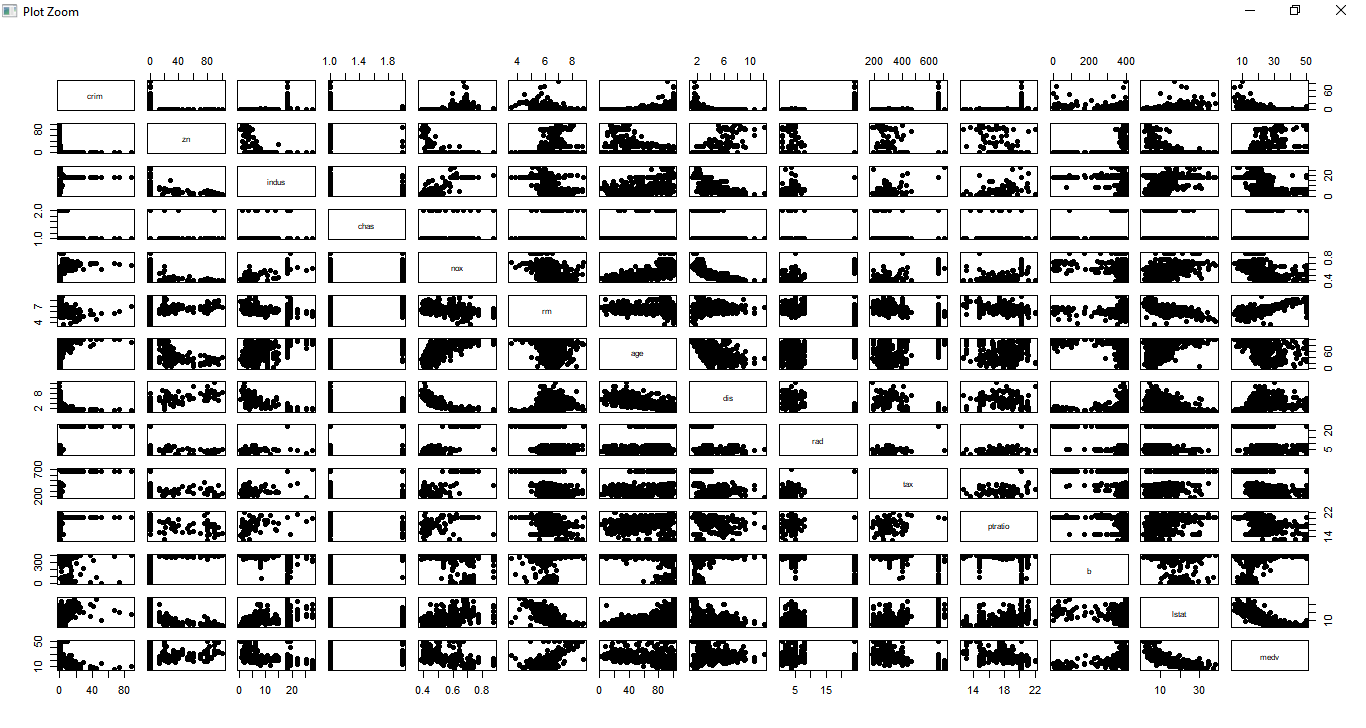
# 1. Create a scatterplot matrix of all variables in the data set. Save your output.

> pairs(BostonHousing[,1:14], pch = 19)



# 2. For each numeric variable in BostonHousing, create a separate boxplot using

# "Method 2" listed in the class notes. Do this programmatically; meaning do

# not simply hardcode the creation of every boxplot. Instead, loop over the

# approriate columns and create the boxplots. Save your output. Ensure your boxplots

# all have proper titles

coltype<-sapply(BostonHousing, class)

> coltype

crim zn indus chas nox rm age dis rad

"numeric" "numeric" "numeric" "factor" "numeric" "numeric" "numeric" "numeric" "numeric"

tax ptratio b lstat medv

"numeric" "numeric" "numeric" "numeric" "numeric"

> for ( i in (1:ncol(BostonHousing))){

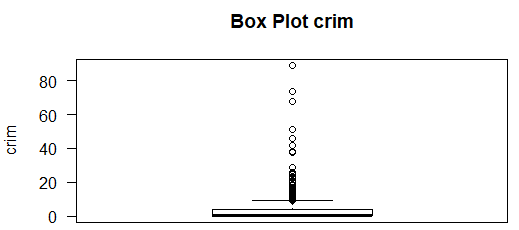
+ if ("numeric" == coltype[i]){

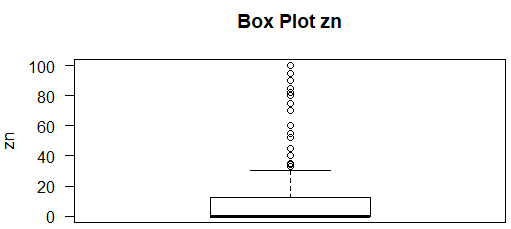
+ #box plot of each columns

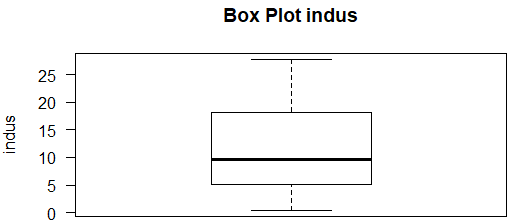
+ boxplot(BostonHousing[i], main =paste("Box Plot",names(BostonHousing[i])), ylab = names(BostonHousing[i]), las=1)

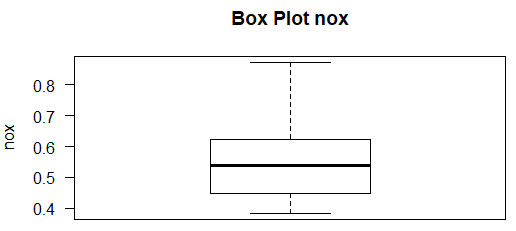
+ }

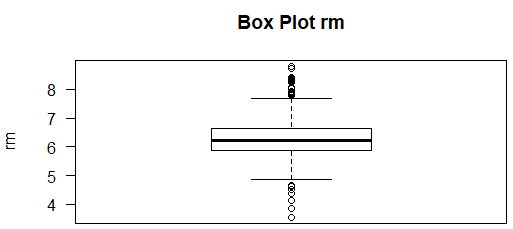
+ }

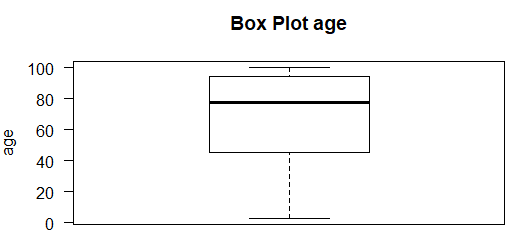


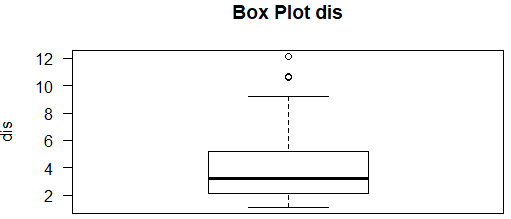


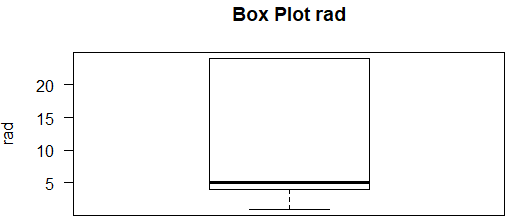


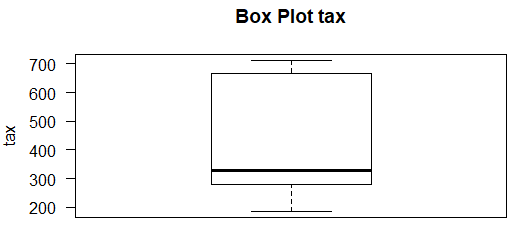


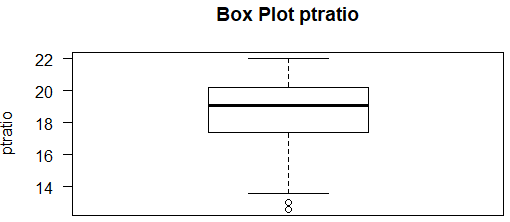


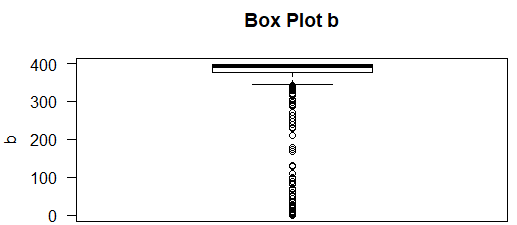


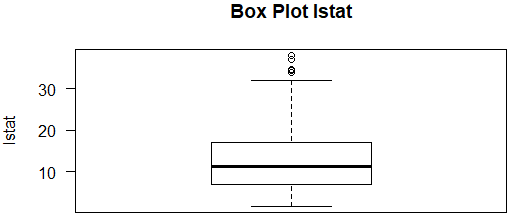


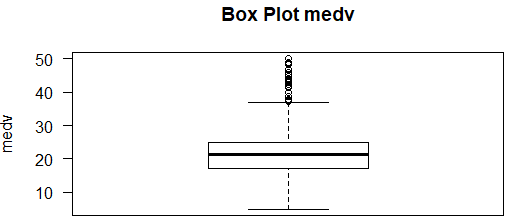












# 3. Create a correlation matrix and correlation plot

# for the BostonHousing data set. Save your output.

> library(corrplot)

> cor\_Mat<-cor(BostonHousing[,sapply(BostonHousing, is.numeric)], method = c("pearson", "kendall", "spearman"))

> cor\_Mat<- round(cor\_Mat,2)

> print(cor\_Mat)

crim zn indus nox rm age dis rad tax ptratio b lstat medv

crim 1.00 -0.20 0.41 0.42 -0.22 0.35 -0.38 0.63 0.58 0.29 -0.39 0.46 -0.39

zn -0.20 1.00 -0.53 -0.52 0.31 -0.57 0.66 -0.31 -0.31 -0.39 0.18 -0.41 0.36

indus 0.41 -0.53 1.00 0.76 -0.39 0.64 -0.71 0.60 0.72 0.38 -0.36 0.60 -0.48

nox 0.42 -0.52 0.76 1.00 -0.30 0.73 -0.77 0.61 0.67 0.19 -0.38 0.59 -0.43

rm -0.22 0.31 -0.39 -0.30 1.00 -0.24 0.21 -0.21 -0.29 -0.36 0.13 -0.61 0.70

age 0.35 -0.57 0.64 0.73 -0.24 1.00 -0.75 0.46 0.51 0.26 -0.27 0.60 -0.38

dis -0.38 0.66 -0.71 -0.77 0.21 -0.75 1.00 -0.49 -0.53 -0.23 0.29 -0.50 0.25

rad 0.63 -0.31 0.60 0.61 -0.21 0.46 -0.49 1.00 0]\.91 0.46 -0.44 0.49 -0.38

tax 0.58 -0.31 0.72 0.67 -0.29 0.51 -0.53 0.91 1.00 0.46 -0.44 0.54 -0.47

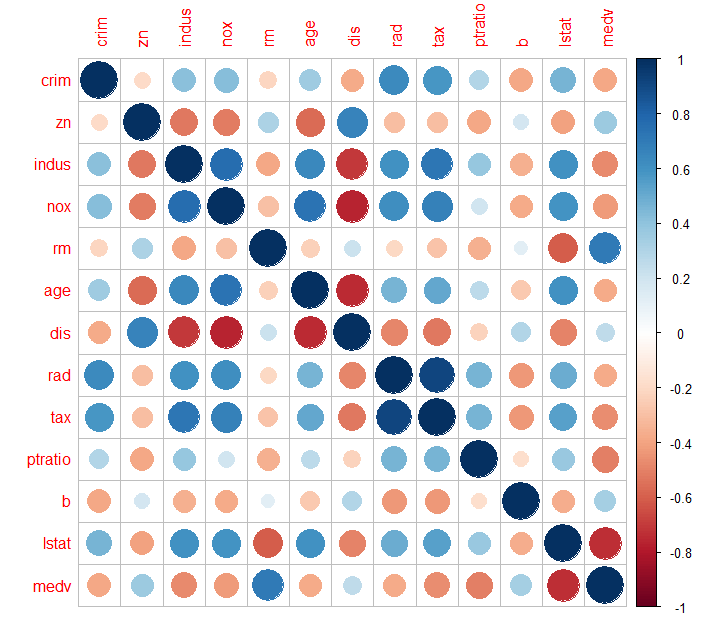
ptratio 0.29 -0.39 0.38 0.19 -0.36 0.26 -0.23 0.46 0.46 1.00 -0.18 0.37 -0.51

b -0.39 0.18 -0.36 -0.38 0.13 -0.27 0.29 -0.44 -0.44 -0.18 1.00 -0.37 0.33

lstat 0.46 -0.41 0.60 0.59 -0.61 0.60 -0.50 0.49 0.54 0.37 -0.37 1.00 -0.74

medv -0.39 0.36 -0.48 -0.43 0.70 -0.38 0.25 -0.38 -0.47 -0.51 0.33 -0.74 1.00

> corrplot(cor\_Mat)



> # 4. Identify the top 3 strongest absolute correlations in the data set. Save your output.

> # because diagonl and half matrix have self correlation cofficient and same correlation cofficient with others

> cor\_Mat[lower.tri(cor\_Mat,diag=TRUE)]<-NA

> cor\_Cof<-as.data.frame(as.table(cor\_Mat))

> #removing NA

> cor\_Cof<-cor\_Cof[complete.cases(cor\_Cof),]

> cor\_Cof<-cor\_Cof[order(abs(cor\_Cof$Freq),decreasing = TRUE),]

> # TOP 3 STRONGEST ABSOLUTE CORRELATION

> cor\_Cof[1:3,]

Var1 Var2 Freq

112 rad tax 0.91

82 nox dis -0.77

42 indus nox 0.76

> # 5. Create a new variable call ageGroup quartiles. Divide the age variable

> # into four even sections and assign it to one quartile.

> BostonHousing$ageGroup<-NULL

> BostonHousing$ageGroup<-cut(BostonHousing$age, breaks = quantile(BostonHousing$age, probs = seq(0, 1, 0.25)), include.lowest = TRUE)

> head(BostonHousing)

crim zn indus chas nox rm age dis rad tax ptratio b lstat medv

1 0.00632 18 2.31 0 0.538 6.575 65.2 4.0900 1 296 15.3 396.90 4.98 24.0

2 0.02731 0 7.07 0 0.469 6.421 78.9 4.9671 2 242 17.8 396.90 9.14 21.6

3 0.02729 0 7.07 0 0.469 7.185 61.1 4.9671 2 242 17.8 392.83 4.03 34.7

4 0.03237 0 2.18 0 0.458 6.998 45.8 6.0622 3 222 18.7 394.63 2.94 33.4

5 0.06905 0 2.18 0 0.458 7.147 54.2 6.0622 3 222 18.7 396.90 5.33 36.2

6 0.02985 0 2.18 0 0.458 6.430 58.7 6.0622 3 222 18.7 394.12 5.21 28.7

ageGroup

1 (45,77.5]

2 (77.5,94.1]

3 (45,77.5]

4 (45,77.5]

5 (45,77.5]

6 (45,77.5]

> # 6. Go to the website listed below. Convert the html table into a

> # dataframe with columns NO, Player, Highlights

> library('rvest')

> library('tidyr')

> url = 'http://www.espn.com/nfl/superbowl/history/mvps'

> my\_df <- as.data.frame(read\_html(url) %>% html\_table(trim = TRUE, fill=TRUE))

> my\_df<-my\_df[-(1:2),]

> names(my\_df)<-c('NO', 'Player', 'Highlights')

> head(my\_df)

NO Player Highlights

3 I Bart Starr, QB, Green Bay Two touchdown passes

4 II Bart Starr, QB, Green Bay 202 yards passing, 1 TD

5 III Joe Namath, QB, New York Jets 206 yards passing

6 IV Len Dawson, QB, Kansas City 142 yards passing, 1 TD

7 V Chuck Howley, LB, Dallas Two interceptions, fumble recovery

8 VI Roger Staubach, QB, Dallas 119 yards passing, 2 TDs

|  |
| --- |
| > # 7.Extract the names of the MVPs, Position and Team into columns  > # MVP1, MVP2, Position, Team  > my\_df<-separate(my\_df, Player, c('MVPs', 'Position', 'Team')  + , sep=', ' # We want to split this where the comma is located  + , remove=TRUE)  > my\_df$MVPs <- ifelse(!grepl('&', my\_df$MVPs), paste0(my\_df$MVPs,"&"), my\_df$MVPs)  > my\_df<-separate(my\_df, MVPs, c('MVP1', 'MVP2')  + , sep='&' # We want to split this where the & is located  + , remove=TRUE)  > #my\_df <- sapply(my\_df, as.character)  > #my\_df[is.na(my\_df)] <- ""  > head(my\_df)  NO MVP1 MVP2 Position Team Highlights  3 I Bart Starr QB Green Bay Two touchdown passes  4 II Bart Starr QB Green Bay 202 yards passing, 1 TD  5 III Joe Namath QB New York Jets 206 yards passing  6 IV Len Dawson QB Kansas City 142 yards passing, 1 TD  7 V Chuck Howley LB Dallas Two interceptions, fumble recovery  8 VI Roger Staubach QB Dallas 119 yards passing, 2 TDs  > print(my\_df[10:15,])  NO MVP1 MVP2 Position Team  12 X Lynn Swann WR Pittsburgh  13 XI Fred Biletnikoff WR Oakland  **14 XII Harvey Martin Randy White** DL Dallas  15 XIII Terry Bradshaw QB Pittsburgh  16 XIV Terry Bradshaw QB Pittsburgh  17 XV Jim Plunkett QB Oakland  Highlights  12 4 catches, 161 yards, 1 TD  13 4 catches, 79 yards  14 Led Dallas defense that forced eight turnovers  15 318 yards passing, 4 TDs  16 309 yards passing, 2 TDs  17 261 yards passing, 3 TDs |
|  |
| |  | | --- | |  | |

# 8. Determine the 90th%, 92.5th%, 95th%, 97.5th% and 99th% confidence intervals# for the mean of passing yards

# (as listed in "Highlights" column) for quarterbacks.

> #quarterbacks = QB in positions.

> df\_QB <- subset(my\_df, Position == 'QB')

> split\_Higlights<-unlist(strsplit(df\_QB$Highlights, " "))

> yards\_values<-NULL

> for (i in 1:length(split\_Higlights)){

+ if (grepl("yards",split\_Higlights[i])){

+ yards\_values<-append(yards\_values,split\_Higlights[i-1])

+ }

+ }

> print(yards\_values)

[1] "202" "206" "142" "119" "318" "309" "261" "157" "331" "268" "340" "297" "292" "273"

[15] "325" "336" "414" "145" "354" "247" "255" "288" "304" "296" "287" "328" "466" "373"

[29] "286"

> # confidence intervals

> t.test(as.numeric(yards\_values), conf.level = 0.9)

One Sample t-test

data: as.numeric(yards\_values)

t = 19.259, df = 28, p-value < 2.2e-16

alternative hypothesis: true mean is not equal to 0

90 percent confidence interval:

**258.3797 308.4479**

sample estimates:

mean of x

283.4138

> t.test(as.numeric(yards\_values), conf.level = 0.925)

One Sample t-test

data: as.numeric(yards\_values)

t = 19.259, df = 28, p-value < 2.2e-16

alternative hypothesis: true mean is not equal to 0

92.5 percent confidence interval:

**256.1994 310.6282**

sample estimates:

mean of x

283.4138

> t.test(as.numeric(yards\_values), conf.level = 0.95)

One Sample t-test

data: as.numeric(yards\_values)

t = 19.259, df = 28, p-value < 2.2e-16

alternative hypothesis: true mean is not equal to 0

95 percent confidence interval:

**253.2691 313.5585**

sample estimates:

mean of x

283.4138

> t.test(as.numeric(yards\_values), conf.level = 0.975)

One Sample t-test

data: as.numeric(yards\_values)

t = 19.259, df = 28, p-value < 2.2e-16

alternative hypothesis: true mean is not equal to 0

97.5 percent confidence interval:

**248.5593 318.2683**

sample estimates:

mean of x

283.4138

> t.test(as.numeric(yards\_values), conf.level = 0.99)

One Sample t-test

data: as.numeric(yards\_values)

t = 19.259, df = 28, p-value < 2.2e-16

alternative hypothesis: true mean is not equal to 0

99 percent confidence interval:

**242.7492 324.0784**

sample estimates:

mean of x

283.4138

> # 9. The following contains data on the calorie counts of four types

> # of foods. Perform an ANOVA and determine the Pr(>F)

> food1 <- c(164, 172, 168, 177, 156, 195)

> food2 <- c(178, 191, 197, 182, 185, 177)

> food3 <- c(175, 193, 178, 171, 163, 176)

> food4 <- c(155, 166, 149, 164, 170, 168)

> food\_df <- data.frame(food1,food2,food3,food4)

> food\_df <- stack(food\_df)

> anova<-aov(food\_df$values ~ food\_df$ind, food\_df)

> print(anova)

Call:

aov(formula = food\_df$values ~ food\_df$ind, data = food\_df)

Terms:

food\_df$ind Residuals

Sum of Squares 1636.5 2018.0

Deg. of Freedom 3 20

Residual standard error: 10.0449

Estimated effects may be unbalanced

> summary(anova)

Df Sum Sq Mean Sq F value Pr(>F)

food\_df$ind 3 1636 545.5 5.406 0.00688 \*\*

Residuals 20 2018 100.9

---

Signif. codes: 0 ‘\*\*\*’ 0.001 ‘\*\*’ 0.01 ‘\*’ 0.05 ‘.’ 0.1 ‘ ’ 1

# 10. Determine how many

# Tuesdays fell on the first of the month

# during the 19th century (1 Jan 1801 to 31 Dec 1901).

install.packages("lubridate")

> library(lubridate)

> beginDate<-as.Date("1801-01-01")

> endDate<-as.Date("1901-12-31")

> sum(wday(seq(beginDate,endDate,"months"), label = TRUE) == "Tue")

[1] **173**