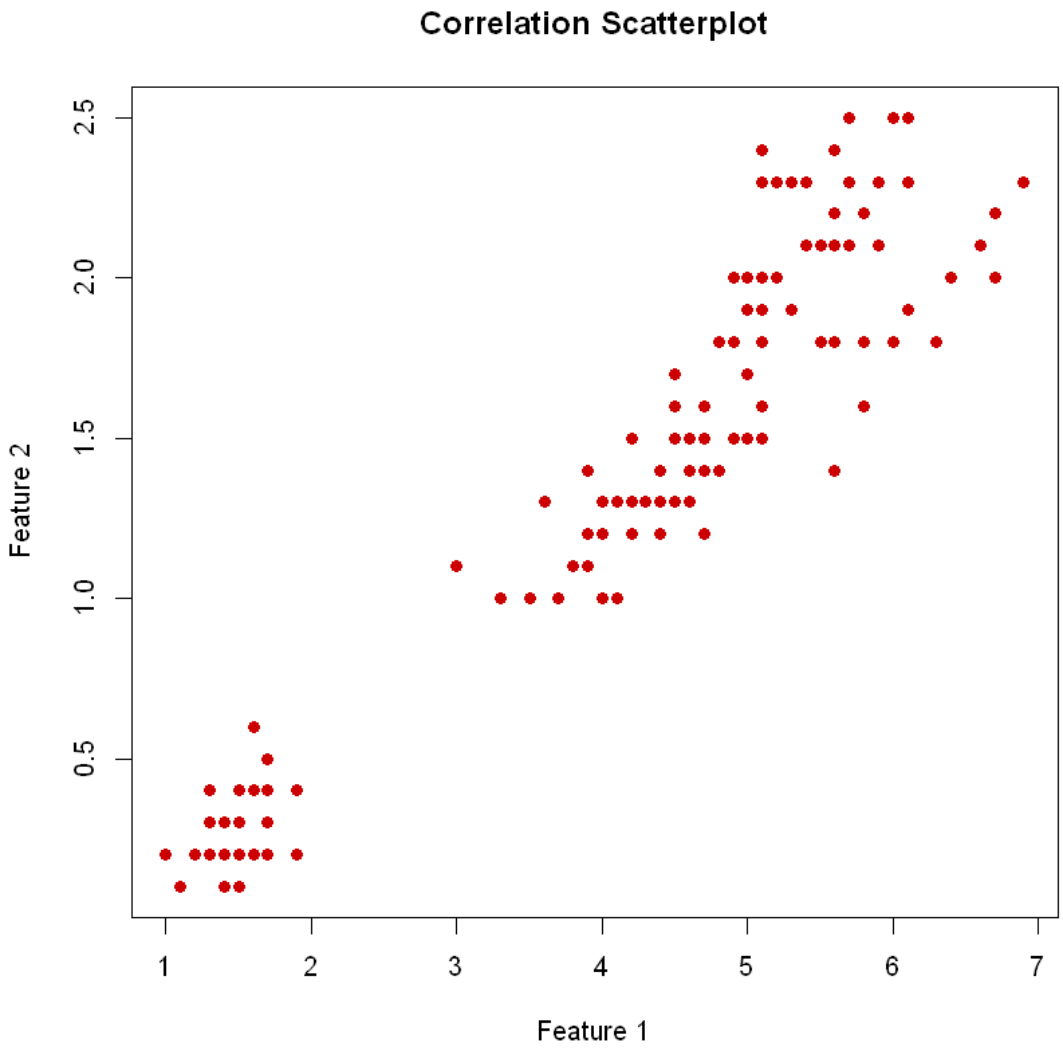
[35]:
```
plot(iris$Sepal.Length, iris$Sepal.Width, col=iris$Species)
```



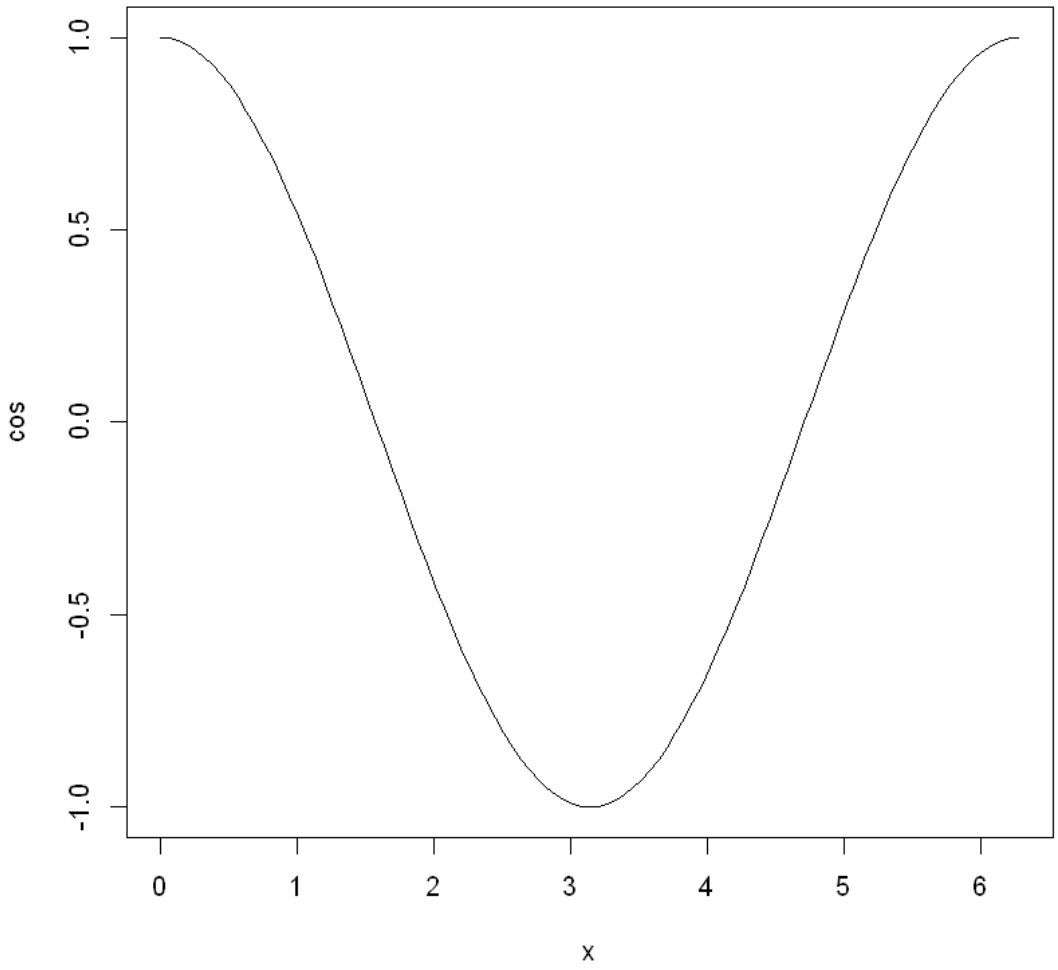
[37]:
```
plot(iris$Species, iris$Sepal.Length)
```

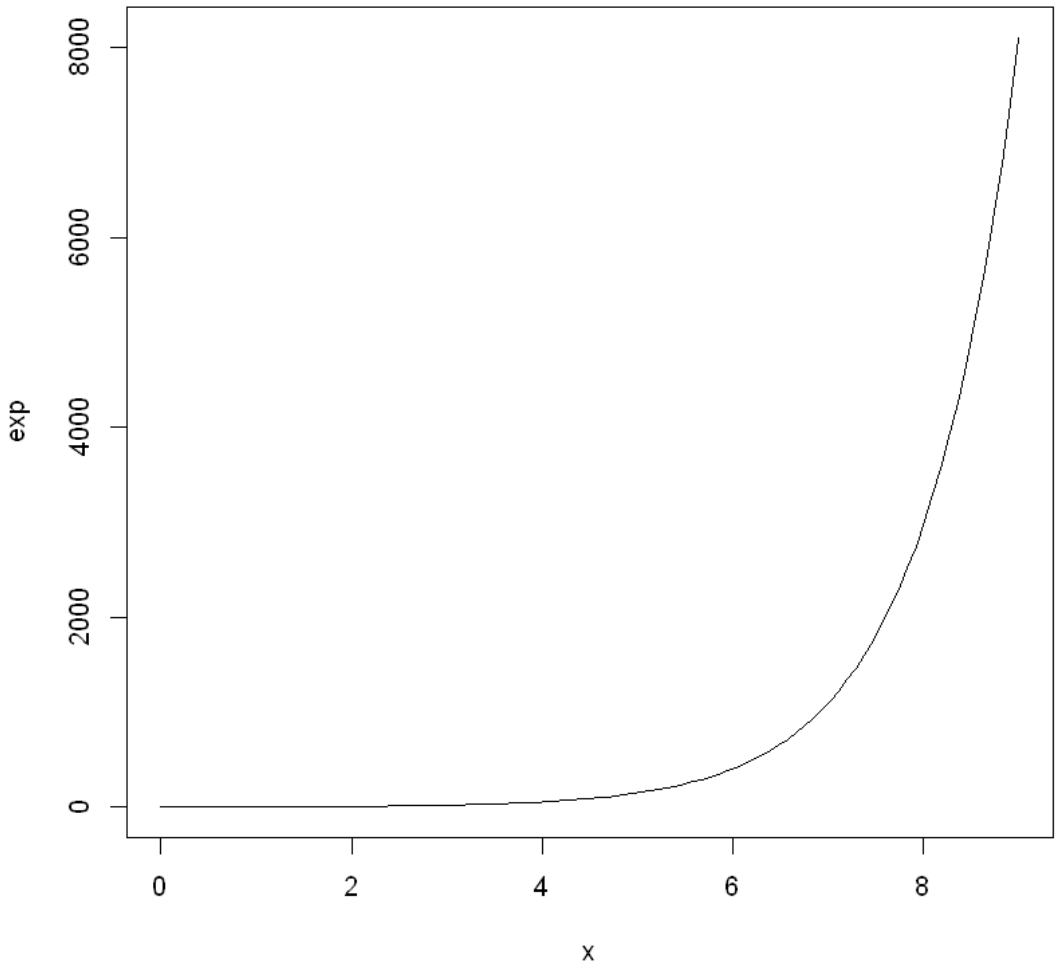


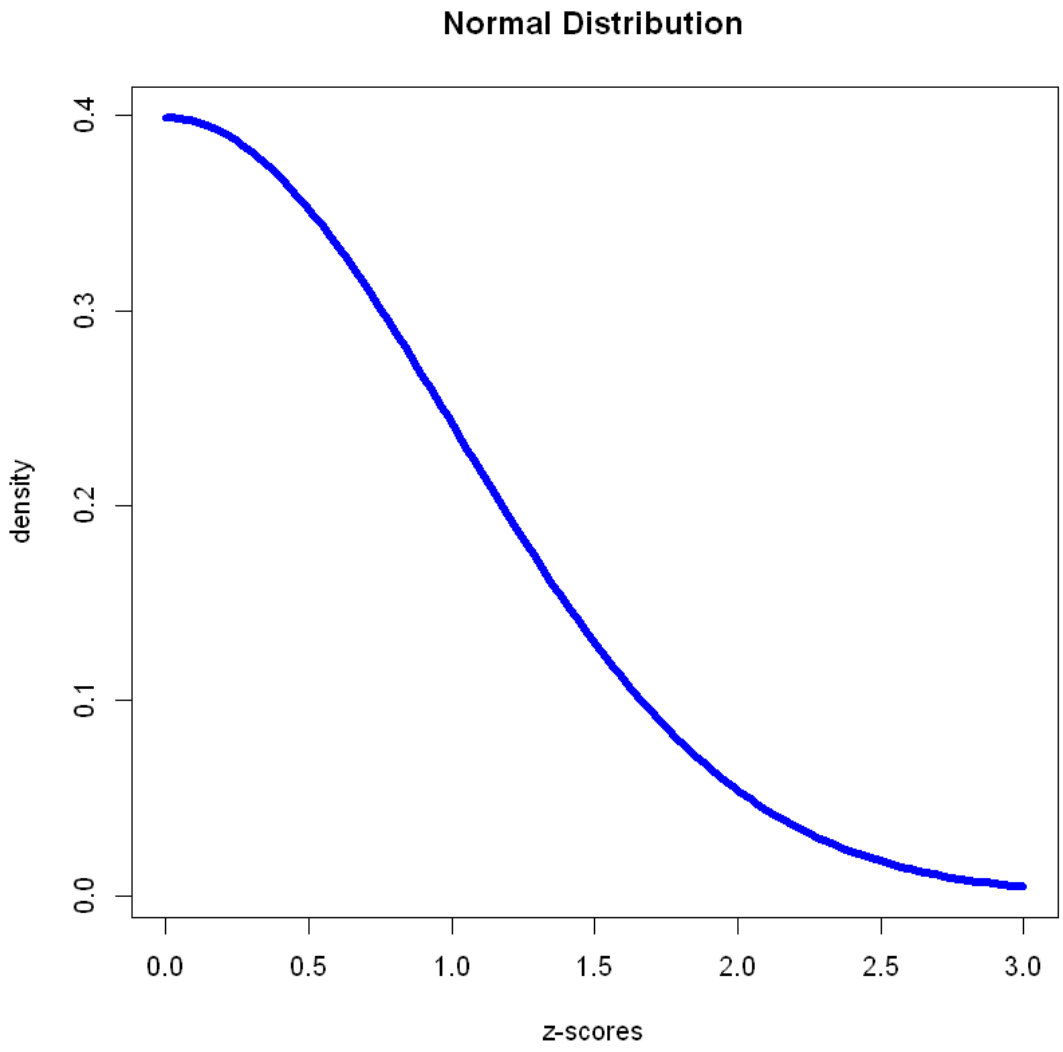
[44]:
```
plot(iris$Petal.Length, iris$Petal.Width, col="#CC0000", pch=19,␣
↪main="Correlation Scatterplot", xlab="Feature 1", ylab="Feature 2")
```



[46]:
```
plot(cos, 0, 2*pi)
plot(exp, 0, 9)
plot(dnorm, 0, 3, col='blue', lwd=5, main='Normal Distribution',␣
 ↪xlab='z-scores', ylab='density')
```


cos
1.0
0.5
0.0
-0.5
-1.0
0
1
2
3
4
5
6
x

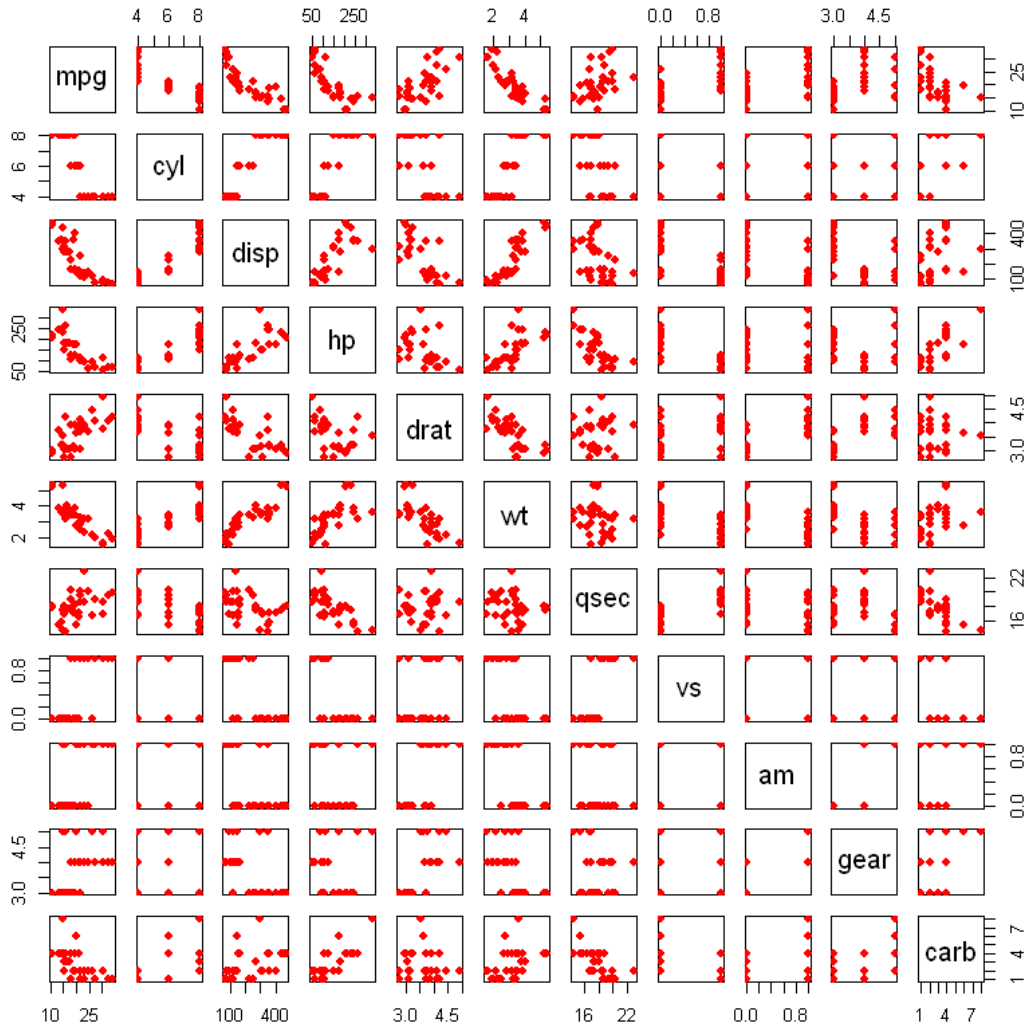

exp
8000
6000
4000
2000
0
0
2
4
6
8
x



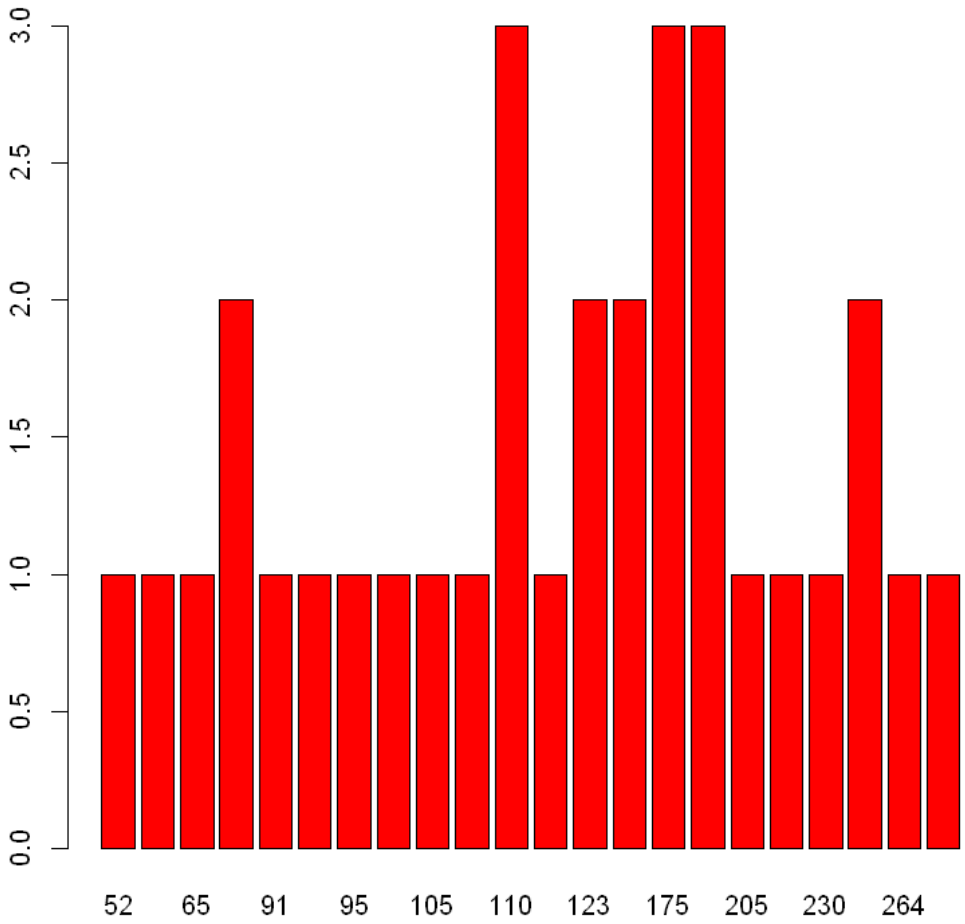
[47]:
```
head(mtcars)
```

	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

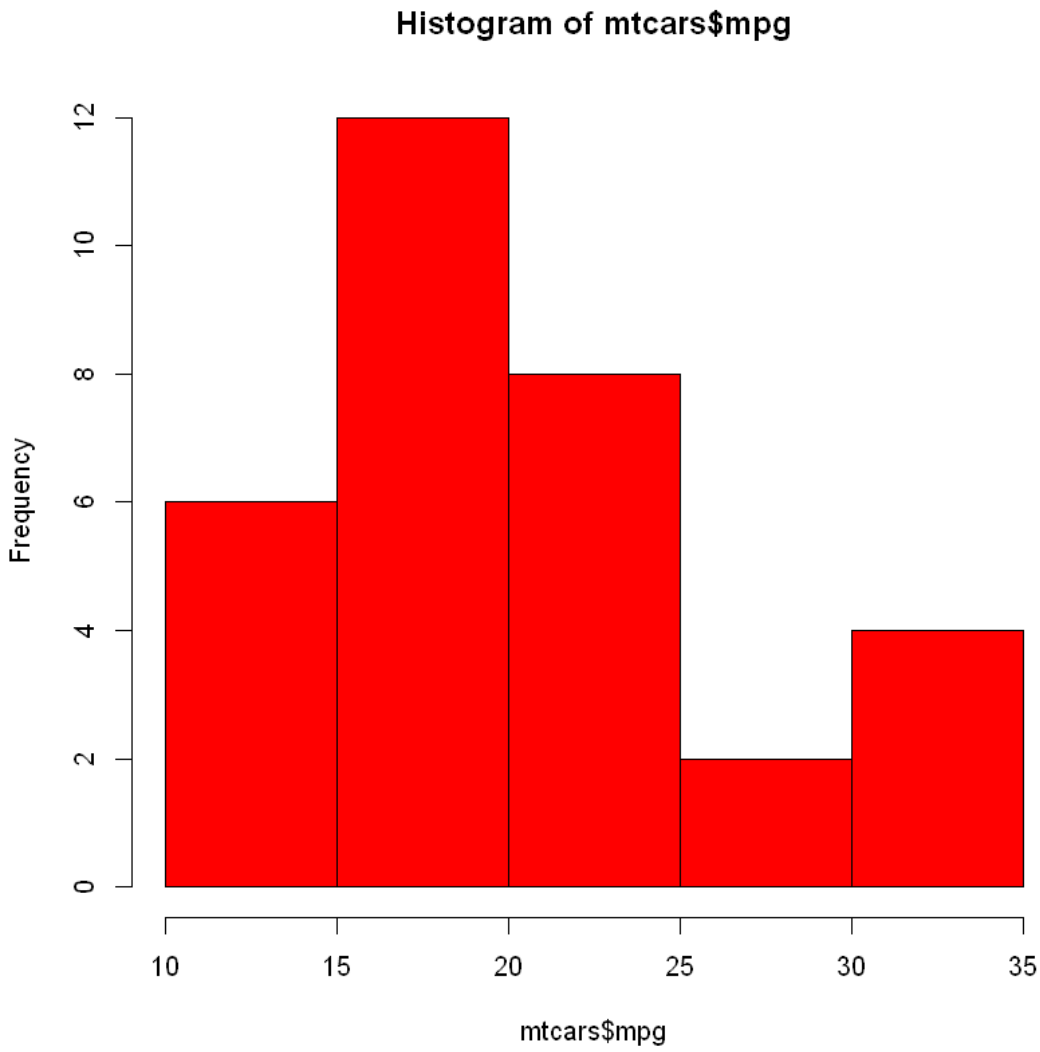
[52]:
```
plot(mtcars, pch=19, col='red')
```

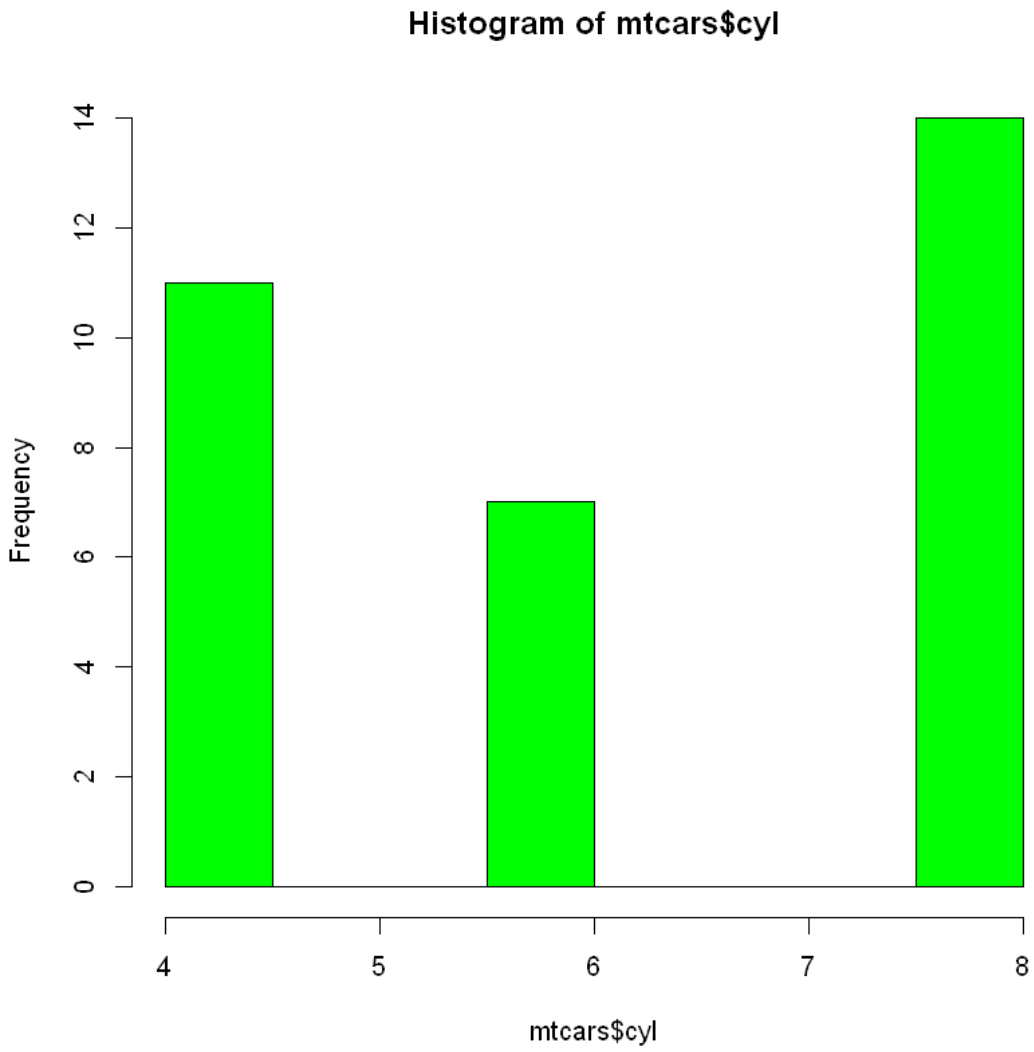


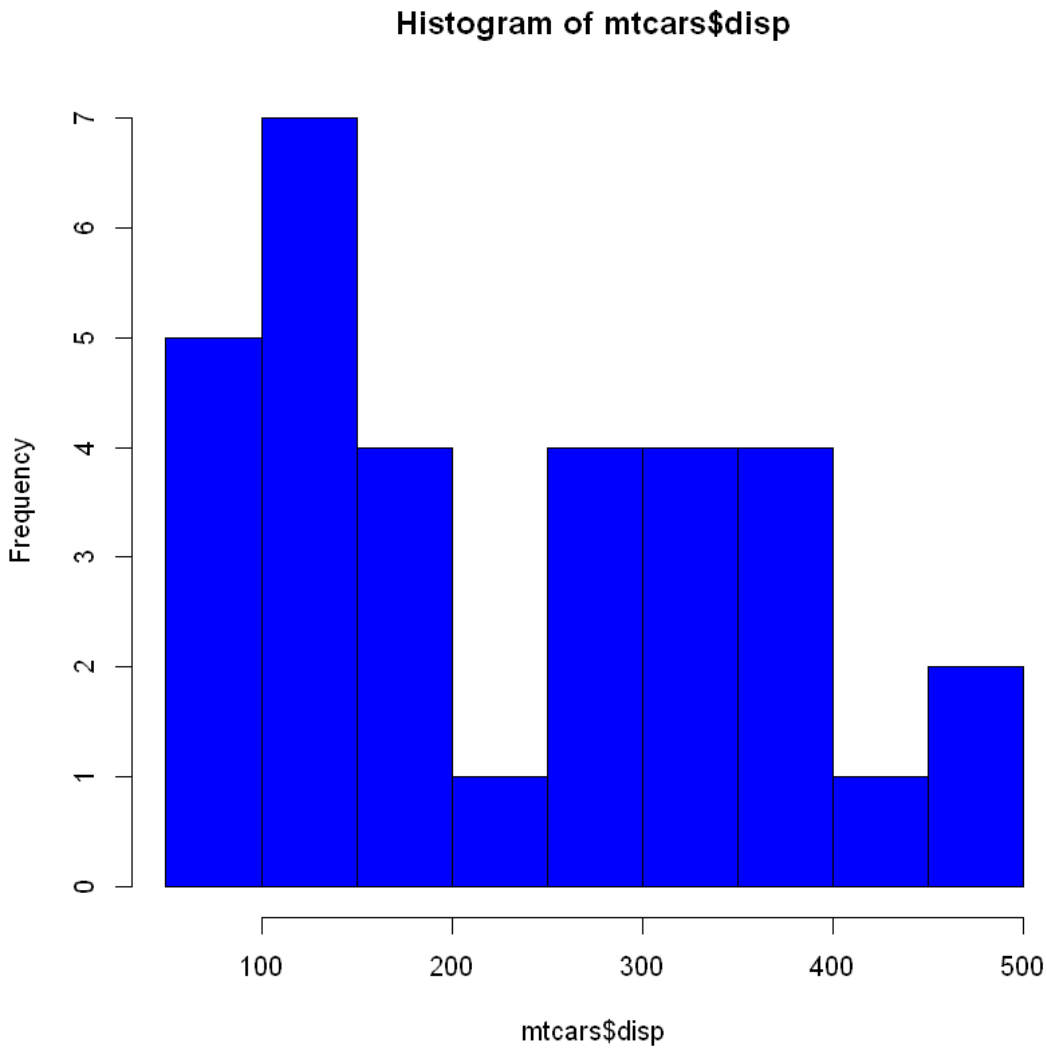
[60]:
```
hp <- table(mtcars$hp)
barplot(hp, col='red')
```

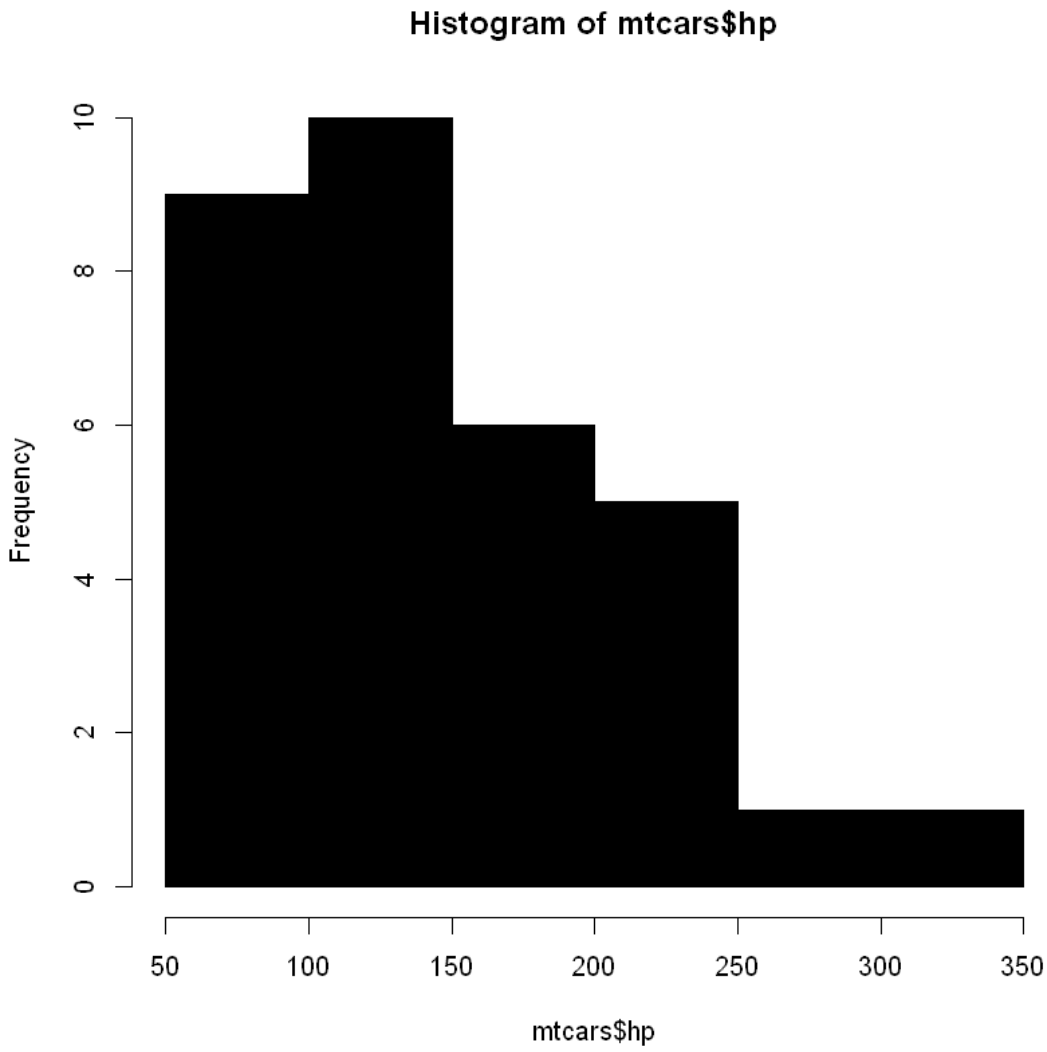


[70]:
```
hist(mtcars$mpg, col='red')
hist(mtcars$cyl, col='green')
hist(mtcars$disp, col='blue')
hist(mtcars$hp, col='black')
```

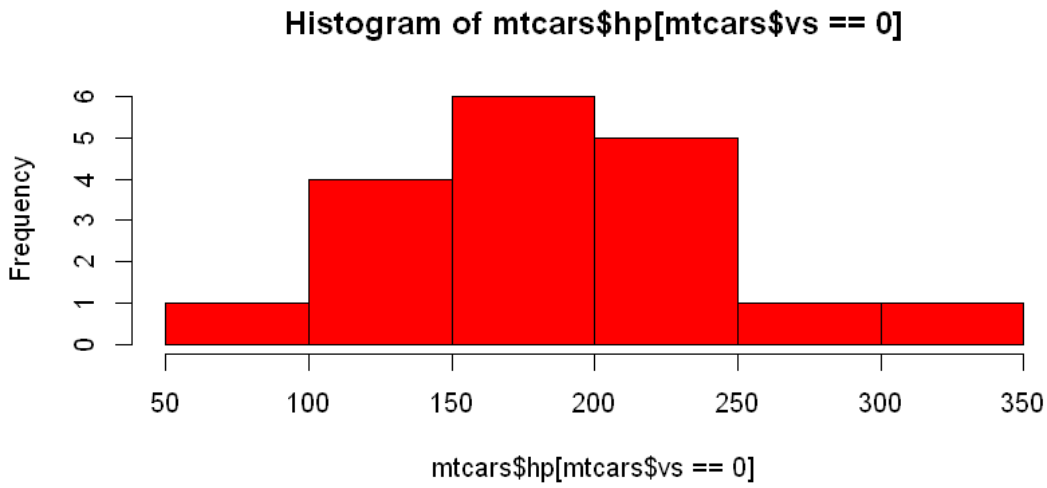

Histogram of mtcars$mpg
Frequency
0
2
4
6
8
10
12
10
15
20
25
30
35
mtcars$mpg

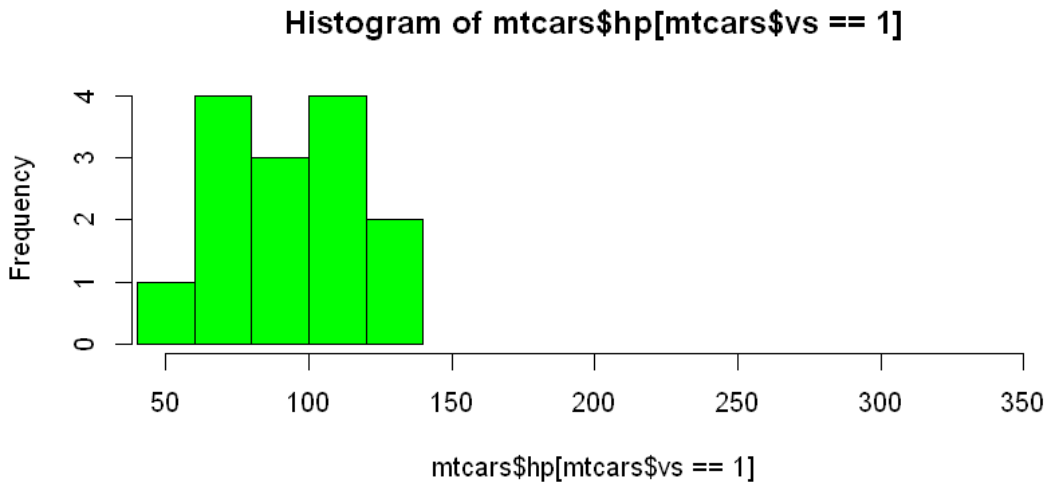

Histogram of mtcars$cyl
Frequency
0
2
4
6
8
10
12
14
4
5
6
7
8
mtcars$cyl


Histogram of mtcars$disp
Frequency
0
1
2
3
4
5
6
7
100
200
300
400
500
mtcars$disp

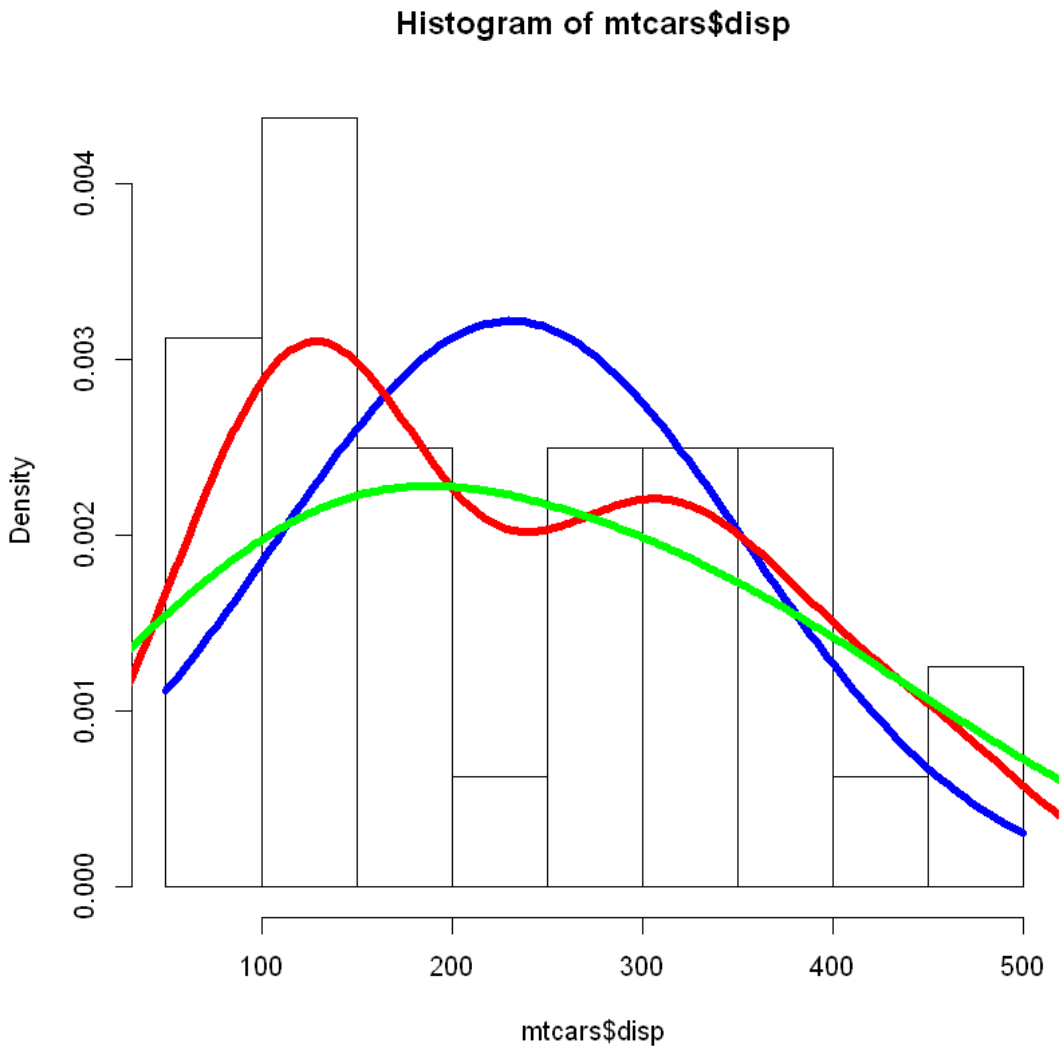


[76]:
```
par(mfrow=c(2, 1))
hist(mtcars$hp [mtcars$vs==0], xlim=c(50, 350), col='red')
hist(mtcars$hp [mtcars$vs==1], xlim=c(50, 350), col='green')
# vs - Engine Shape (0 = V-Shaped, 1 = Straight)
```





[93]:
```
hist(mtcars$disp, freq=FALSE)
curve(dnorm(x, mean(mtcars$disp), sd(mtcars$disp)), add=TRUE, col='blue', lwd=5)
lines(density(mtcars$disp), lwd=5, col='red')
lines(density(mtcars$disp, adjust=2), lwd=5, col='green')
```

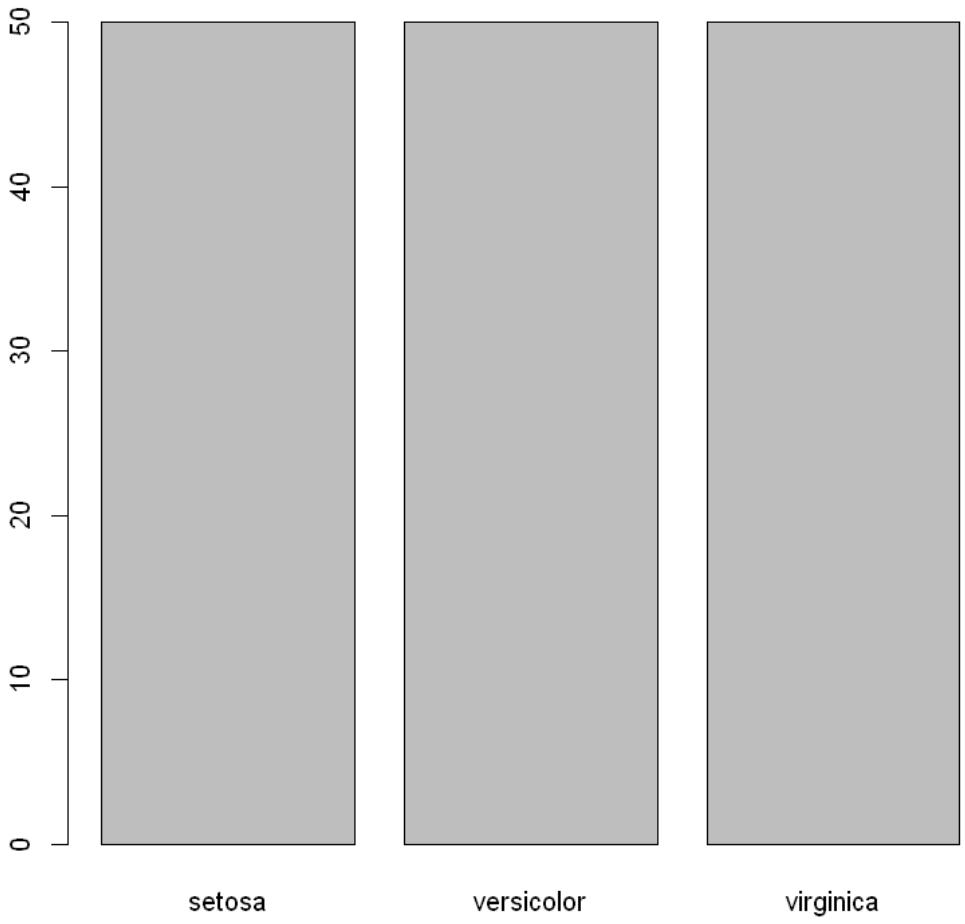


[97]:
```
summary(iris$Sepal.Width)
summary(iris)
barplot(summary(iris$Species))
```

```
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
  2.000   2.800   3.000   3.057   3.300   4.400 
```

```
  Sepal.Length    Sepal.Width     Petal.Length    Petal.Width   
 Min.   :4.300   Min.   :2.000   Min.   :1.000   Min.   :0.100  
 1st Qu.:5.100   1st Qu.:2.800   1st Qu.:1.600   1st Qu.:0.300  
 Median :5.800   Median :3.000   Median :4.350   Median :1.300  
 Mean   :5.843   Mean   :3.057   Mean   :3.758   Mean   :1.199  
 3rd Qu.:6.400   3rd Qu.:3.300   3rd Qu.:5.100   3rd Qu.:1.800  
```

```
Max.   :7.900   Max.   :4.400   Max.   :6.900   Max.   :2.500
      Species
setosa    :50
versicolor:50
virginica :50
```



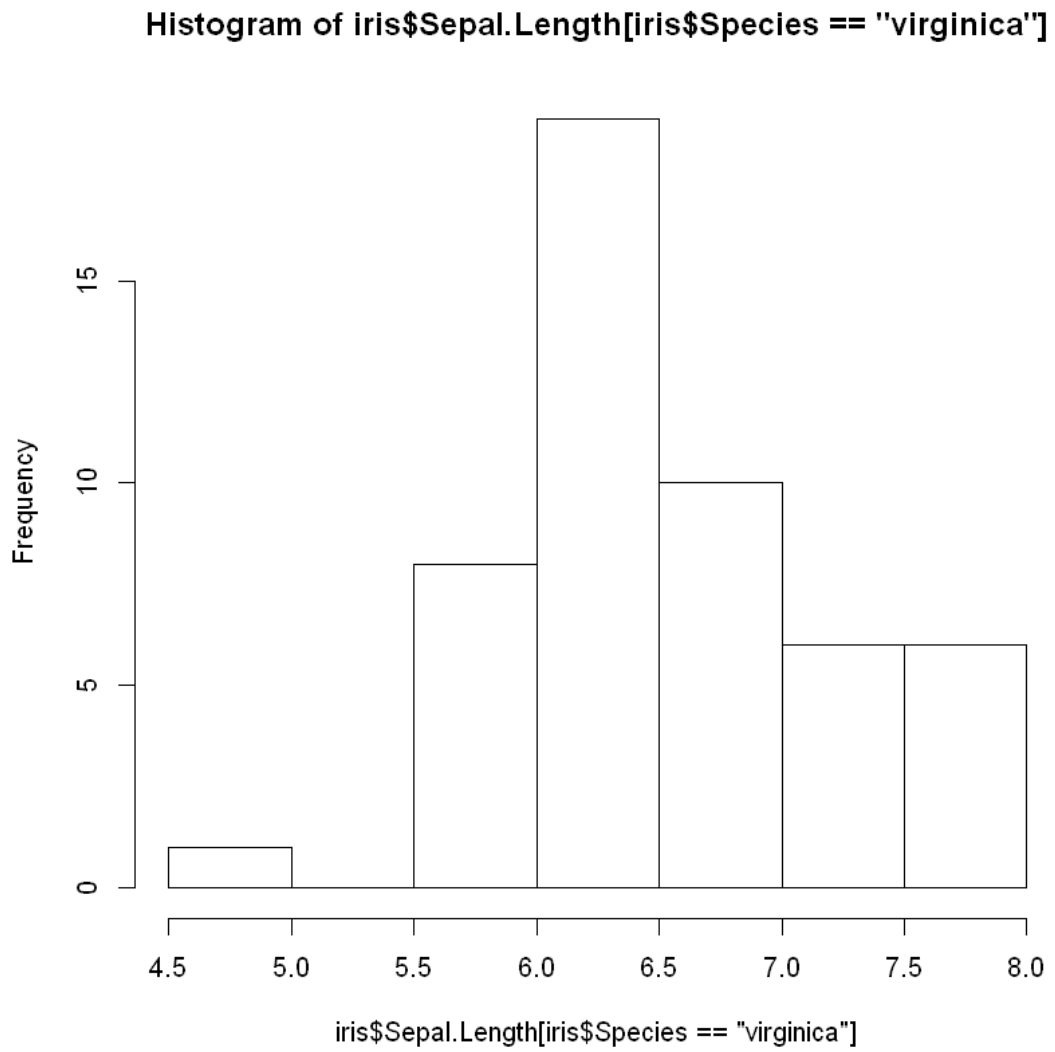
[100]:
```
library(psych)
describe(iris$Sepal.Width)
describe(iris)
```

	vars	n	mean	sd	median	trimmed	mad	min	max	range	skew	ku
X1	1	150	3.057333	0.4358663	3	3.043333	0.44478	2	4.4	2.4	0.3126147	0.

	vars	n	mean	sd	median	trimmed	mad	min	max	range	ske
Sepal.Length	1	150	5.843333	0.8280661	5.80	5.808333	1.03782	4.3	7.9	3.6	0.30
Sepal.Width	2	150	3.057333	0.4358663	3.00	3.043333	0.44478	2.0	4.4	2.4	0.31
Petal.Length	3	150	3.758000	1.7652982	4.35	3.760000	1.85325	1.0	6.9	5.9	-0.2
Petal.Width	4	150	1.199333	0.7622377	1.30	1.184167	1.03782	0.1	2.5	2.4	-0.1
Species*	5	150	2.000000	0.8192319	2.00	2.000000	1.48260	1.0	3.0	2.0	0.00

[102]:
```
mean(iris$Sepal.Length [iris$Species == 'virginica'])
hist(iris$Sepal.Length [iris$Species == 'virginica'])
```

6.588



[115]:
```
tb1 <- iris[iris$Species == 'setosa', which(names(iris) == "Sepal.Length" |␣
 ↪names(iris) == "Sepal.Width")]
tb2 <- iris[iris$Species == 'setosa', which(names(iris) == "Petal.Length" |␣
 ↪names(iris) == "Petal.Width")]
tb3 <- iris[iris$Species == 'setosa', -which(names(iris) == "Species")]
head(tb1)
head(tb2)
head(tb3)
```

Sepal.Length	Sepal.Width
5.1	3.5
4.9	3.0
4.7	3.2
4.6	3.1
5.0	3.6
5.4	3.9

Petal.Length	Petal.Width
1.4	0.2
1.4	0.2
1.3	0.2
1.5	0.2
1.4	0.2
1.7	0.4

Sepal.Length	Sepal.Width	Petal.Length	Petal.Width
5.1	3.5	1.4	0.2
4.9	3.0	1.4	0.2
4.7	3.2	1.3	0.2
4.6	3.1	1.5	0.2
5.0	3.6	1.4	0.2
5.4	3.9	1.7	0.4

Variable types: numeric, character, logical, complex, & raw Common Structures: Vector, Data Frame, Matrix/Array, List Vector - one or more numbers in a 1D array, must have same data type, R's basic data object (all other data structure are variations of vector object) Matrix - rows and columns, two dimensional, all must have same length and same class, columns are not named and referred to using index numbers Array - identical to a matrix with 3 or more dimensions, Data Frame - can have vectors of different types, all must have the same lengths, analogous to spreadsheet List - most flexible data form, ordered collection of elements, lists can include lists

1 Data Types

1.1 Numeric

[1]:
```
n1 <- 15
n1
typeof(n1)
n2 <- 2.5
n2
```

```
typeof(n2)
```

15
'double'
2.5
'double'

1.2 Character

[2]:
```
c1 <- "F"
c1
typeof(c1)
c2 <- "This is an introductory notebook"
c2
typeof(c2)
```

'F'
'character'
'This is an introductory notebook'
'character'

1.3 Logical

[3]:
```
l1 <- TRUE
l1
typeof(l1)
l2 <- T
l2
typeof(l2)
```

TRUE
'logical'
TRUE
'logical'

2 Data Structures

2.1 Vector

[8]:
```
v1 <- c(1, 2, 3, 5)
v1
is.vector(v1)

v2 <- c("one", "two", "three", "five")
v2
is.vector(v2)

v3 <- c(T, F, T, F, F)
```

```
v3
is.vector(v3)

v4 <- c(1, "one", F)
v4
is.vector(v4)

# one ways of printing a vector
for (var in 0:3){
    print(v4[var])
}

# second ways to print
for (var in v4){
    print(var)
}
```

1. 1 2. 2 3. 3 4. 5
TRUE
1. 'one' 2. 'two' 3. 'three' 4. 'five'
TRUE
1. TRUE 2. FALSE 3. TRUE 4. FALSE 5. FALSE
TRUE
1. '1' 2. 'one' 3. 'FALSE'
TRUE

```
character(0)
[1] "1"
[1] "one"
[1] "FALSE"
[1] "1"
[1] "one"
[1] "FALSE"
```

2.2 Matrix

[15]:
```
n1 <- matrix(c(1, 2, 3, 4, 5, 6, 7, 8, 9), nrow=3, byrow=T)
n1
is.matrix(n1)
is.vector(n1)


n1 <- matrix(c(1, 2, 3, 4, 5, 6, 7, 8, 9), nrow=3, byrow=F)
n1
```

1	2	3
4	5	6
7	8	9

TRUE

FALSE

1	4	7
2	5	8
3	6	9

2.3 Array

You need to specify first rows, then columns followed by the number of tables.

[56]:
```
a1 <- array(c(1:24), dim = c(3, 4, 2))
column.names <- c("COL1","COL2","COL3")
row.names <- c("ROW1","ROW2","ROW3")
matrix.names <- c("Matrix1","Matrix2")
a1.dimnames <- c(row.names, matrix.names, matrix.names)
a1[,,1]
a1[,,2]

matrix(a1[,,1], nrow=3, byrow=T)
```

1	4	7	10
2	5	8	11
3	6	9	12

13	16	19	22
14	17	20	23
15	18	21	24

1	2	3	4
5	6	7	8
9	10	11	12

2.4 Dataframe

[85]:
```
cNumeric <- c(1, 2, 3)
cChar <- c("one", "two", "three")
cLogic <- c(T, F, F)

dfa <- cbind(cNumeric, cChar, cLogic)
dfa[,3]
dfa <- as.data.frame(cbind(cNumeric, cChar, cLogic))
dfa[,3]
```

1. 'TRUE' 2. 'FALSE' 3. 'FALSE'

1. TRUE 2. FALSE 3. FALSE

Levels: 1. 'FALSE' 2. 'TRUE'

2.5 List

[89]:
```
cNumeric <- c(1, 2, 3, 4)
cChar <- c("one", "two", "three", "four", "five")
cLogic <- c(T, F, F, F, T, T)
l1 <- list(cNumeric, cChar, cLogic)
```

```
l1
for(x in l1){
    for(y in x){
        print(y)
    }
}
```

1. (a) 1 (b) 2 (c) 3 (d) 4
2. (a) 'one' (b) 'two' (c) 'three' (d) 'four' (e) 'five'
3. (a) TRUE (b) FALSE (c) FALSE (d) FALSE (e) TRUE (f) TRUE

```
[1] 1
[1] 2
[1] 3
[1] 4
[1] "one"
[1] "two"
[1] "three"
[1] "four"
[1] "five"
[1] TRUE
[1] FALSE
[1] FALSE
[1] FALSE
[1] TRUE
[1] TRUE
```

[104]:
```
c1 <- c(1, "two", F)
c1
typeof(c1[1])
typeof(c1[2])
typeof(c1[3])

c2 <- as.logical(c1)
typeof(c2[1])
typeof(c2[2])
typeof(c2[3])

mt <- matrix(1:18, nrow=3)
mt
is.matrix(mt)
df <- as.data.frame(mt)
df
is.data.frame(mt)
is.data.frame(df)
```

1. '1' 2. 'two' 3. 'FALSE'

'character'
'character'
'character'
'logical'
'logical'
'logical'
1 4 7 10 13 16
2 5 8 11 14 17
3 6 9 12 15 18
TRUE

V1	V2	V3	V4	V5	V6
1	4	7	10	13	16
2	5	8	11	14	17
3	6	9	12	15	18

FALSE
TRUE

2.6 Use of factor

[125]:
```
x1 <- 1:3
y <- 1:9

df <- cbind.data.frame(x1, y)
head(df, n=5L)
typeof(df$x1)
str(df)

x1 <- as.factor(c(1:3))
df <- cbind.data.frame(x1, y)
# df$x1 <- factor(df$x1, levels=c("one", "two", "three"))
typeof(df$x1)
head(df)
str(df)


x2 <- c(1:3)
df <- cbind.data.frame(x2, y)
df$x2 <- factor(df$x2, levels=c(1, 2, 3))
typeof(df$x2)
head(df)
str(df)

x3 <- c(1:3)
df <- cbind.data.frame(x3, y)
df$x3 <- factor(df$x3, levels=c(1, 2, 3), labels=c("one", "two", "three"))
typeof(df$x3)
head(df)
str(df)
```

```
x4 <- c(1:3)
df <- cbind.data.frame(x4, y)
df$x4 <- ordered(df$x4, levels=c(3, 1, 2), labels=c("three", "one", "two"))
typeof(df$x4)
head(df)
str(df)
```

x1	y
1	1
2	2
3	3
1	4
2	5

'integer'

```
'data.frame':   9 obs. of  2 variables:
 $ x1: int  1 2 3 1 2 3 1 2 3
 $ y : int  1 2 3 4 5 6 7 8 9
```

'integer'

x1	y
1	1
2	2
3	3
1	4
2	5
3	6

```
'data.frame':   9 obs. of  2 variables:
 $ x1: Factor w/ 3 levels "1","2","3": 1 2 3 1 2 3 1 2 3
 $ y : int  1 2 3 4 5 6 7 8 9
```

'integer'

x2	y
1	1
2	2
3	3
1	4
2	5
3	6

```
'data.frame':   9 obs. of  2 variables:
 $ x2: Factor w/ 3 levels "1","2","3": 1 2 3 1 2 3 1 2 3
 $ y : int  1 2 3 4 5 6 7 8 9
```

'integer'

x3	y
one	1
two	2
three	3
one	4
two	5
three	6

```
'data.frame':   9 obs. of  2 variables:
 $ x3: Factor w/ 3 levels "one","two","three": 1 2 3 1 2 3 1 2 3
 $ y : int  1 2 3 4 5 6 7 8 9
```

'integer'

x4	y
one	1
two	2
three	3
one	4
two	5
three	6

```
'data.frame':   9 obs. of  2 variables:
 $ x4: Ord.factor w/ 3 levels "three"<"one"<..: 2 3 1 2 3 1 2 3 1
 $ y : int  1 2 3 4 5 6 7 8 9
```

3 Creation of ad hoc data

colon (:) - generate sequence of data seq - sequence generation rep - replicate data c - concatenation of arbitrary data scan - read user data

[131]:
```
x1 <- 0:10
x1
x2 <- 10:0
x2
x3 <- seq(10)
x3
x4 <- seq(30, 0, by=-3)
x4
x5 <- c(1, 8, 3, -3, 0)
x5
x6 <- scan()
x6
x7 <- rep(list(T, "one"), 5)
x7
x8 <- rep(list(T, "one"), each=5)
x8
```

1. 0 2. 1 3. 2 4. 3 5. 4 6. 5 7. 6 8. 7 9. 8 10. 9 11. 10
1. 10 2. 9 3. 8 4. 7 5. 6 6. 5 7. 4 8. 3 9. 2 10. 1 11. 0

1. 1 2. 2 3. 3 4. 4 5. 5 6. 6 7. 7 8. 8 9. 9 10. 10
1. 30 2. 27 3. 24 4. 21 5. 18 6. 15 7. 12 8. 9 9. 6 10. 3 11. 0
1. 1 2. 8 3. 3 4. -3 5. 0

1. TRUE
2. 'one'
3. TRUE
4. 'one'
5. TRUE
6. 'one'
7. TRUE
8. 'one'
9. TRUE
10. 'one'

1. TRUE
2. TRUE
3. TRUE
4. TRUE
5. TRUE
6. 'one'
7. 'one'
8. 'one'
9. 'one'
10. 'one'

[132]:
```
library(rio)
```

```
Warning message:
```

[134]:
```
data <- import('mbb.csv')
head(data)
data <- import('mbb.txt')
head(data)
data <- import('mbb.xlsx')
head(data)
```

Month	Mozart	Beethoven	Bach
2004-01	12	8	15
2004-02	12	9	15
2004-03	12	9	14
2004-04	12	8	14
2004-05	11	9	13
2004-06	9	7	12

Month	Mozart	Beethoven	Bach
2004-01	12	8	15
2004-02	12	9	15
2004-03	12	9	14
2004-04	12	8	14
2004-05	11	9	13
2004-06	9	7	12

Month	Mozart	Beethoven	Bach
2004-01	12	8	15
2004-02	12	9	15
2004-03	12	9	14
2004-04	12	8	14
2004-05	11	9	13
2004-06	9	7	12

[139]:
```
head(data)
df_data <- as.data.frame(data)
typeof(df_data)
```

Month	Mozart	Beethoven	Bach
2004-01	12	8	15
2004-02	12	9	15
2004-03	12	9	14
2004-04	12	8	14
2004-05	11	9	13
2004-06	9	7	12

'list'

4 Elementary Data Analysis

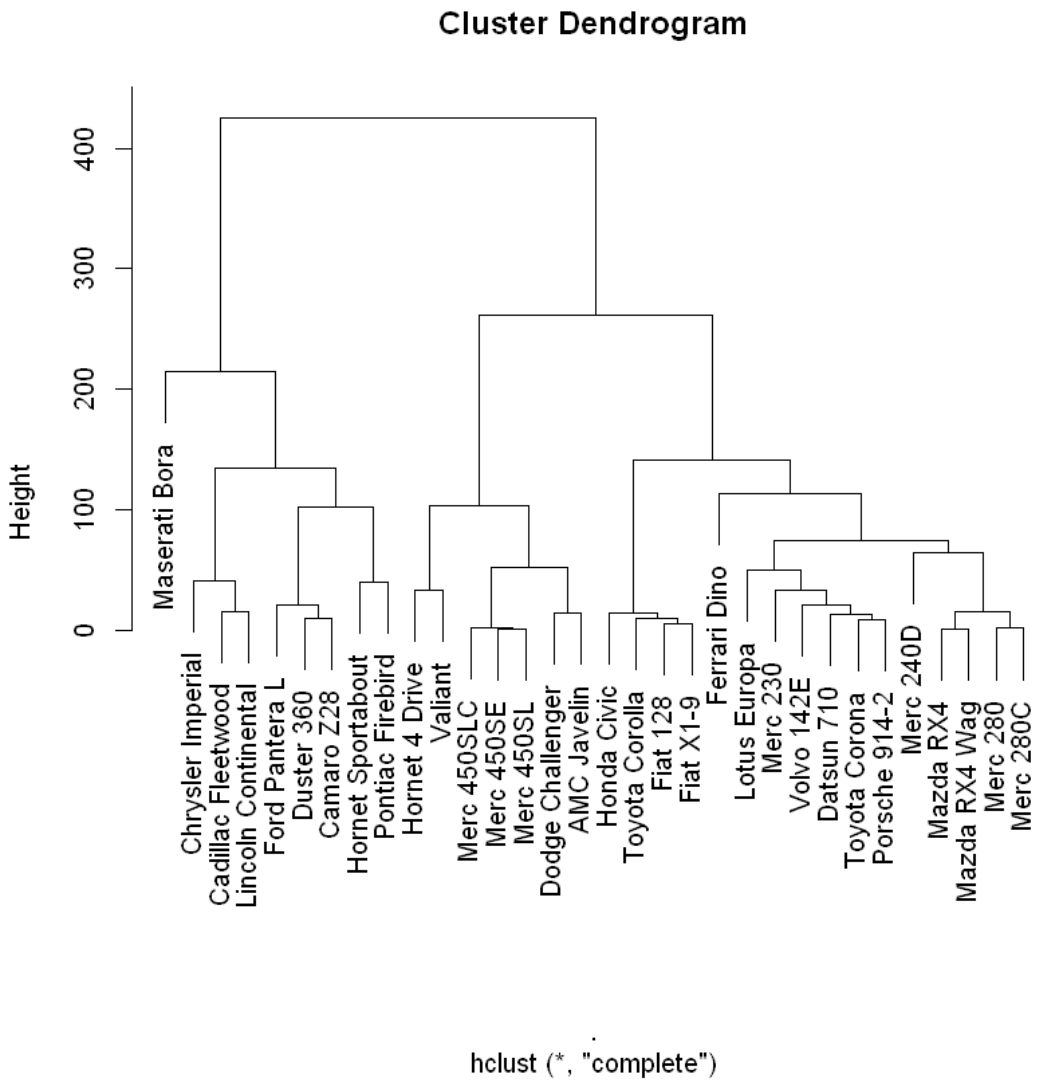
4.1 Hierchical Modelling

[141]:
```
library(datasets)
head(mtcars, n=5L)
cars <- mtcars[, c(1:4, 6:7, 9:11)]
head(cars)
```

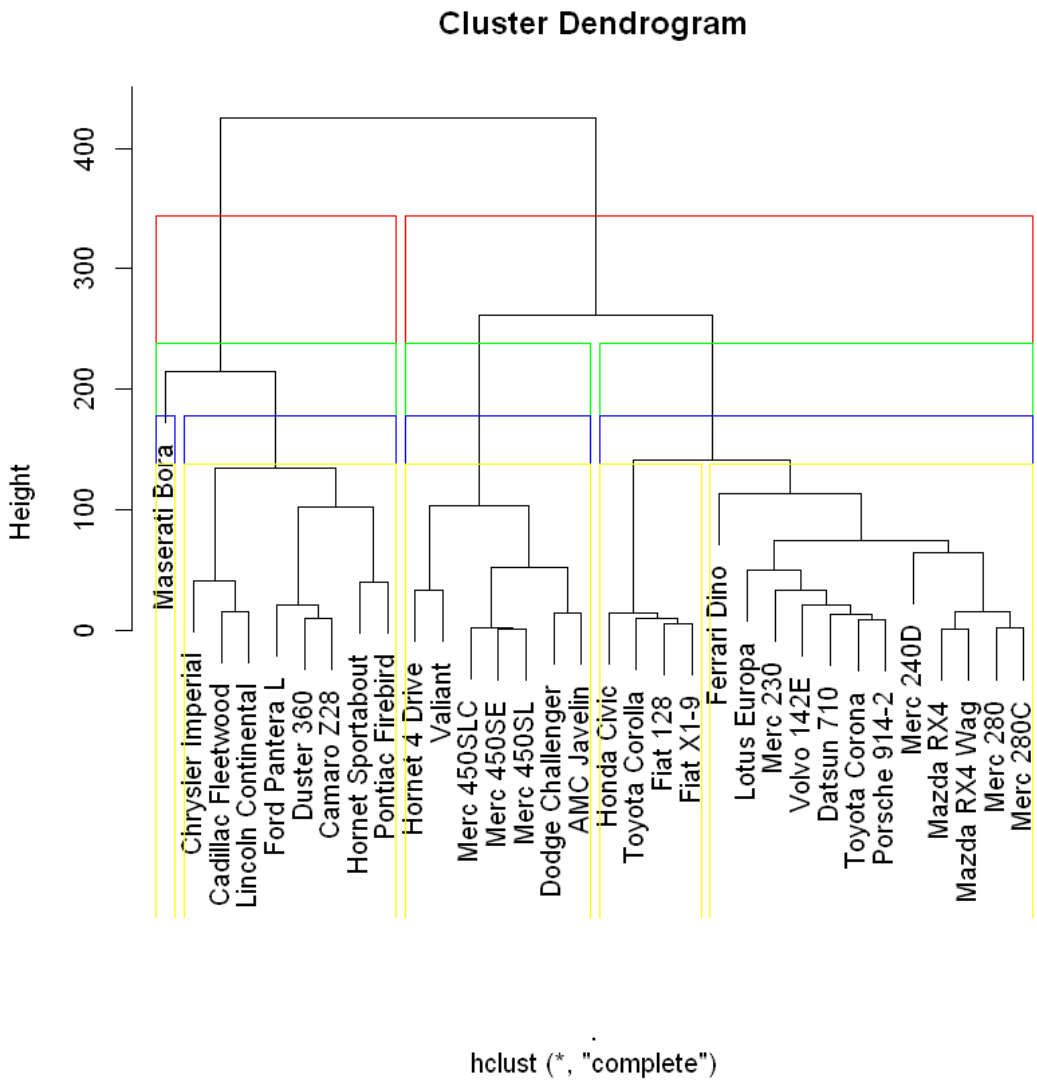
	mpg	cyl	disp	hp	drat	wt	qsec	vs	am	gear	carb
Mazda RX4	21.0	6	160	110	3.90	2.620	16.46	0	1	4	4
Mazda RX4 Wag	21.0	6	160	110	3.90	2.875	17.02	0	1	4	4
Datsun 710	22.8	4	108	93	3.85	2.320	18.61	1	1	4	1
Hornet 4 Drive	21.4	6	258	110	3.08	3.215	19.44	1	0	3	1
Hornet Sportabout	18.7	8	360	175	3.15	3.440	17.02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

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

[145]:
```
library(dplyr)
hc <- cars %>% dist %>% hclust
plot(hc)
```



[148]:
```
plot(hc)
rect.hclust(hc, k = 2, border='red')
rect.hclust(hc, k = 3, border='green')
rect.hclust(hc, k = 4, border='blue')
rect.hclust(hc, k = 5, border='yellow')
```



5 Dimensionality Reduction

[152]:
```
pc <- prcomp(cars, center=T, scale=T)
head(pc)
```

$sdev 1. 2.33914097350423 2. 1.52993832678814 3. 0.718364551062696 4. 0.464905158685231 5. 0.389034831804177 6. 0.350991075344758 7. 0.317137343413083 8. 0.240698933482934 9. 0.149896234132938

$rotation

	PC1	PC2	PC3	PC4	PC5	PC6	PC7
mpg	-0.4023287	0.02205294	-0.17272803	-0.1366169	0.31654561	-0.718609897	0.3633216
cyl	0.4068870	0.03589482	-0.27747610	0.1410976	0.02066646	-0.214224005	0.2099893
disp	0.4046964	-0.06479590	-0.17669890	-0.5089434	0.21525777	0.010052074	0.2007152
hp	0.3699702	0.26518848	-0.01046827	-0.1273173	0.42166543	-0.254229405	-0.6741641
wt	0.3850686	-0.15955242	0.33740464	-0.4469327	-0.21141143	0.002897706	0.3392809
qsec	-0.2168575	-0.48343885	0.54815205	-0.2545226	0.05466817	-0.226660704	-0.2986852
am	-0.2594512	0.46039449	-0.19492256	-0.5354196	-0.55331460	-0.087616182	-0.2135605
gear	-0.2195660	0.50608232	0.34579810	-0.1799814	0.50533262	0.393990378	0.2484622
carb	0.2471604	0.44322600	0.53847588	0.3203064	-0.25696817	-0.398353829	0.1321064

$center **mpg** 20.090625 **cyl** 6.1875 **disp** 230.721875 **hp** 146.6875 **wt** 3.21725 **qsec** 17.84875 **am** 0.40625 **gear** 3.6875 **carb** 2.8125

$scale **mpg** 6.0269480520891 **cyl** 1.78592164694654 **disp** 123.938693831382 **hp** 68.5628684893206 **wt** 0.978457442989697 **qsec** 1.78694323609684 **am** 0.498990917235846 **gear** 0.737804065256947 **carb** 1.61519997763185

	PC1	PC2	PC3	PC4	PC5	PC6
Mazda RX4	-0.81883768	1.45577333	-0.21204263	0.315888300	-0.84958691	-0.01150126
Mazda RX4 Wag	-0.78644303	1.26268953	0.04767210	0.119647855	-0.88755160	-0.08177799
Datsun 710	-2.49423117	0.02762658	-0.32023017	-0.401948370	-0.36518038	0.53888511
Hornet 4 Drive	-0.29454234	-1.92903945	-0.32211475	-0.069818183	0.20547103	-0.04600804
Hornet Sportabout	1.56041411	-0.80821419	-1.04219408	0.050065675	0.38197028	-0.13573066
Valiant	-0.20722532	-2.19417266	0.14402455	-0.073226863	-0.08498911	0.26511187
Duster 360	2.73226603	0.29328994	-0.57716172	0.525124977	0.19900274	-0.21386156
Merc 240D	-1.79527743	-1.27281225	1.03388048	0.136366170	0.39973745	0.22142233
Merc 230	-1.89734058	-1.92598643	1.95890184	-0.259206293	0.60577005	-0.07860918
Merc 280	0.01565012	-0.05866208	1.06454809	0.737712361	0.13700873	0.10015509
Merc 280C	0.03629307	-0.22610850	1.28872352	0.683986341	0.08183421	0.19097540
Merc 450SE	1.82083345	-0.68439747	-0.18980574	0.295092091	-0.13790858	-0.17982680
Merc 450SL	1.60267678	-0.67977004	-0.27149159	0.401507010	-0.01105796	-0.31351178
Merc 450SLC	1.71399687	-0.80382315	-0.07136381	0.369296647	-0.11991960	-0.11371190
Cadillac Fleetwood	3.54393557	-0.78715158	0.61681226	-0.844299902	-0.35483328	0.14208110
$x Lincoln Continental	3.64660694	-0.72728678	0.64331413	-0.870281313	-0.35666482	0.12483822
Chrysler Imperial	3.39264826	-0.52198151	0.39635946	-0.820419326	-0.06847485	-0.39460143
Fiat 128	-3.52803830	-0.23945546	-0.32703554	-0.516783758	-0.02567396	-0.61745094
Honda Civic	-3.44178368	0.32746057	-0.42306580	0.167700576	-0.28378711	-0.45517710
Toyota Corolla	-3.85421097	-0.29067456	-0.35299640	-0.412244409	0.12577796	-0.84883188
Toyota Corona	-1.64164478	-1.97896631	0.10056967	0.621710410	0.04761048	0.14446951
Dodge Challenger	1.55167305	-0.86712498	-0.90521454	0.326318496	-0.03467077	0.35437059
AMC Javelin	1.44035057	-0.96337487	-0.77406360	0.368187375	-0.04322194	0.33421087
Camaro Z28	2.92480902	0.36716333	-0.57304474	0.526775004	0.05762007	-0.04009785
Pontiac Firebird	1.81339410	-0.90145453	-0.96469148	-0.314790674	0.39111452	-0.19470872
Fiat X1-9	-3.22172493	-0.06085364	-0.44753150	-0.200178011	-0.25319420	0.06217622
Porsche 914-2	-2.66209565	1.53159161	-0.27507492	-0.212645194	0.31823141	0.69486607
Lotus Europa	-3.19041442	1.69409211	-0.52346685	0.008155493	0.78245261	0.05939704
Ford Pantera L	1.59533098	3.09923346	-0.61246644	-0.694517979	0.68539841	0.59731381
Ferrari Dino	-0.24630742	3.18027405	0.72936287	0.507145572	-0.23921602	0.06422736
Maserati Bora	2.62596044	4.40241877	0.97303537	-0.006628448	0.27345257	-0.57263382
Volvo 142E	-1.93672169	0.27969720	0.18785195	-0.463691632	-0.57652141	0.40354034

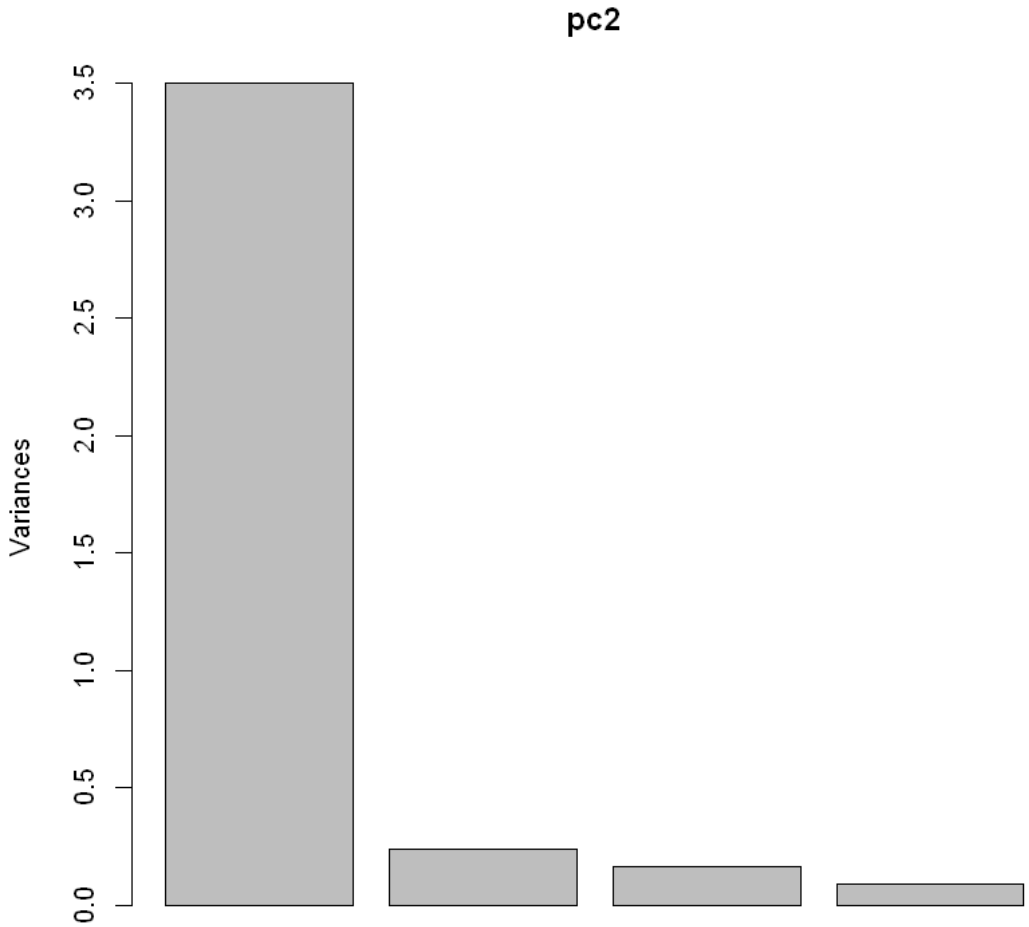
[153]:
```
pc2 <- prcomp(~mpg + disp + hp + cyl, data=cars, center=T, scale=T)
head(pc2)
summary(pc2)
```

```
$sdev
[1] 1.8714034 0.4893434 0.4065238 0.3051731

$rotation
            PC1         PC2        PC3          PC4
mpg  -0.4963126 -0.41505710  0.7624369 -0.009557844
disp  0.5060829  0.31928855  0.5109886  0.617110666
hp    0.4844917 -0.84776090 -0.1441097  0.160628854
cyl   0.5126614  0.08416586  0.3698824 -0.770247652
```

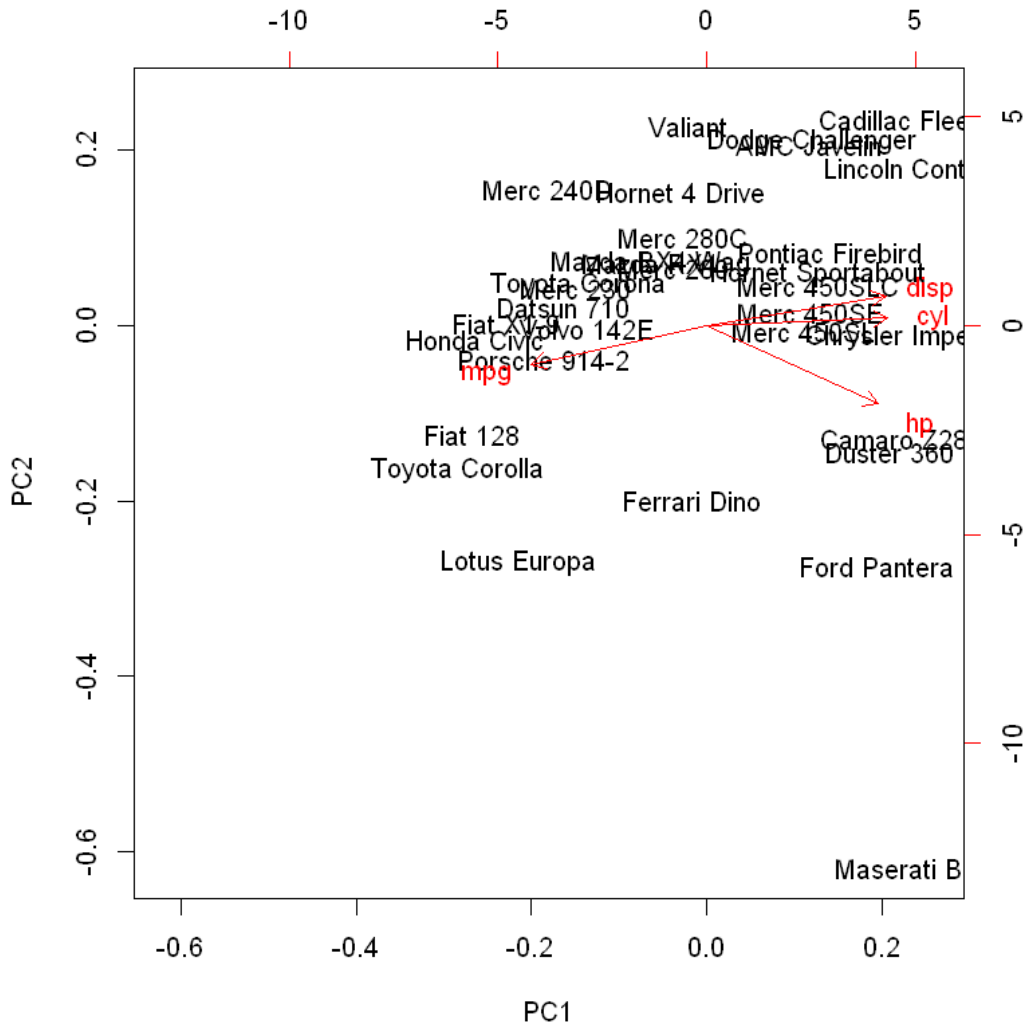
```
$center
      mpg      disp        hp       cyl
 20.09062 230.72188 146.68750   6.18750

$scale
       mpg       disp         hp        cyl
  6.026948 123.938694  68.562868   1.785922

$x
                          PC1          PC2          PC3         PC4
Mazda RX4           -0.6767382  0.199976205 -0.138261099 -0.35866245
Mazda RX4 Wag       -0.6767382  0.199976205 -0.138261099 -0.35866245
Datsun 710          -1.7315422  0.057999871 -0.503432795  0.20231633
Hornet 4 Drive      -0.3095112  0.424895223  0.316386321  0.12866096
Hornet Sportabout    1.3627600  0.164154097  0.672960127 -0.06947829
Valiant             -0.2078417  0.628965554 -0.226625858 -0.04213202
Duster 360           2.2197422 -0.398362269 -0.030790698  0.10149522
Merc 240D           -1.9243334  0.430817504 -0.076310853  0.31984579
Merc 230            -1.5834761  0.117769171 -0.372404944  0.37031839
Merc 280            -0.4056138  0.182774161 -0.361959531 -0.28750994
Merc 280C           -0.2903254  0.279187791 -0.539066036 -0.28528975
Merc 450SE           1.2436778  0.043809833  0.024340777 -0.47336217
Merc 450SL           1.1695638 -0.018170357  0.138194959 -0.47478944
Merc 450SLC          1.3424965  0.126450087 -0.127464799 -0.47145915
Cadillac Fleetwood   2.7155808  0.653339795  0.021682040  0.57163417
Lincoln Continental  2.7372446  0.498778386 -0.048811553  0.53531221
Chrysler Imperial    2.4074735 -0.034343665  0.381172211  0.46405178
Fiat 128            -2.8325258 -0.344756743  0.646960724 -0.02205276
Honda Civic         -2.7790074 -0.041645405  0.411008765 -0.06661769
Toyota Corolla      -2.9941488 -0.455271315  0.807485411 -0.06461595
Toyota Corona       -1.5468147  0.129239713 -0.626409027  0.27399697
Dodge Challenger     1.2781168  0.585446574  0.147529439 -0.33209823
AMC Javelin          1.2456549  0.570040098  0.051857250 -0.40133072
Camaro Z28           2.2612578 -0.355257204 -0.198524483  0.05328946
Pontiac Firebird     1.4849188  0.232767911  0.901129006  0.12889521
Fiat X1-9           -2.4113213  0.007237189  0.003023901 -0.01247117
Porsche 914-2       -1.9589665 -0.105957663 -0.043702360  0.25379969
Lotus Europa        -2.2687411 -0.745915608  0.362779678  0.17288862
Ford Pantera L       2.1937302 -0.759777960  0.081924658  0.09881713
Ferrari Dino        -0.1716195 -0.552846586 -0.501181915 -0.27900647
Maserati Bora        2.5571561 -1.711388877 -0.374656997  0.01746634
Volvo 142E          -1.4501082 -0.009931715 -0.660571220  0.30675036

$call
prcomp(formula = ~mpg + disp + hp + cyl, data = cars, center = T,
    scale = T)
```

```
Importance of components:
                          PC1     PC2     PC3     PC4
Standard deviation     1.8714 0.48934 0.40652 0.30517
Proportion of Variance 0.8755 0.05986 0.04132 0.02328
Cumulative Proportion  0.8755 0.93540 0.97672 1.00000
```

[154]:
```
plot(pc2)
```



[155]:
```
biplot(pc2)
```



6 Regression

[157]:
```
head(USJudgeRatings)
?USJudgeRatings
```

	CONT	INTG	DMNR	DILG	CFMG	DECI	PREP	FAMI	ORAL	WRIT
AARONSON,L.H.	5.7	7.9	7.7	7.3	7.1	7.4	7.1	7.1	7.1	7.0
ALEXANDER,J.M.	6.8	8.9	8.8	8.5	7.8	8.1	8.0	8.0	7.8	7.9
ARMENTANO,A.J.	7.2	8.1	7.8	7.8	7.5	7.6	7.5	7.5	7.3	7.4
BERDON,R.I.	6.8	8.8	8.5	8.8	8.3	8.5	8.7	8.7	8.4	8.5
BRACKEN,J.J.	7.3	6.4	4.3	6.5	6.0	6.2	5.7	5.7	5.1	5.3
BURNS,E.B.	6.2	8.8	8.7	8.5	7.9	8.0	8.1	8.0	8.0	8.0

Format A data frame containing 43 observations on 12 numeric variables.

[,1] CONT Number of contacts of lawyer with judge. [,2] INTG Judicial integrity. [,3] DMNR Demeanor. [,4] DILG Diligence. [,5] CFMG Case flow managing. [,6] DECI Prompt decisions. [,7] PREP Preparation for trial. [,8] FAMI Familiarity with law. [,9] ORAL Sound oral rulings. [,10] WRIT Sound written rulings. [,11] PHYS Physical ability. [,12] RTEN Worthy of retention.

[160]:
```
df <- USJudgeRatings
x <- as.matrix(df[-12])
y <- as.matrix(df[,12])

# linear model function name : lm
reg_1 <- lm(y ~ x)

reg_1

summary(reg_1)
```

```
Call:
lm(formula = y ~ x)

Coefficients:
(Intercept)        xCONT        xINTG        xDMNR        xDILG        xCFMG
   -2.11943      0.01280      0.36484      0.12540      0.06669     -0.19453
      xDECI        xPREP        xFAMI        xORAL        xWRIT        xPHYS
    0.27829     -0.00196     -0.13579      0.54782     -0.06806      0.26881



Call:
lm(formula = y ~ x)

Residuals:
     Min       1Q   Median       3Q      Max
-0.22123 -0.06155 -0.01055  0.05045  0.26079

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept) -2.11943    0.51904  -4.083 0.000290 ***
xCONT        0.01280    0.02586   0.495 0.624272
xINTG        0.36484    0.12936   2.820 0.008291 **
xDMNR        0.12540    0.08971   1.398 0.172102
xDILG        0.06669    0.14303   0.466 0.644293
xCFMG       -0.19453    0.14779  -1.316 0.197735
xDECI        0.27829    0.13826   2.013 0.052883 .
xPREP       -0.00196    0.24001  -0.008 0.993536
xFAMI       -0.13579    0.26725  -0.508 0.614972
xORAL        0.54782    0.27725   1.976 0.057121 .
xWRIT       -0.06806    0.31485  -0.216 0.830269
xPHYS        0.26881    0.06213   4.326 0.000146 ***
```

```
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.1174 on 31 degrees of freedom
Multiple R-squared:  0.9916,Adjusted R-squared:  0.9886
F-statistic: 332.9 on 11 and 31 DF,  p-value: < 2.2e-16
```

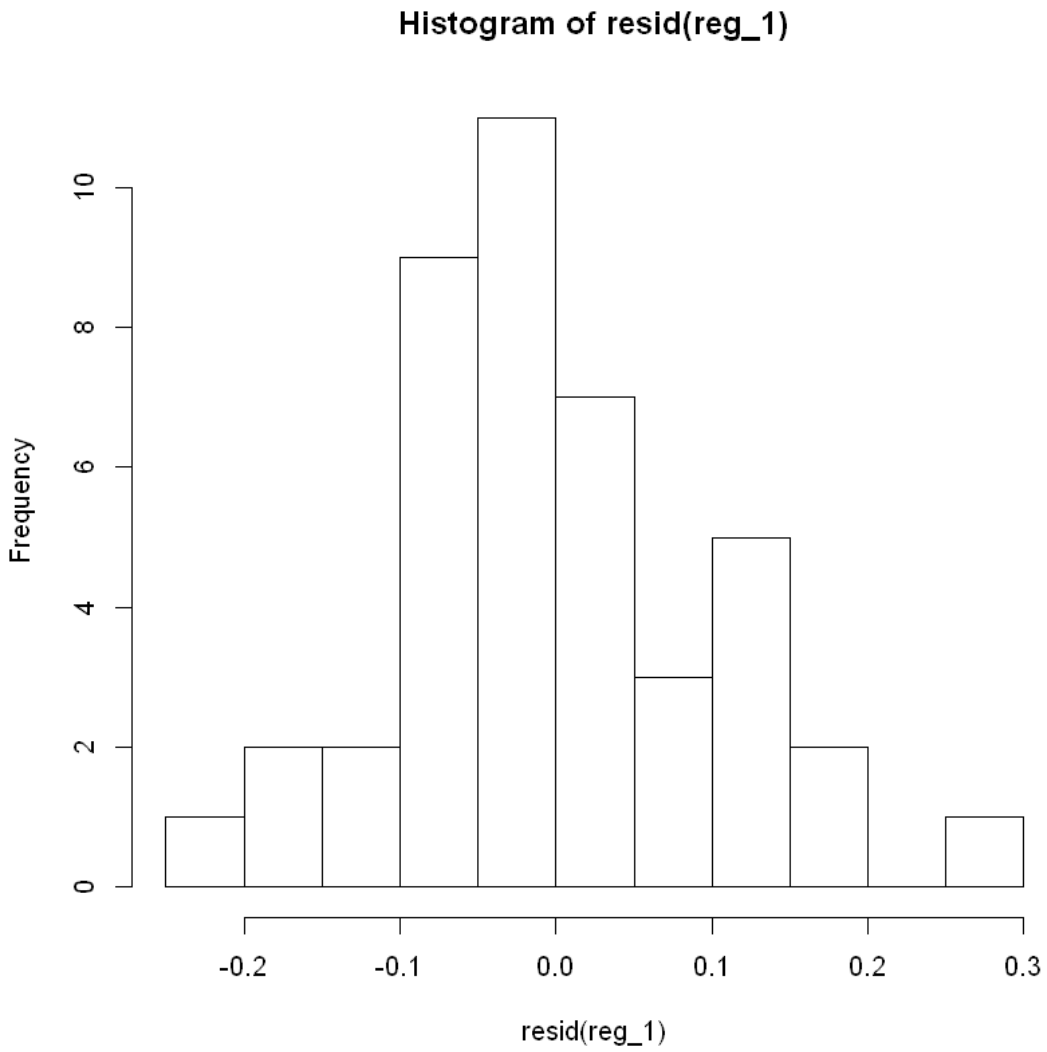
[164]:
```
anova(reg_1)
coef(reg_1)
confint(reg_1)
resid(reg_1)
hist(resid(reg_1))
```

	Df	Sum Sq	Mean Sq	F value	Pr(>F)
x	11	50.4823548	4.5893050	332.8597	5.745717e-29
Residuals	31	0.4274127	0.0137875	NA	NA

(Intercept) -2.11942968179327 **xCONT** 0.0127963773918441 **xINTG** 0.364840272014892 **xDMNR** 0.125399137854698 **xDILG** 0.0666909760662366 **xCFMG** -0.194527026617029 **xDECI** 0.278292931605456 **xPREP** -0.00196011133377008 **xFAMI** -0.135790972195287 **xORAL** 0.547817679832884 **xWRIT** -0.0680615953914752 **xPHYS** 0.268811919161933

	2.5 %	97.5 %
(Intercept)	-3.178010347	-1.06084902
xCONT	-0.039955335	0.06554809
xINTG	0.101011150	0.62866939
xDMNR	-0.057571651	0.30836993
xDILG	-0.225031708	0.35841366
xCFMG	-0.495940888	0.10688683
xDECI	-0.003683181	0.56026904
xPREP	-0.491456059	0.48753584
xFAMI	-0.680844080	0.40926214
xORAL	-0.017628284	1.11326364
xWRIT	-0.710196975	0.57407378
xPHYS	0.142088434	0.39553540

1 0.167428295017044 **2** 0.159904302772648 **3** 0.131818800299843 **4** -0.0721243487660769 **5** -0.166351358367794 **6** 0.0344455088067175 **7** -0.122867277430027 **8** -0.035984506502978 **9** -0.0414643392723508 **10** 0.105484916712862 **11** 0.0315661299294979 **12** 0.0279048489769697 **13** -0.0066302843520421 **14** 0.121511625754103 **15** -0.0707169454541466 **16** 0.0963751277156313 **17** 0.0966781230774998 **18** 0.0587324089912389 **19** 0.260791430392931 **20** -0.0613783951036348 **21** -0.0105476009574108 **22** -0.0926140135370371 **23** -0.0964022148655106 **24** -0.0479617599866199 **25** 0.0279999236295551 **26** -0.063366251055408 **27** -0.014242307647406 **28** -0.191822695590151 **29** 0.0253091921664977 **30** -0.0179725261543004 **31** -0.0144131915072458 **32** 0.114510447001218 **33** -0.0617147924758255 **34** -0.0608608819965285 **35** 0.0421019215183924 **36** 0.147460609559765 **37** 0.0421784996965396 **38** -0.221232591120357 **39** -0.0375263259888027 **40** -0.000753779929522136 **41** -0.00242778449423707 **42** -0.120465634669108 **43** -0.0603603047944356



6.1 Least angle regression (LARS)

6.2 Classification and regression training (CARET)

[168]:
```
# library(lars)
# library(caret)
pacman::p_load(lars, caret)
```

```
Installing package into 'C:/Users/HP/Documents/R/win-library/3.6'
(as 'lib' is unspecified)
Warning message:
"unable to access index for repository
```

```
http://www.stats.ox.ac.uk/pub/RWin/bin/windows/contrib/3.6:
  cannot open URL

package 'lars' successfully unpacked and MD5 sums checked

The downloaded binary packages are in
        C:\Users\HP\AppData\Local\Temp\Rtmp8Y3Hmf\downloaded_packages


lars installed
```

[169]:
```
library(lars)
library(caret)
```

[184]:
```
stepwise <- lars(x, y, type='stepwise')
forward <- lars(x, y, type='forward.stagewise')
lar <- lars(x, y, type='lar')
stepwise$R2[6] %>% round(2)
forward$R2[6] %>% round(2)
lar$R2[6] %>% round(2)
```

5: 0.99
5: 0.99
5: 0.99