Comprehensive Overview of Statistical Learning

Statistical Learning Overview

Statistical learning aims to estimate a function f that explains the relationship between input variables (predictors) and an output variable (response). The primary goal is to build models that can either provide accurate predictions or offer insights into the relationships among variables.

The expected squared error of a prediction can be decomposed into two components:

- **Reducible error**: This can be minimized by improving the estimation of f. It arises from approximating the true relationship with a model.
- **Irreducible error**: This component is due to inherent variability in the data and cannot be reduced, even with a perfect model.

Inference vs. Prediction

Inference seeks to understand the relationships between variables, answering questions such as which predictors significantly affect the response, and the nature of those relationships.

Prediction, on the other hand, focuses on accurately estimating future observations without necessarily understanding the underlying relationships. The choice between inference and prediction depends on the problem context.

For example:

- A marketing analyst predicting future sales based on advertising expenditure focuses on **prediction**.
- A medical researcher identifying risk factors for a disease is more interested in **inference**.
- Real estate modeling may involve both, as predicting home prices is essential while understanding the effects of location and features is also valuable.

Estimating f

The function f can be estimated using either **parametric** or **non-parametric** methods:

- **Parametric Methods** assume a specific functional form, simplifying estimation to a small number of parameters. A common example is the **linear model**, which assumes:

$$f(X) = \beta_0 + \beta_1 X_1 + \cdots + \beta_p X_p$$

  These models are computationally efficient but may not capture complex relationships.

- **Non-Parametric Methods** make fewer assumptions about the form of f, allowing for greater flexibility. However, they require large amounts of data for accurate estimation. Examples include **thin-plate splines** and **local regression**.

Overfitting and Model Flexibility

A key challenge in statistical learning is balancing **model flexibility** and **generalizability**. More flexible models can fit the training data very well but may also capture noise, leading to **overfitting**. Less flexible models may generalize better but might miss important patterns (underfitting).

Strategies to mitigate overfitting include:

- **Regularization methods** such as **lasso regression**, which penalizes large coefficients.
- **Cross-validation** to evaluate model performance on unseen data.
- **Pruning in decision trees** to remove unnecessary complexity.

There is a trade-off between **prediction accuracy** and **model interpretability**. While **linear models** are easier to interpret, they may not capture non-linear relationships. More flexible methods, such as **generalized additive models (GAMs)** and **neural networks**, provide better predictions but are harder to interpret.

Supervised vs. Unsupervised Learning

- **Supervised Learning**: Each observation has both predictor variables and a response variable, allowing us to model the relationship. Examples include **linear regression, logistic regression, GAMs, and boosting**.
- **Unsupervised Learning**: There is no response variable; the goal is to discover patterns within the predictors. Methods such as **clustering, principal component analysis (PCA)**, and **autoencoders** are used for this purpose.

K-Nearest Neighbors (KNN) Classifier

The **K-Nearest Neighbors (KNN)** classifier is a simple yet powerful non-parametric method for classification. Given a positive integer K and a test observation x_0, KNN identifies the K training observations closest to x_0 and assigns it to the most common class among those neighbors.

The conditional probability of an observation belonging to class j is estimated as:

$$Pr(Y = j|X = x_0) = \frac{1}{K} \sum_{i \in N_0} I(y_i = j)$$

where N_0 represents the set of nearest neighbors.

Key considerations for KNN:

- **Choice of** K: A small K increases variance and overfitting, while a large K results in high bias and oversmoothing.

- **Distance metric**: Common choices include Euclidean distance and Manhattan distance.
- **Scalability**: KNN can be computationally expensive for large datasets.

KNN is highly flexible, but the choice of K significantly affects its performance. The test error rate typically follows a U-shaped curve, decreasing initially with increased flexibility but rising again due to overfitting.

Assessing Model Accuracy

Selecting the best model depends on the data structure and task requirements. Different statistical learning methods exist because **no single method dominates across all datasets** (*no free lunch theorem*).

A key metric for evaluating model performance in regression is the **mean squared error (MSE)**:

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{f}(x_i))^2$$

where $\hat{f}(x_i)$ is the model's prediction for observation i. A small MSE indicates that predicted values are close to actual values.

Other key considerations:

- **Training vs. Test MSE**: Training MSE is computed on the training data and tends to be lower, while test MSE reflects generalization performance.
- **Cross-validation**: A technique used to estimate test MSE by splitting the data into training and validation sets.
- **Bias-Variance Trade-off**: More complex models typically have low bias but high variance, while simpler models have high bias but low variance.

Conclusion

This chapter introduced fundamental concepts in statistical learning, covering the trade-offs between model flexibility and interpretability, supervised and unsupervised learning, and key evaluation metrics. Understanding these principles is crucial for selecting the appropriate statistical learning method for a given problem. The next steps involve exploring specific techniques in greater depth and applying them to real-world datasets to enhance predictive accuracy and interpretability.

Stat 702 HW1 Intro to R

2025-02-05

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

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

```
x = c(1, 6, 2)
y = c(1, 4, 3)

length(x)
```

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

```
length(y)
```

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

```
x + y
```

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

```
ls()
```

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

```
rm(x, y)
ls()
```

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

```
rm(list = ls())

# Creating a matrix
?matrix
x <- matrix(data = c(1, 2, 3, 4), nrow = 2, ncol = 2)
x
```

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

```
x <- matrix(c(1, 2, 3, 4), 2, 2)
x
```

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

```
matrix(c(1, 2, 3, 4), 2, 2, byrow = TRUE)
```

```
##      [,1] [,2]
## [1,]    1    2
```

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

```
sqrt(x)
```

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

```
x^2
```

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

```
# Random normal variables and correlation
x <- rnorm(50)
y <- x + rnorm(50, mean = 50, sd = 0.1)
cor(x, y)
```

```
## [1] 0.9937619
```

```
set.seed(1303)
rnorm(50)
```

```
##  [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
```

```
set.seed(3)
y <- rnorm(100)
mean(y)
```

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

```
var(y)
```

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

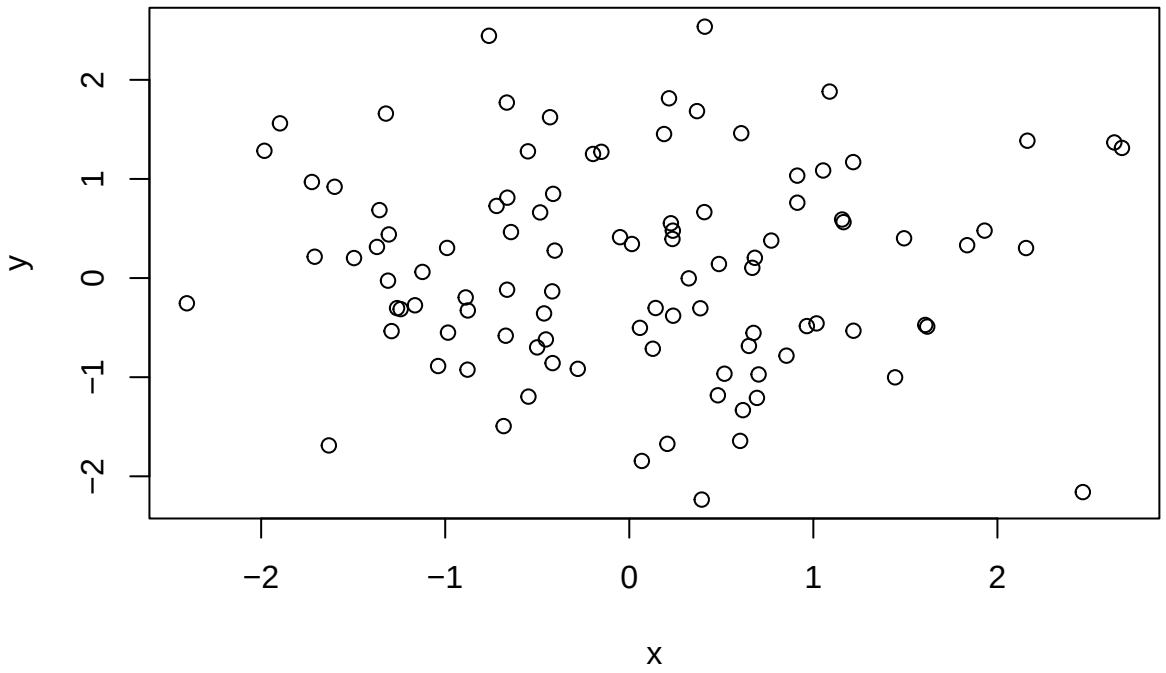
```
sqrt(var(y))
```

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

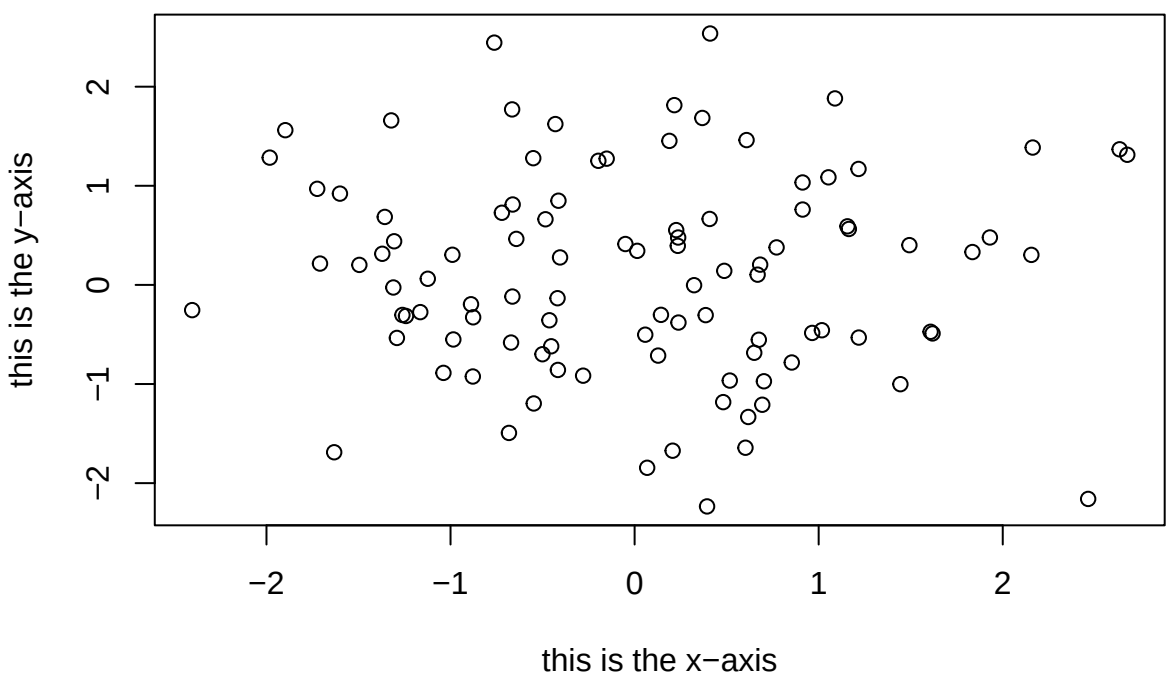
```
sd(y)
```

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

```
# Graphics
x <- rnorm(100)
y <- rnorm(100)
plot(x, y)
```




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




```
pdf("Figure.pdf")
plot(x, y, col = "green")
dev.off()
```

```
## pdf
##   2
```

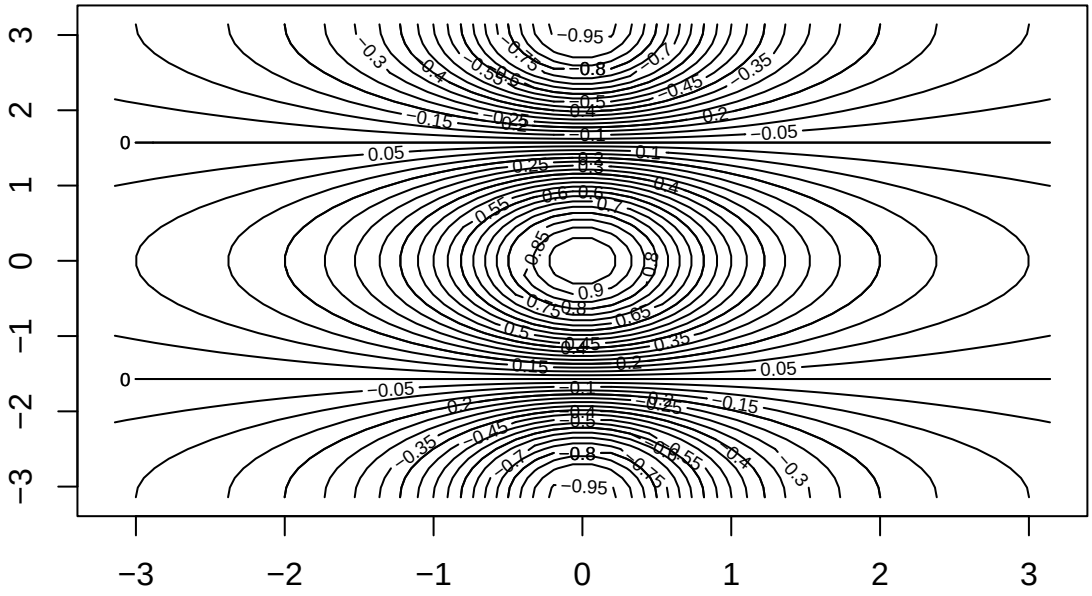
```r
# Sequences
x <- seq(1, 10)
x
```

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

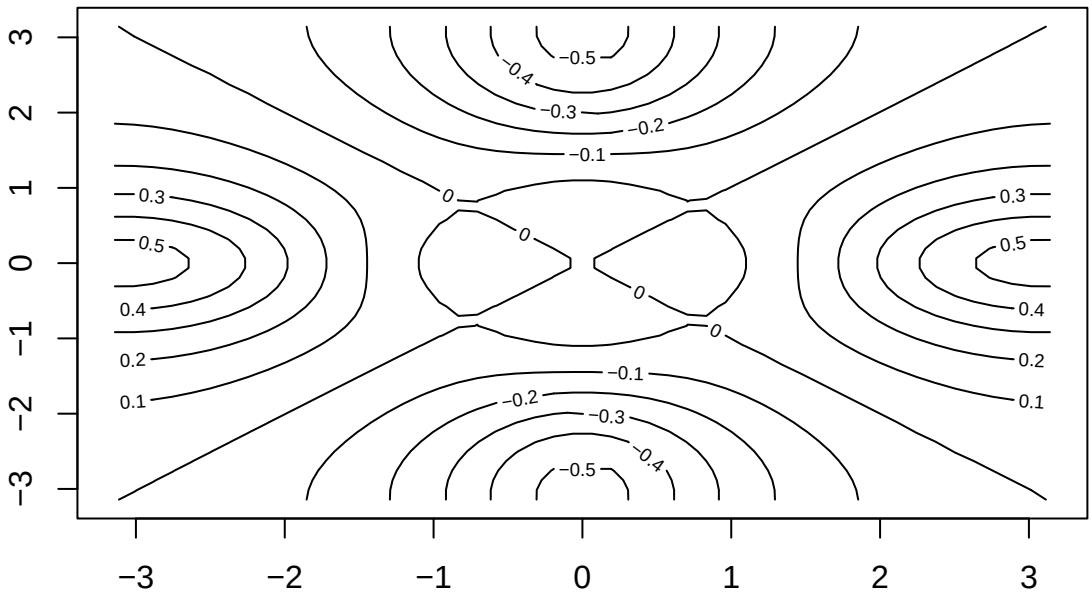
```r
x <- 1:10
x
```

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

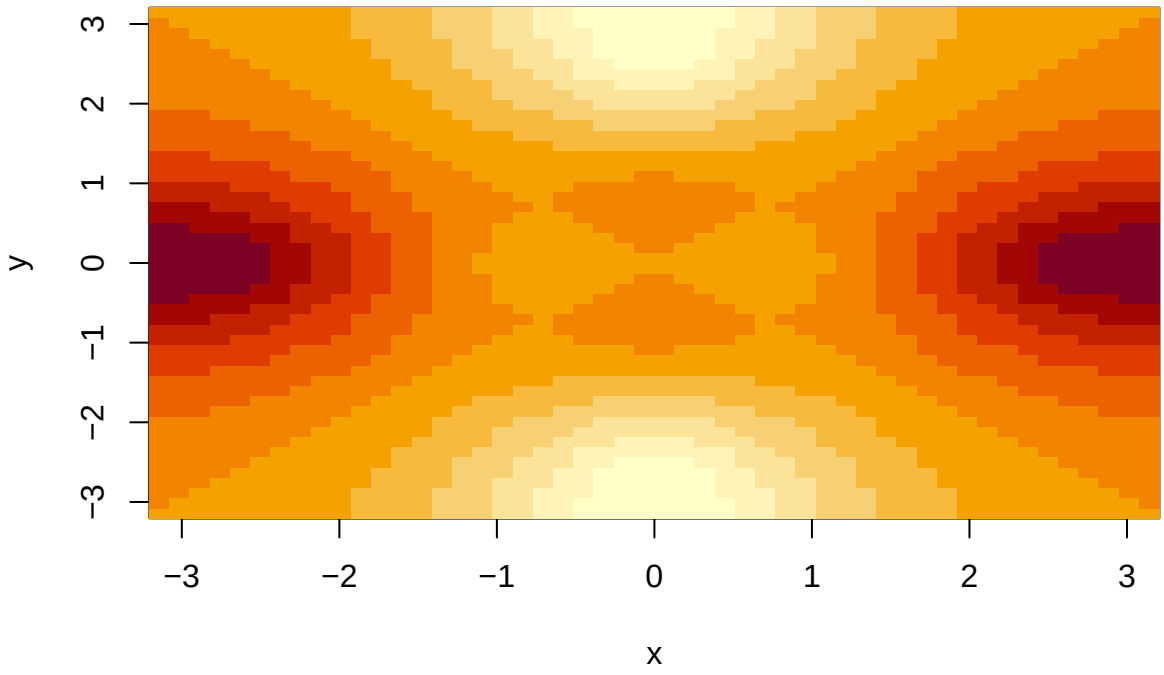
```r
x <- seq(-pi, pi, length = 50)

# Contour, Image, and Perspective plots
y <- x
f <- outer(x, y, function(x, y) cos(y) / (1 + x^2))
contour(x, y, f)
contour(x, y, f, nlevels = 45, add = TRUE)
```



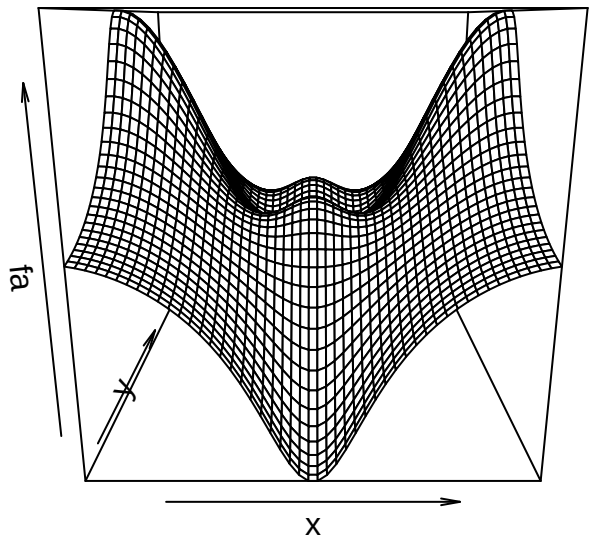

```r
fa <- (f - t(f)) / 2
contour(x, y, fa, nlevels = 15)
```



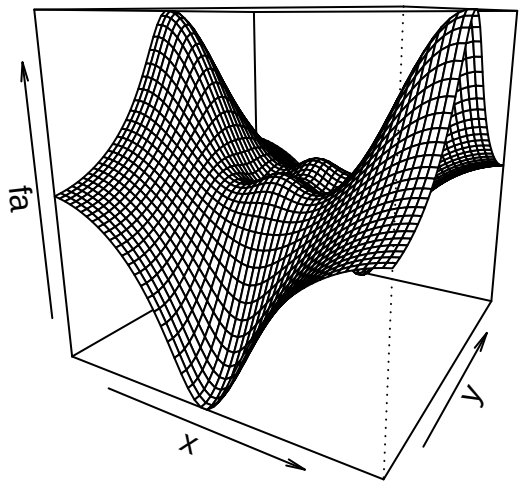

```
image(x, y, fa)
```



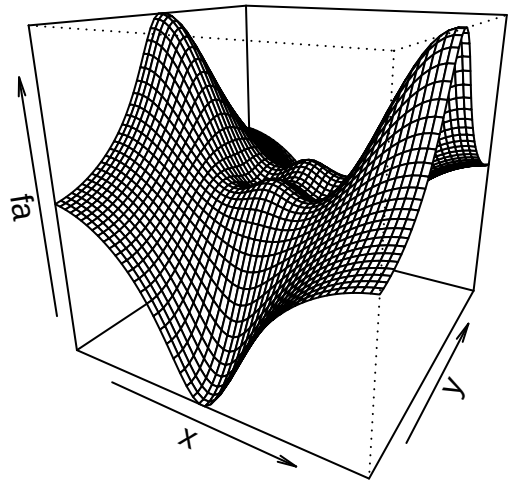

```
persp(x, y, fa)
```



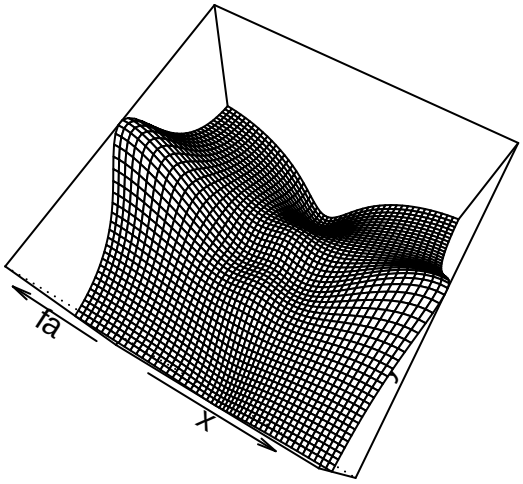

```
persp(x, y, fa, theta = 30)
```



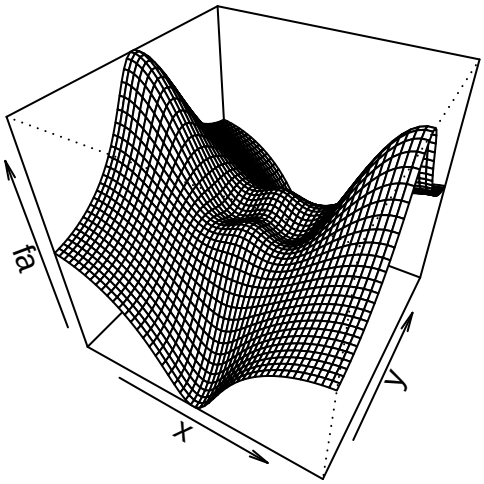

```
persp(x, y, fa, theta = 30, phi = 20)
```




```
persp(x, y, fa, theta = 30, phi = 70)
```




```
persp(x, y, fa, theta = 30, phi = 40)
```




```
# 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
```

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

```
##      [,1] [,2] [,3]
## [1,]    5    9   13
```

```
## [2,]    6   10   14
## [3,]    7   11   15
```

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

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

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

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

```
A[1, ]
```

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

```
A[-c(1, 3), ]
```

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

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

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

```
dim(A)
```

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

```
# Loading Data

Auto <- read.table("Auto.data")
View(Auto)
head(Auto)
```

```
##     V1        V2           V3         V4     V5           V6   V7     V8
## 1  mpg cylinders displacement horsepower weight acceleration year origin
## 2 18.0         8        307.0      130.0  3504.         12.0   70      1
## 3 15.0         8        350.0      165.0  3693.         11.5   70      1
## 4 18.0         8        318.0      150.0  3436.         11.0   70      1
## 5 16.0         8        304.0      150.0  3433.         12.0   70      1
## 6 17.0         8        302.0      140.0  3449.         10.5   70      1
##                          V9
## 1                      name
## 2 chevrolet chevelle malibu
## 3         buick skylark 320
## 4        plymouth satellite
## 5             amc rebel sst
## 6               ford torino
```

```
Auto <- read.table("Auto.data", header = TRUE, na.strings = "?", stringsAsFactors = TRUE)
View(Auto)
#Auto <- read.csv("Auto.csv", na.strings = "?", stringsAsFactors = TRUE)
View(Auto)
```

```
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
```

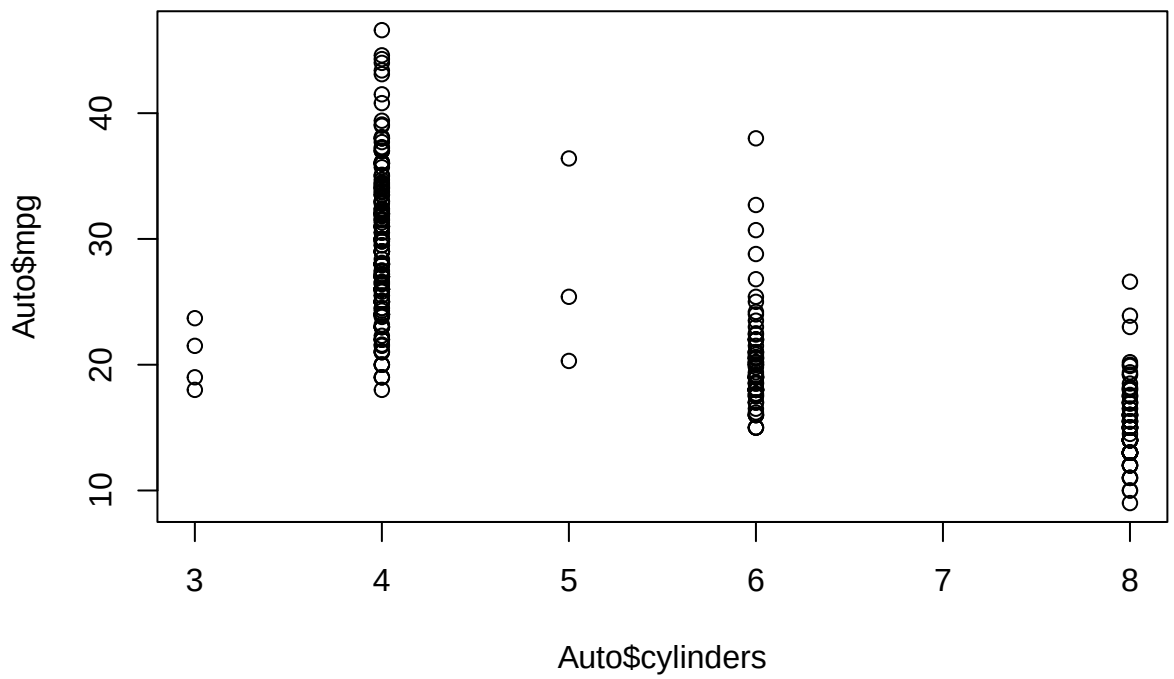
```
Auto <- na.omit(Auto)
dim(Auto)
```

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

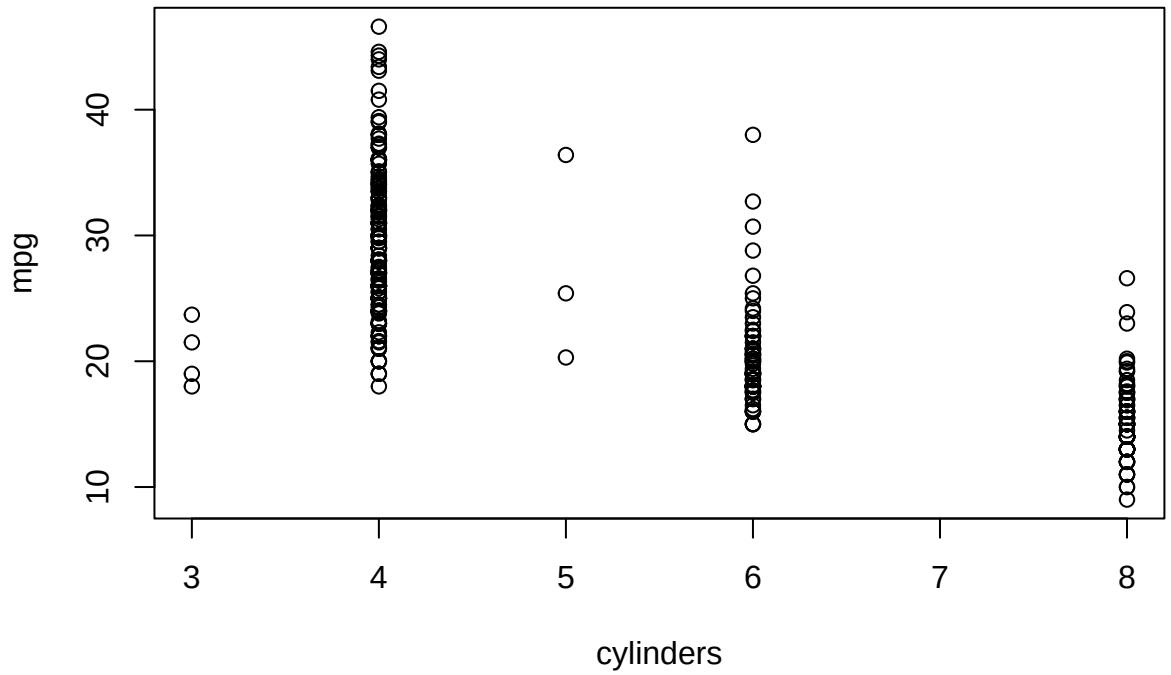
```
names(Auto)
```

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

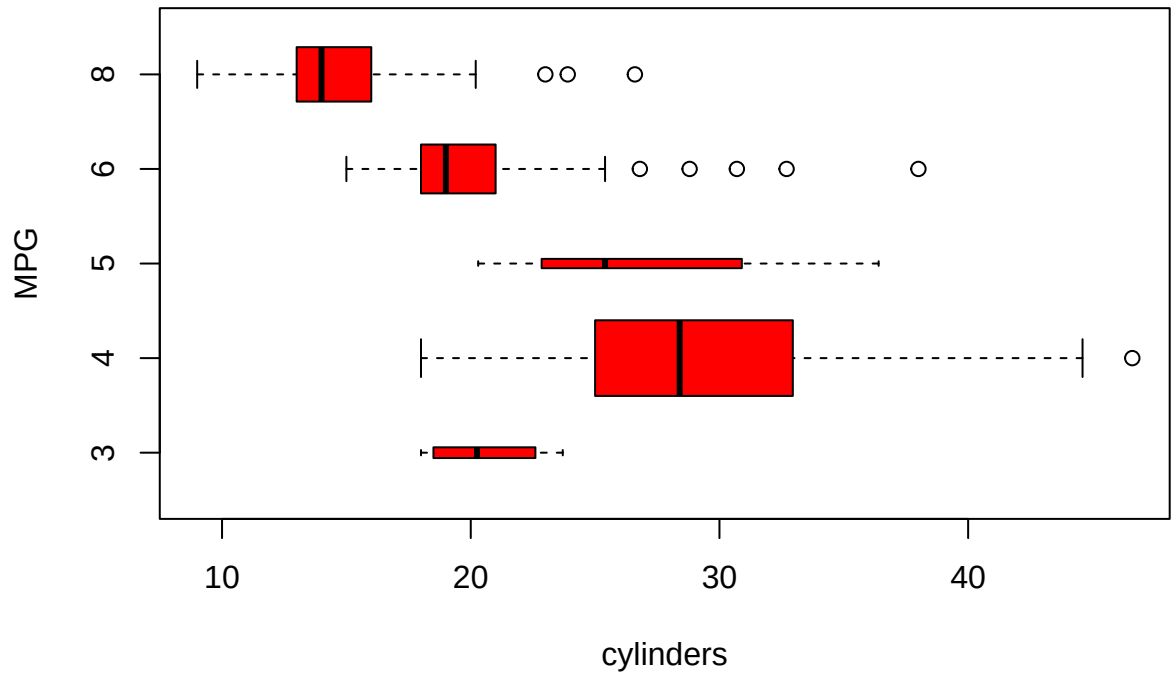
```
# Additional Graphical and Numerical Summaries
plot(Auto$cylinders, Auto$mpg)
```



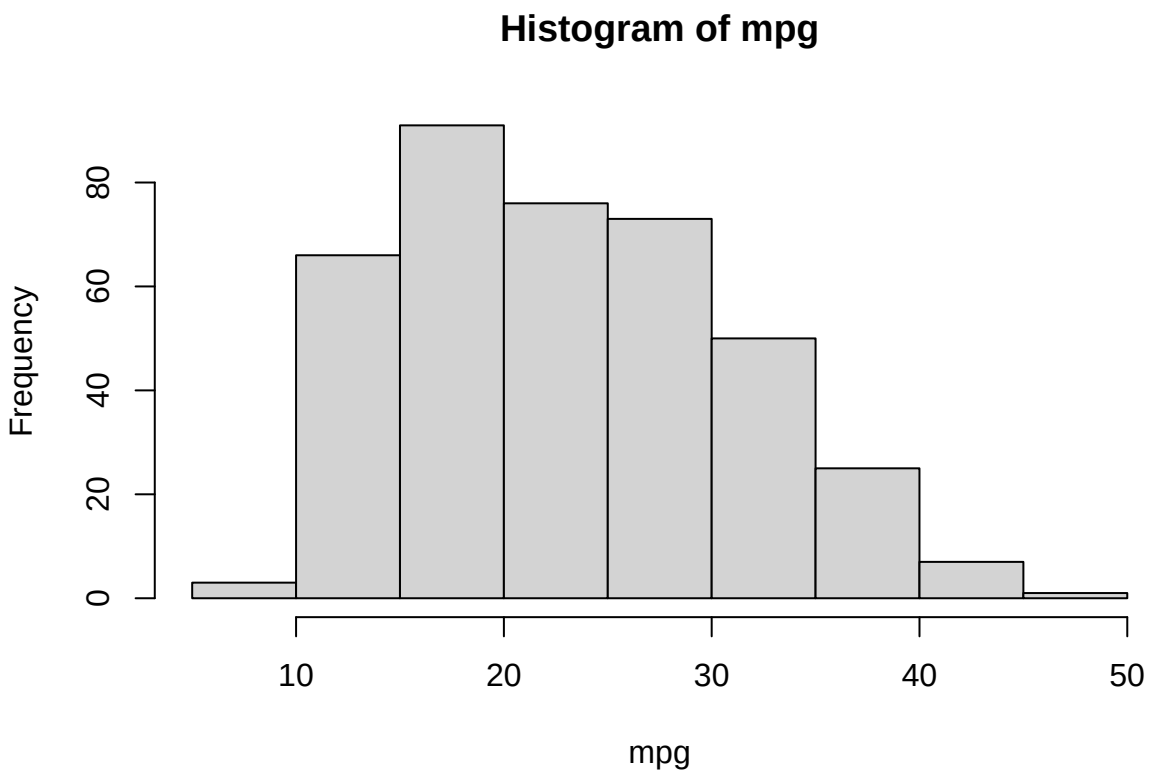

```
attach(Auto)
plot(cylinders, mpg)
```



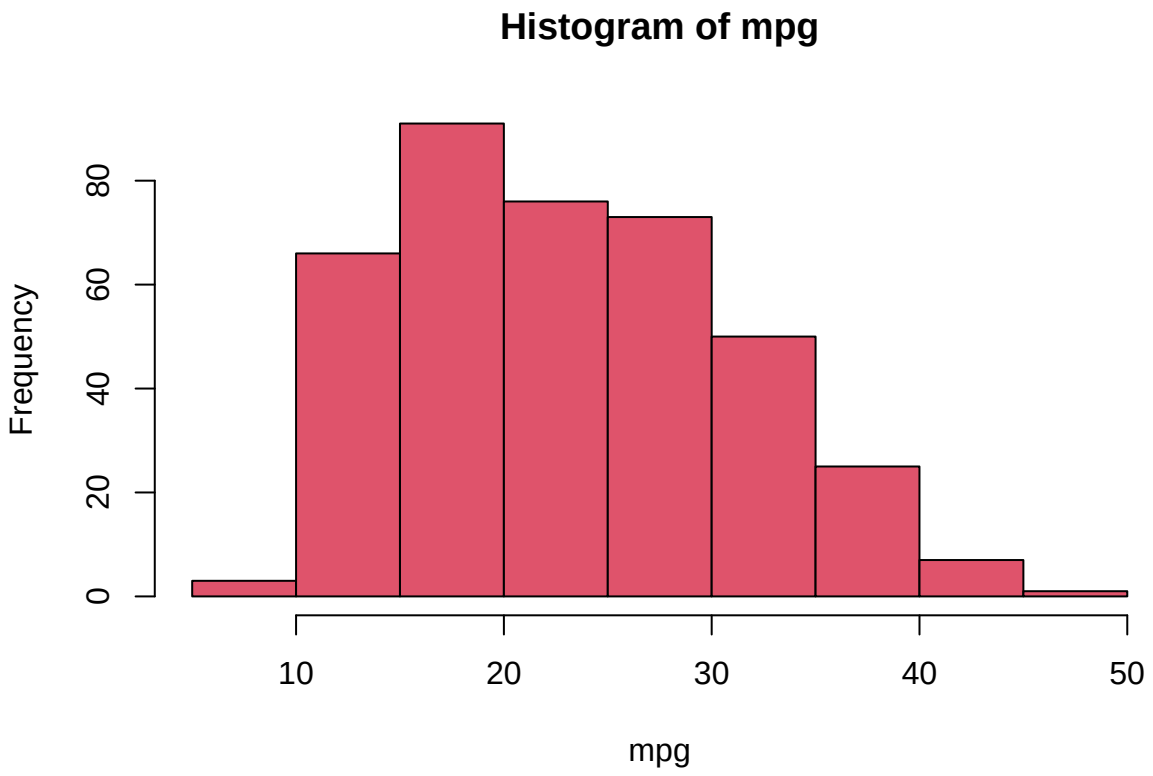

```
cylinders <- as.factor(cylinders)
plot(cylinders, mpg, col = "red", varwidth = TRUE, horizontal = TRUE, xlab = "cylinders", ylab = "MPG")
```



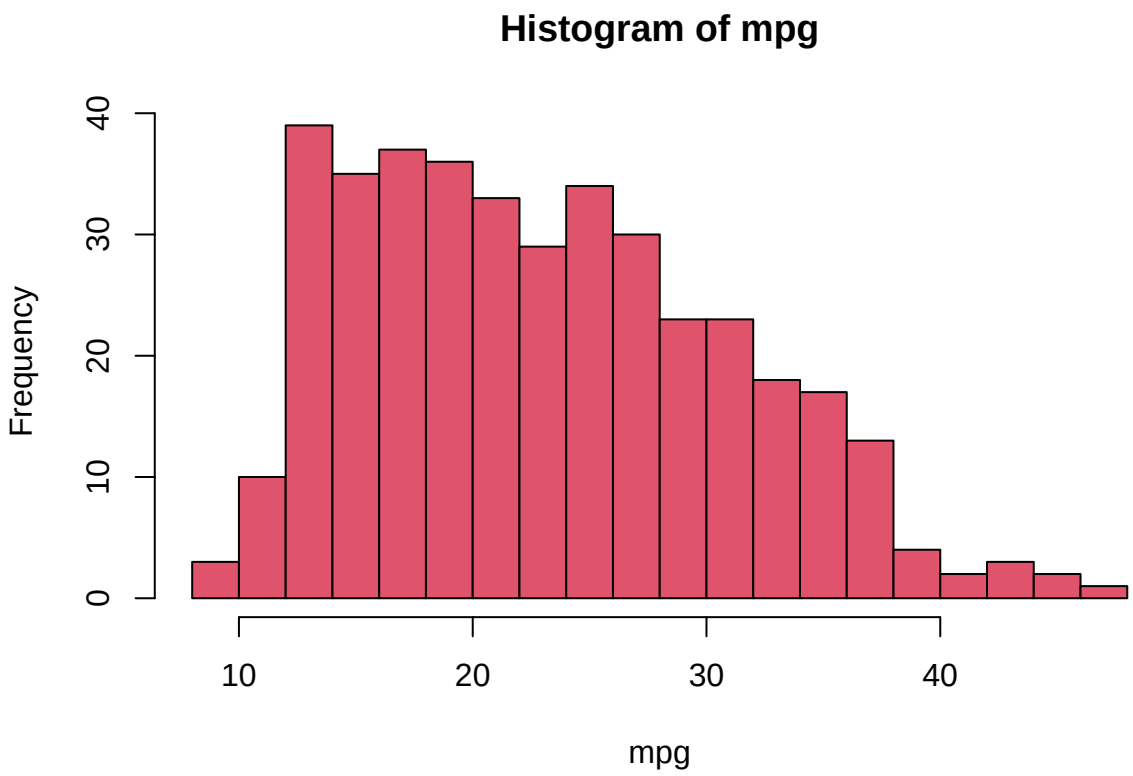

```
hist(mpg)
```



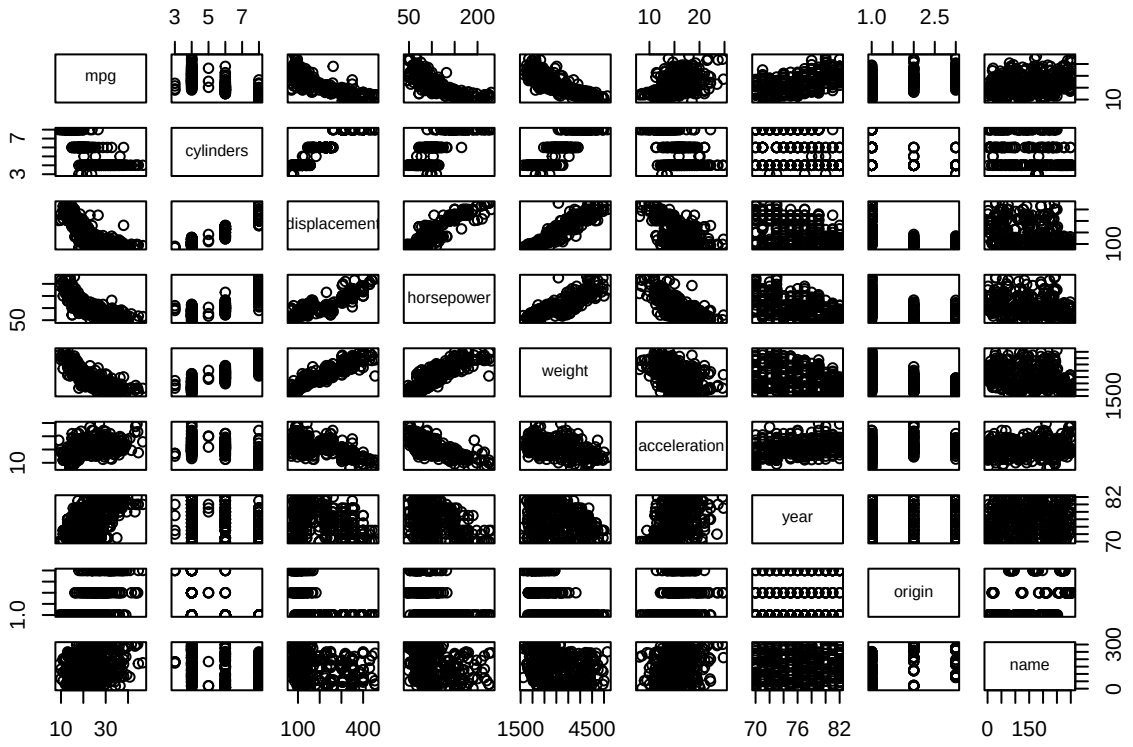

```
hist(mpg, col = 2)
```



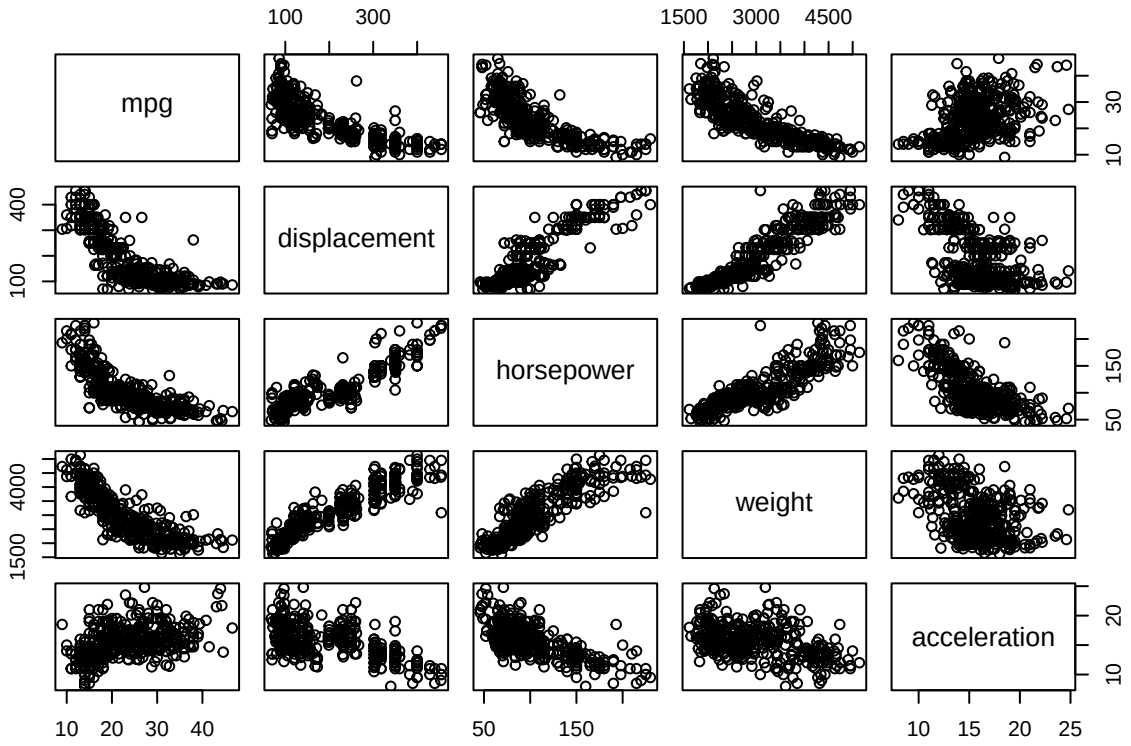

```
hist(mpg, col = 2, breaks = 15)
```



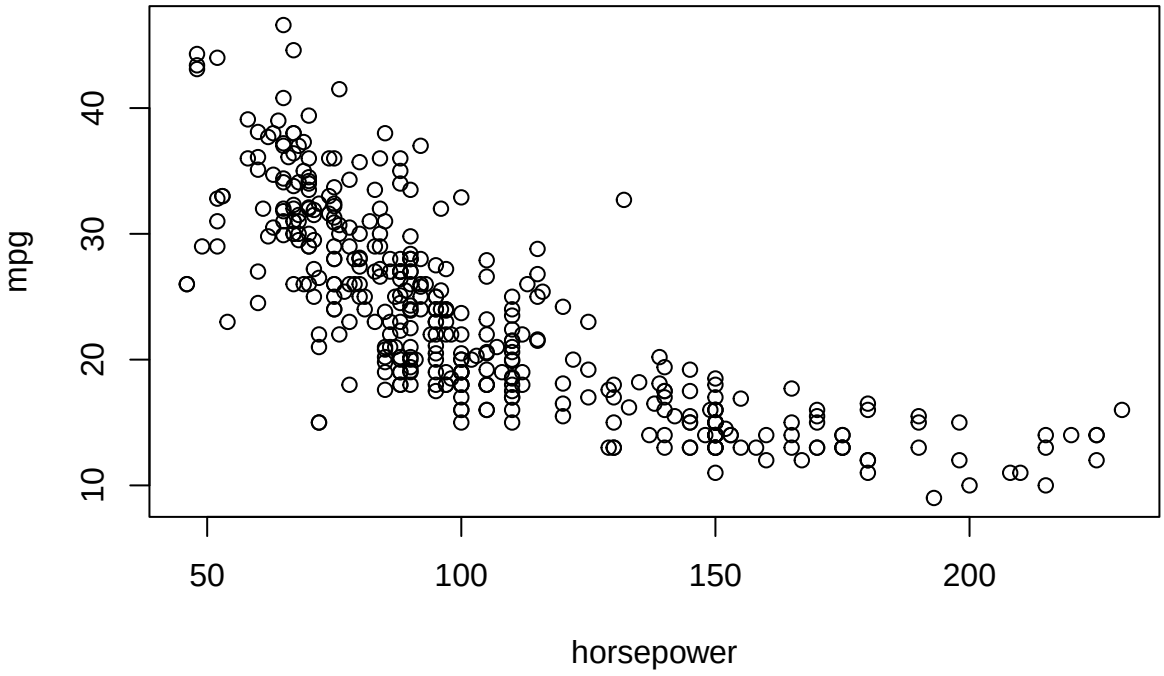

```
pairs(Auto)
```




```
pairs(~ mpg + displacement + horsepower + weight + acceleration, data = Auto)
```




```
plot(horsepower, mpg)
identify(horsepower, mpg, name)
```




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

```
summary(Auto)
```

```
##       mpg          cylinders      displacement     horsepower        weight
##  Min.   : 9.00   Min.   :3.000   Min.   : 68.0   Min.   : 46.0   Min.   :1613
##  1st Qu.:17.00   1st Qu.:4.000   1st Qu.:105.0   1st Qu.: 75.0   1st Qu.:2225
```

```
##  Median :22.75   Median :4.000   Median :151.0   Median : 93.5   Median :2804
##  Mean   :23.45   Mean   :5.472   Mean   :194.4   Mean   :104.5   Mean   :2978
##  3rd Qu.:29.00   3rd Qu.:8.000   3rd Qu.:275.8   3rd Qu.:126.0   3rd Qu.:3615
##  Max.   :46.60   Max.   :8.000   Max.   :455.0   Max.   :230.0   Max.   :5140
##
##   acceleration        year           origin                      name
##  Min.   : 8.00   Min.   :70.00   Min.   :1.000   amc matador       :  5
##  1st Qu.:13.78   1st Qu.:73.00   1st Qu.:1.000   ford pinto        :  5
##  Median :15.50   Median :76.00   Median :1.000   toyota corolla    :  5
##  Mean   :15.54   Mean   :75.98   Mean   :1.577   amc gremlin       :  4
##  3rd Qu.:17.02   3rd Qu.:79.00   3rd Qu.:2.000   amc hornet        :  4
##  Max.   :24.80   Max.   :82.00   Max.   :3.000   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
```

STAT 702 - Homework 1 - 8 & 10

Noah Javadi

2025-02-02

Setup

Problem 8 Problem 8a - Read.csv

```
College <- read.csv("C:/Users/noahj/OneDrive/Documents/SDSU/2024-2025/Spring 2025/STAT 702 - Data Mining
College$Private <- as.factor(College$Private)
```

Problem 8b - View()

```
rownames(College) <- College[, 1]
View(College)
College <- College[, -1]
View(College)
```

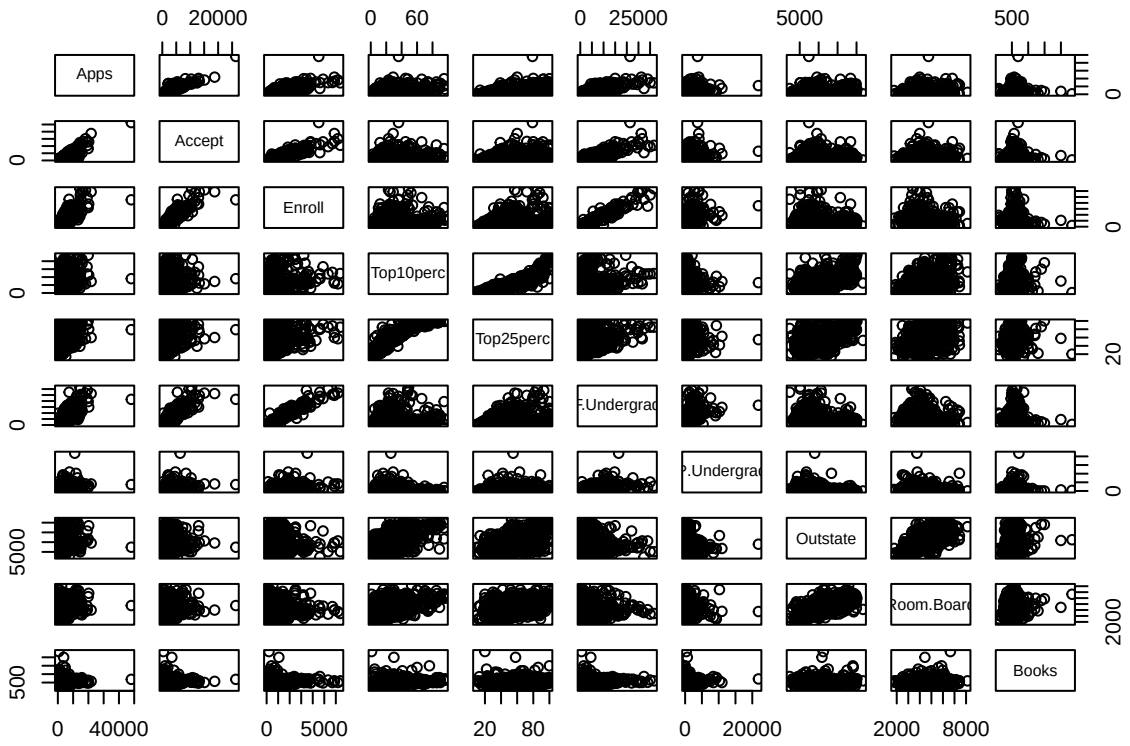
Problem 8c [i - vi] - summary() adjusted due to first row containing Yes/No data

```
summary(College)
```

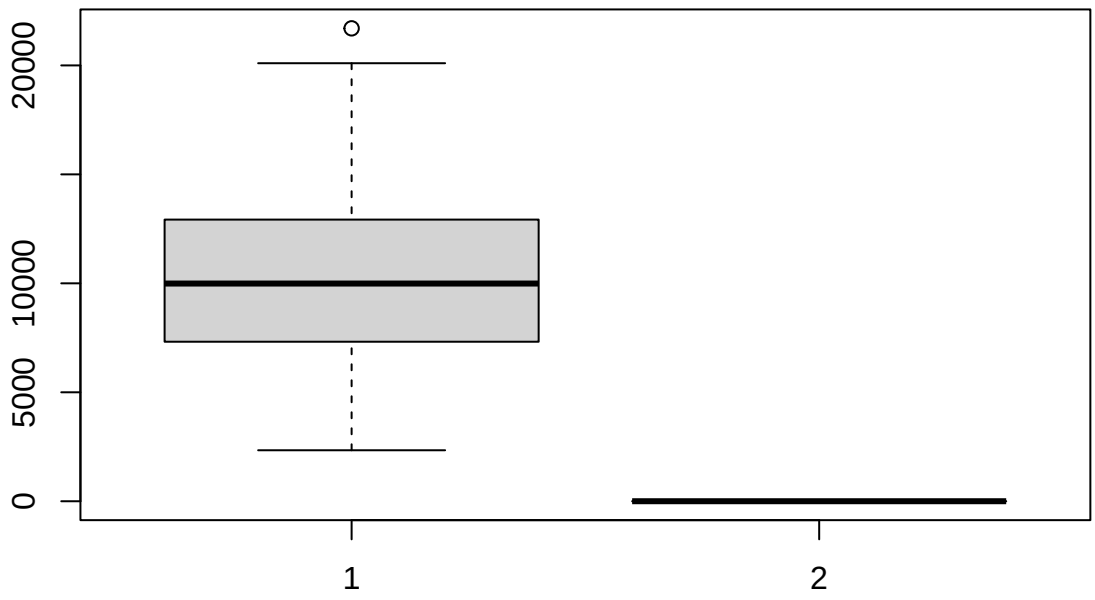
```
##  Private        Apps           Accept          Enroll       Top10perc
##  No :212   Min.   :   81   Min.   :   72   Min.   :  35   Min.   : 1.00
##  Yes:565   1st Qu.:  776   1st Qu.:  604   1st Qu.: 242   1st Qu.:15.00
##            Median : 1558   Median : 1110   Median : 434   Median :23.00
##            Mean   : 3002   Mean   : 2019   Mean   : 780   Mean   :27.56
##            3rd Qu.: 3624   3rd Qu.: 2424   3rd Qu.: 902   3rd Qu.:35.00
##            Max.   :48094   Max.   :26330   Max.   :6392   Max.   :96.00
##    Top25perc      F.Undergrad     P.Undergrad         Outstate
##  Min.   :  9.0   Min.   :  139   Min.   :    1.0   Min.   : 2340
##  1st Qu.: 41.0   1st Qu.:  992   1st Qu.:   95.0   1st Qu.: 7320
##  Median : 54.0   Median : 1707   Median :  353.0   Median : 9990
##  Mean   : 55.8   Mean   : 3700   Mean   :  855.3   Mean   :10441
##  3rd Qu.: 69.0   3rd Qu.: 4005   3rd Qu.:  967.0   3rd Qu.:12925
##  Max.   :100.0   Max.   :31643   Max.   :21836.0   Max.   :21700
##    Room.Board       Books           Personal         PhD
##  Min.   :1780   Min.   :  96.0   Min.   : 250   Min.   :  8.00
##  1st Qu.:3597   1st Qu.: 470.0   1st Qu.: 850   1st Qu.: 62.00
##  Median :4200   Median : 500.0   Median :1200   Median : 75.00
##  Mean   :4358   Mean   : 549.4   Mean   :1341   Mean   : 72.66
##  3rd Qu.:5050   3rd Qu.: 600.0   3rd Qu.:1700   3rd Qu.: 85.00
##  Max.   :8124   Max.   :2340.0   Max.   :6800   Max.   :103.00
##     Terminal       S.F.Ratio      perc.alumni        Expend
##  Min.   : 24.0   Min.   : 2.50   Min.   : 0.00   Min.   : 3186
```

```
##  1st Qu.: 71.0   1st Qu.:11.50   1st Qu.:13.00   1st Qu.: 6751
##  Median : 82.0   Median :13.60   Median :21.00   Median : 8377
##  Mean   : 79.7   Mean   :14.09   Mean   :22.74   Mean   : 9660
##  3rd Qu.: 92.0   3rd Qu.:16.50   3rd Qu.:31.00   3rd Qu.:10830
##  Max.   :100.0   Max.   :39.80   Max.   :64.00   Max.   :56233
##    Grad.Rate
##  Min.   : 10.00
##  1st Qu.: 53.00
##  Median : 65.00
##  Mean   : 65.46
##  3rd Qu.: 78.00
##  Max.   :118.00
```

```
pairs(College[,2:11])
```




```
boxplot(College$Outstate,College$Private)
```

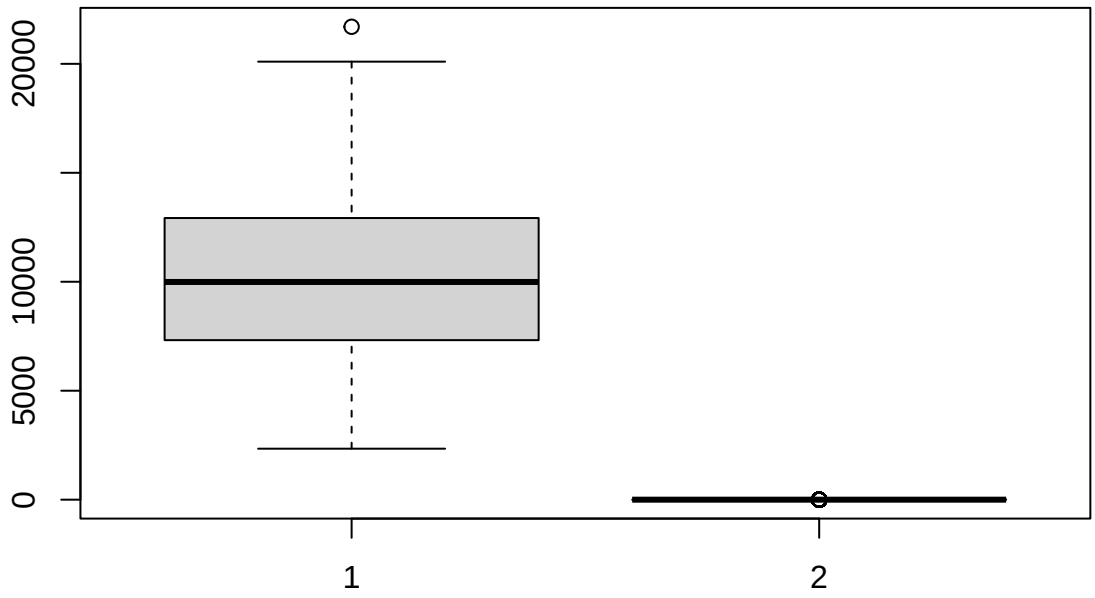



```
College$Elite <- rep("No", nrow(College))
College$Elite[College$Top10perc > 50] <- "Yes"
College$Elite <- as.factor(College$Elite)
College <- data.frame(College, College$Elite)
summary(College)
```

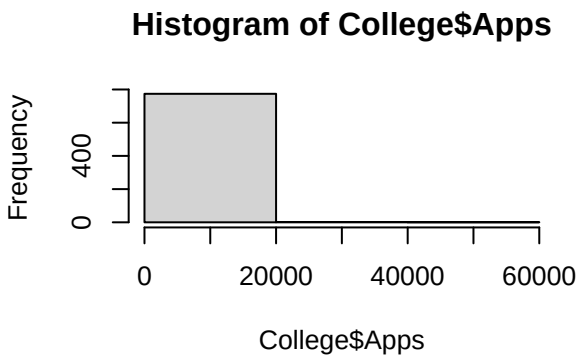
```
##  Private        Apps           Accept          Enroll       Top10perc
##  No :212   Min.   :   81   Min.   :   72   Min.   :  35   Min.   : 1.00
##  Yes:565   1st Qu.:  776   1st Qu.:  604   1st Qu.: 242   1st Qu.:15.00
##            Median : 1558   Median : 1110   Median : 434   Median :23.00
##            Mean   : 3002   Mean   : 2019   Mean   : 780   Mean   :27.56
##            3rd Qu.: 3624   3rd Qu.: 2424   3rd Qu.: 902   3rd Qu.:35.00
##            Max.   :48094   Max.   :26330   Max.   :6392   Max.   :96.00
##    Top25perc      F.Undergrad     P.Undergrad         Outstate
##  Min.   :  9.0   Min.   :  139   Min.   :    1.0   Min.   : 2340
##  1st Qu.: 41.0   1st Qu.:  992   1st Qu.:   95.0   1st Qu.: 7320
##  Median : 54.0   Median : 1707   Median :  353.0   Median : 9990
##  Mean   : 55.8   Mean   : 3700   Mean   :  855.3   Mean   :10441
##  3rd Qu.: 69.0   3rd Qu.: 4005   3rd Qu.:  967.0   3rd Qu.:12925
##  Max.   :100.0   Max.   :31643   Max.   :21836.0   Max.   :21700
##    Room.Board       Books           Personal         PhD
##  Min.   :1780   Min.   :  96.0   Min.   : 250   Min.   :  8.00
##  1st Qu.:3597   1st Qu.: 470.0   1st Qu.: 850   1st Qu.: 62.00
##  Median :4200   Median : 500.0   Median :1200   Median : 75.00
##  Mean   :4358   Mean   : 549.4   Mean   :1341   Mean   : 72.66
```

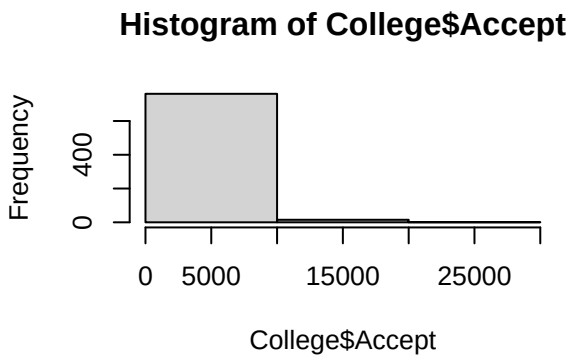
```
##  3rd Qu.:5050   3rd Qu.: 600.0   3rd Qu.:1700   3rd Qu.: 85.00
##  Max.   :8124   Max.   :2340.0   Max.   :6800   Max.   :103.00
##     Terminal       S.F.Ratio      perc.alumni        Expend
##  Min.   : 24.0   Min.   : 2.50   Min.   : 0.00   Min.   : 3186
##  1st Qu.: 71.0   1st Qu.:11.50   1st Qu.:13.00   1st Qu.: 6751
##  Median : 82.0   Median :13.60   Median :21.00   Median : 8377
##  Mean   : 79.7   Mean   :14.09   Mean   :22.74   Mean   : 9660
##  3rd Qu.: 92.0   3rd Qu.:16.50   3rd Qu.:31.00   3rd Qu.:10830
##  Max.   :100.0   Max.   :39.80   Max.   :64.00   Max.   :56233
##    Grad.Rate      Elite     College.Elite
##  Min.   : 10.00   No :699   No :699
##  1st Qu.: 53.00   Yes: 78   Yes: 78
##  Median : 65.00
##  Mean   : 65.46
##  3rd Qu.: 78.00
##  Max.   :118.00
```

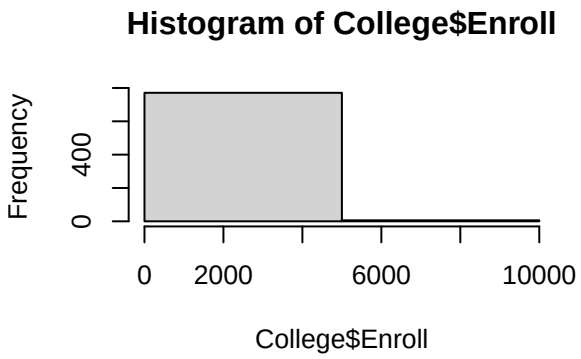
```
boxplot(College$Outstate,College$Elite)
```

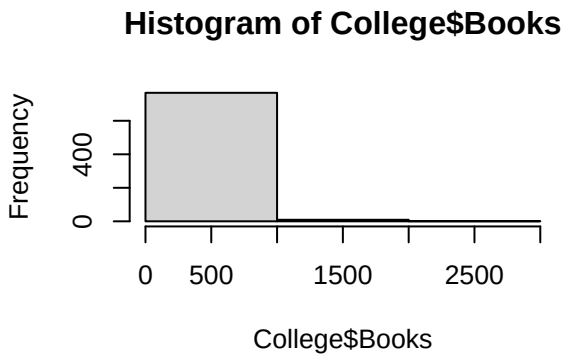



```
par(mfrow = c(2,2))
hist(College$Apps,breaks = length(levels(College$Elite)))
hist(College$Accept,breaks = length(levels(College$Elite)))
hist(College$Enroll,breaks = length(levels(College$Elite)))
hist(College$Books,breaks = length(levels(College$Elite)))
```

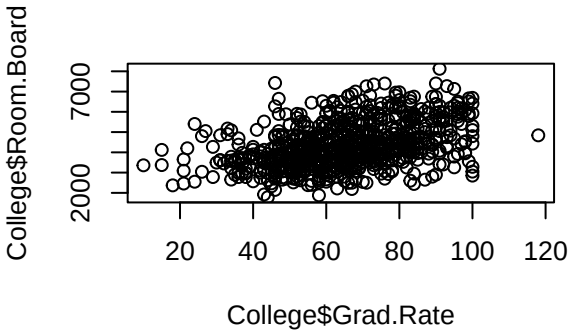


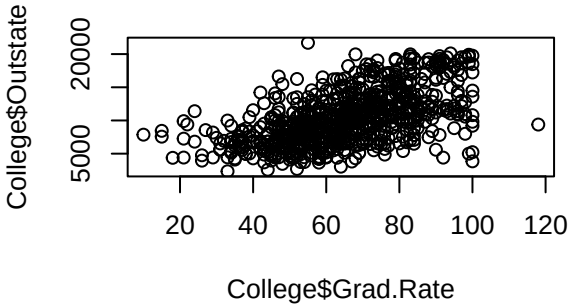


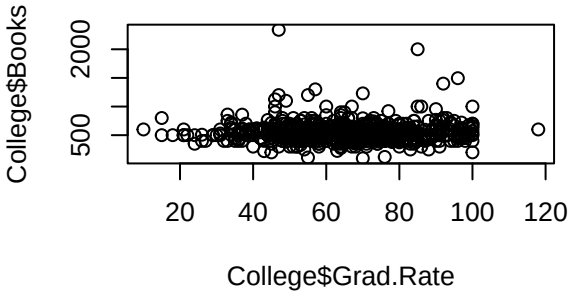


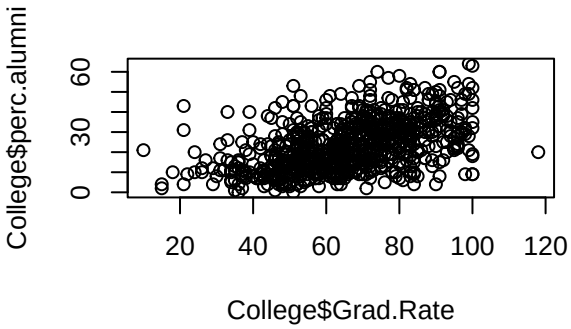



```
plot(College$Grad.Rate,College$Room.Board)
plot(College$Grad.Rate,College$Outstate)
plot(College$Grad.Rate,College$Books)
plot(College$Grad.Rate,College$perc.alumni)
```

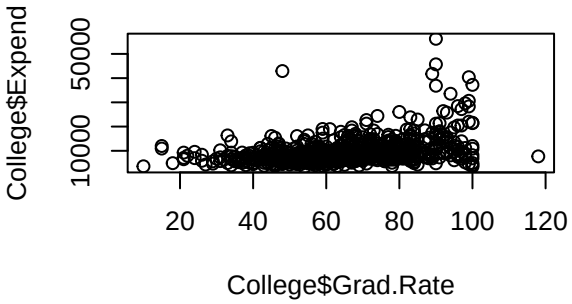









```
plot(College$Grad.Rate,College$Expend)
```



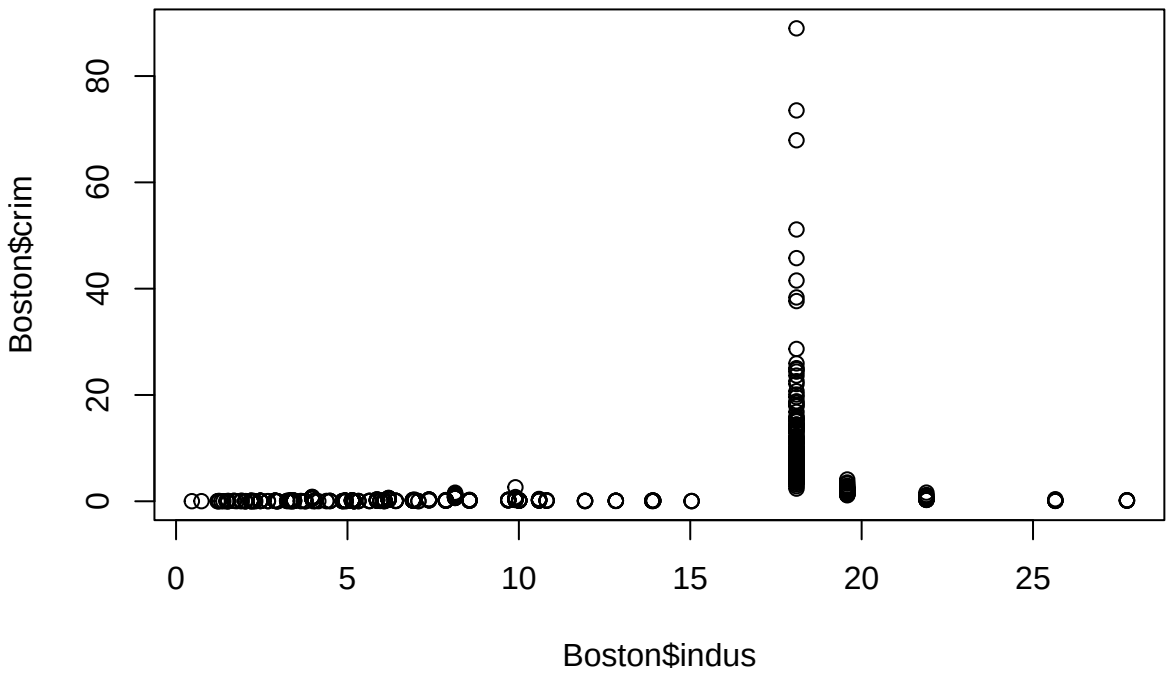
Problem 8d - Summary Plotting graduation rate against Room/Board, Out of state tuition, or alumni donation rates show a positive linear relationship. Suggesting these factors contribute to increase in graduation rates of students. However, there seems to be a little to no relationship between estimated book cost and instructional expenditures per student.

Problem 10

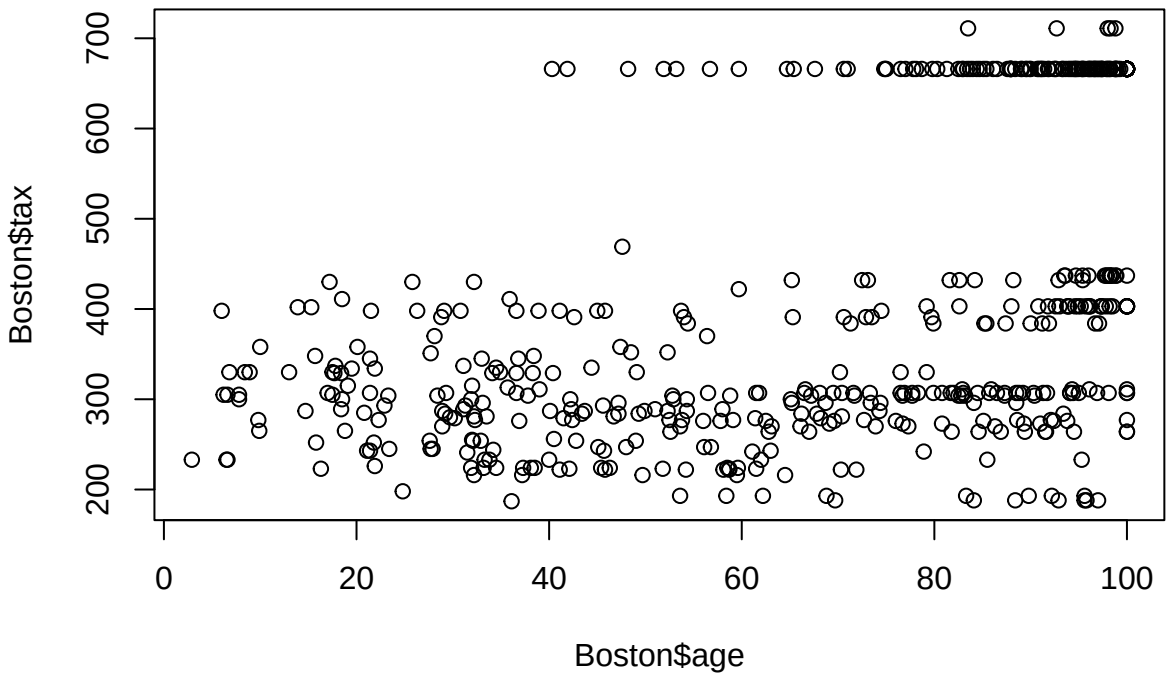
Problem 10a - 506 rows and 13 columns. Each row and column represent housing values and statistics for 506 suburbs in Boston.

Problem 10b - A particular proportion of non-retail business acres per town has a particularly wide range of crime per capita and the maximum values in crime rate per capita. Additionally, there appears to be a consistent band of property tax rates across age groups. However, after 40 years of age there seems to be a group of people who are in a higher tax rate primarily concentrated in the 80+ age group.

```
plot(Boston$indus,Boston$crim)
```




```
plot(Boston$age,Boston$tax)
```

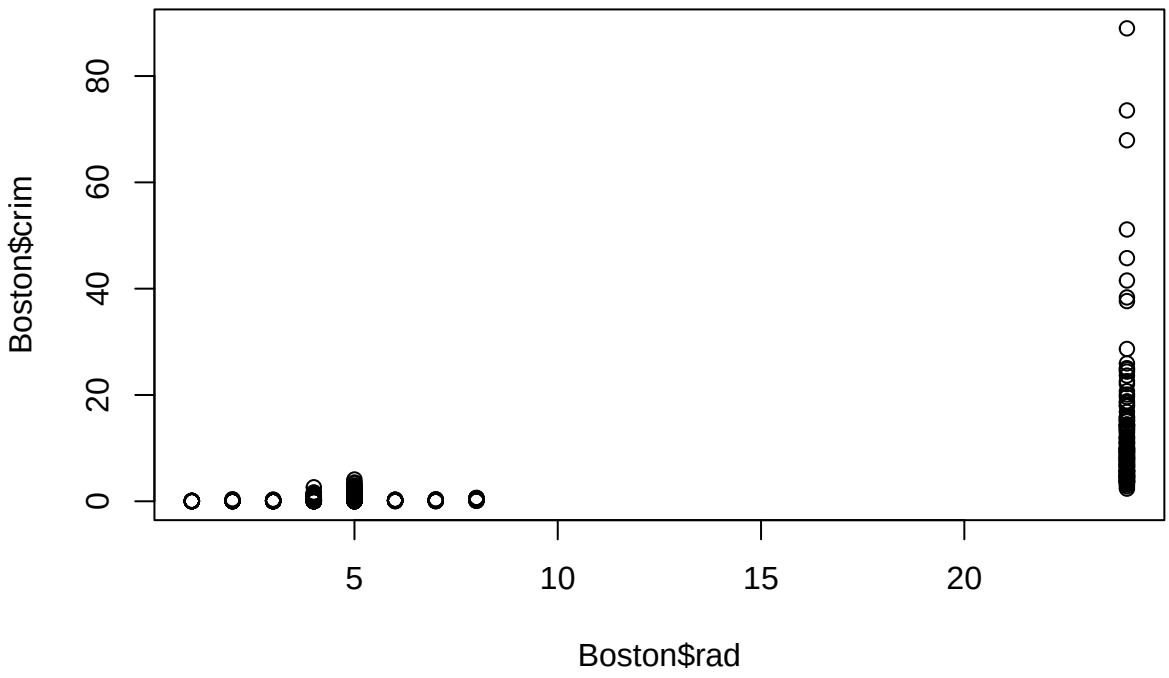


Problem 10c - The index of accessibility to radial highways seems to be correlated to the per capita crime rate by town. However, when plotted rad and crim do not seem to have much of a relationship.

```
C1 <- cor(Boston)
symnum(C1)
```

```
##         cr z i ch n rm a d rd t p l m
## crim    1
## zn         1
## indus   .  . 1
## chas          1
## nox     .  . ,    1
## rm         . .    . 1
## age     .  . ,    ,    1
## dis     .  , ,    ,    , 1
## rad     ,  . .    ,    . . 1
## tax     .  . ,    ,    . . *  1
## ptratio    . .      .      .  . 1
## lstat   .  . ,    . ,  , . .  . . 1
## medv    .  . .    . ,  .   .  . . , 1
## attr(,"legend")
## [1] 0 ' ' 0.3 '.' 0.6 ',' 0.8 '+' 0.9 '*' 0.95 'B' 1
```

```
plot(Boston$rad,Boston$crim)
```



Problem 10d - The top 11 towns have a per capita crime rate of at least 25. There is a distinct cut off in the tax rate band between 470 and 666 where there are no data points, and there are 137 at or above the 666 tax rate. There appears to be 2 towns with particularly high pupil-teacher ratios. The range of crim (per capita crime rate) is from almost 0 to 89. The range of tax (per $10,000) is from 187 to 711. The range of ptratio (pupil-teacher ratio by town) is from 12.6 to 22.

```
summary(Boston$crim >= 25)
```

```
##    Mode   FALSE    TRUE
## logical     495      11
```

```
summary(Boston$crim)
```

```
##      Min.   1st Qu.    Median      Mean   3rd Qu.      Max.
##   0.00632   0.08204   0.25651   3.61352   3.67708 88.97620
```

```
summary(Boston$tax >= 666)
```

```
##    Mode   FALSE    TRUE
## logical     369     137
```

```
summary(Boston$tax)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   187.0   279.0   330.0   408.2   666.0   711.0
```

```
summary(Boston$ptratio >= 25)
```

```
##    Mode   FALSE
## logical     506
```

```
summary(Boston$ptratio)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   12.60   17.40   19.05   18.46   20.20   22.00
```

Problem 10e - 35 census tracts bound the Charles River

```
summary(Boston$chas == 1)
```

```
##    Mode   FALSE    TRUE
## logical     471      35
```

Problem 10f - 19.05 is the median value of pupil-teacher ratio

```
median(Boston$ptratio)
```

```
## [1] 19.05
```

Problem 10g - The 5th row (town) has the lowest medv value. Compared to the other values within the data, crime rate, zn, indus, chas, nox, age, rad, tax, and lstat are all at or below the mean and median for their respective factors. Rm, dis, and ptratio are all at or above the median and mean. It is interesting to see that a higher age population also has a higher ptratio, but the data does not specify the types of teachers or pupils so more investigation or detail is needed.

```
Boston[min(Boston$medv),]
```

```
##      crim zn indus chas   nox    rm  age    dis rad tax ptratio lstat medv
## 5 0.06905  0  2.18    0 0.458 7.147 54.2 6.0622   3 222    18.7  5.33 36.2
```

Problem 10h - There are 64 census tracts with more than 7 rooms per dwelling. There are 13 census tracts with more than 8 rooms per dwelling. The census tracts with more than 8 rooms per dwelling all have below the mean crime rates (row 365 is particularly close to the mean, but is still low).

```
nrow(Boston[Boston$rm > 7,])
```

```
## [1] 64
```

```
Boston[Boston$rm > 8,]
```

```
##        crim zn indus chas    nox    rm  age    dis rad tax ptratio lstat medv
## 98  0.12083  0  2.89    0 0.4450 8.069 76.0 3.4952   2 276    18.0  4.21 38.7
## 164 1.51902  0 19.58    1 0.6050 8.375 93.9 2.1620   5 403    14.7  3.32 50.0
## 205 0.02009 95  2.68    0 0.4161 8.034 31.9 5.1180   4 224    14.7  2.88 50.0
```

```
## 225 0.31533  0  6.20    0 0.5040 8.266 78.3 2.8944   8 307    17.4  4.14 44.8
## 226 0.52693  0  6.20    0 0.5040 8.725 83.0 2.8944   8 307    17.4  4.63 50.0
## 227 0.38214  0  6.20    0 0.5040 8.040 86.5 3.2157   8 307    17.4  3.13 37.6
## 233 0.57529  0  6.20    0 0.5070 8.337 73.3 3.8384   8 307    17.4  2.47 41.7
## 234 0.33147  0  6.20    0 0.5070 8.247 70.4 3.6519   8 307    17.4  3.95 48.3
## 254 0.36894 22  5.86    0 0.4310 8.259  8.4 8.9067   7 330    19.1  3.54 42.8
## 258 0.61154 20  3.97    0 0.6470 8.704 86.9 1.8010   5 264    13.0  5.12 50.0
## 263 0.52014 20  3.97    0 0.6470 8.398 91.5 2.2885   5 264    13.0  5.91 48.8
## 268 0.57834 20  3.97    0 0.5750 8.297 67.0 2.4216   5 264    13.0  7.44 50.0
## 365 3.47428  0 18.10    1 0.7180 8.780 82.9 1.9047  24 666    20.2  5.29 21.9
```

```
nrow(Boston[Boston$rm > 8,])
```

```
## [1] 13
```