Bivariate_Analysis

5/4/2019

Introduction to Inferential Statistics

Inferential statistics is used in Hypothesis testing to make inferences (conclusions) about a sample/population and it is applied on Bivariate data.

What is Hypothesis Testing?

Hypothesis testing is used in statistics to determine if hypothesis is false.

there are 2 types of Hypothesis

1) Null Hypothesis

2) Alternative Hypothesis

1) Null Hypothesis says there is no significant difference between the variable of interest/dataset.

2) Alternative Hypothesis says there is a significant differnece between the variable of interest.

We use inferential statistics to see if the null hypothesis is false.

if null hypothesis is false, we reject it, and when we reject the Null Hypothesis we accept the alternative Hypothesis.

Inferential statistics results in p - value.

P - value is a number between 0 and 1, it shows if the the null hypothesis should be rejected or accepted.

if the p - value is <= alpha i.e .05 should reject the Null hypothesis and accept the alternative Hypothesis.

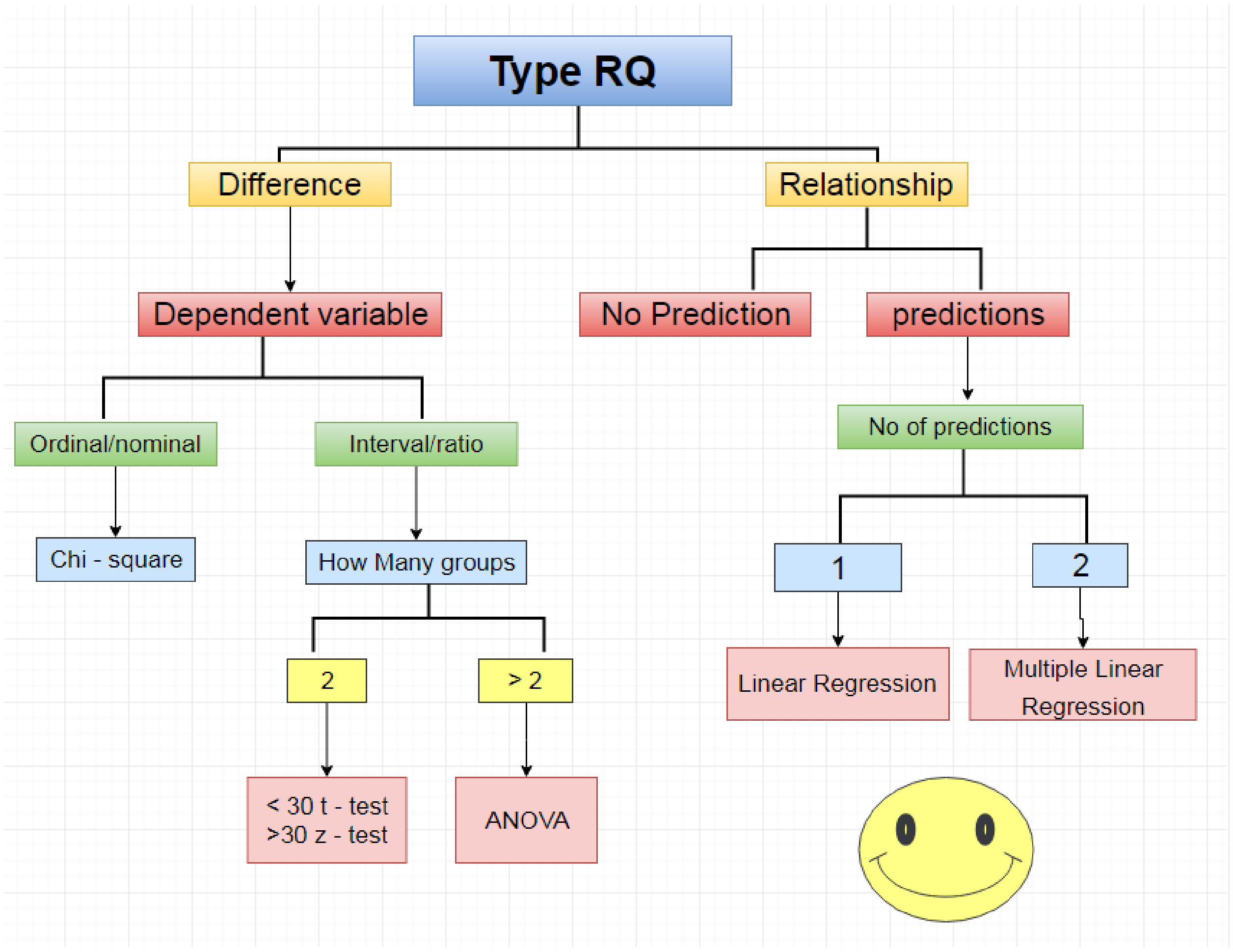
Bivariate Analysis

What is Bivariate Analysis?

Bivariate Analysis is the analysis of two variables, the independent variable should already be nominal/ordinal

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

Here is the Decision Tree of Bivariate Analysis



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

T - TEST

t - test is a statistical test that compares the difference in the mean of the dependent variable.

ANOVA

ANOVA is a statistical test that compares the difference in the mean of the dependent variable of > 2 groups.

Chi-Square :

Chi-square is a statistical test that compares difference between obsereved & expted frequencies.

p-value <= .05 Reject Null Hypothesis

p - value >= .05 Accept Alternative Hypothesis.

One Sample t - test

Conducting One - Sample t-test and constructing One-Sample confidence Interval for the mean.

The one-sample t-test and confidence interval are parametic methods appropriate for examining a single numberic variable . . . .

Parametric and Nonparametric Test

Parametric Test - If the information about the population is completely known by means of its parameters then statistical test is called Parametic test.

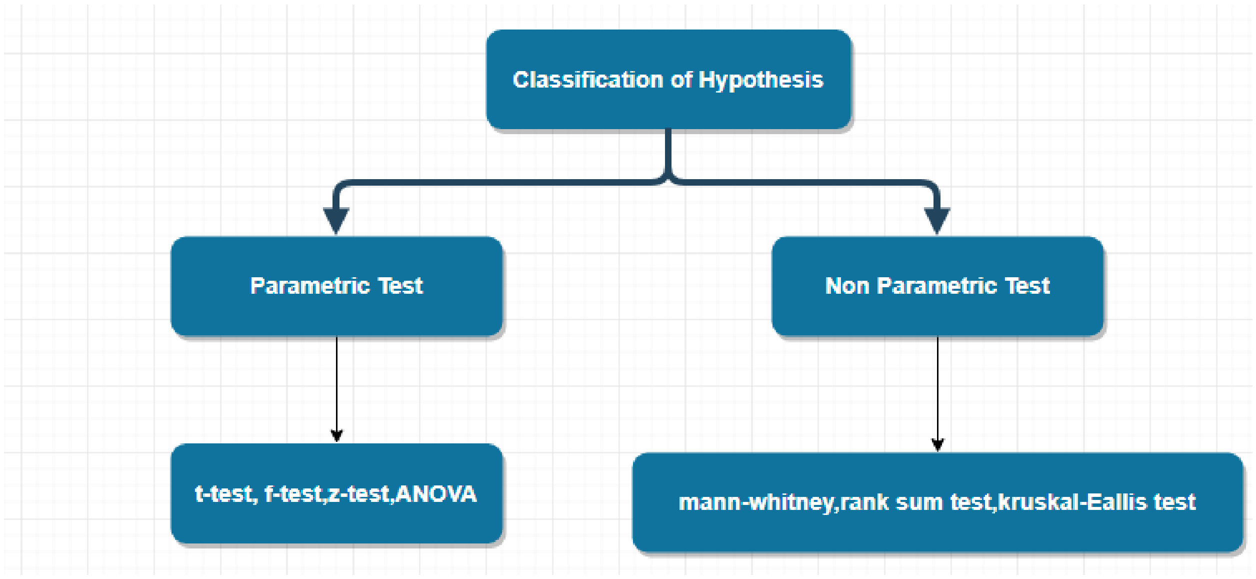
Eg : t-test,f-test,z-test,ANONA

Non-Parametric Test - If there is no knowledge about the population or parameters, but still it is required to test the hypothesis of the population, then it is called non-parametric test.

Eg: Mann-Whiteny, rank sum test, Kruskal-Wallis test.

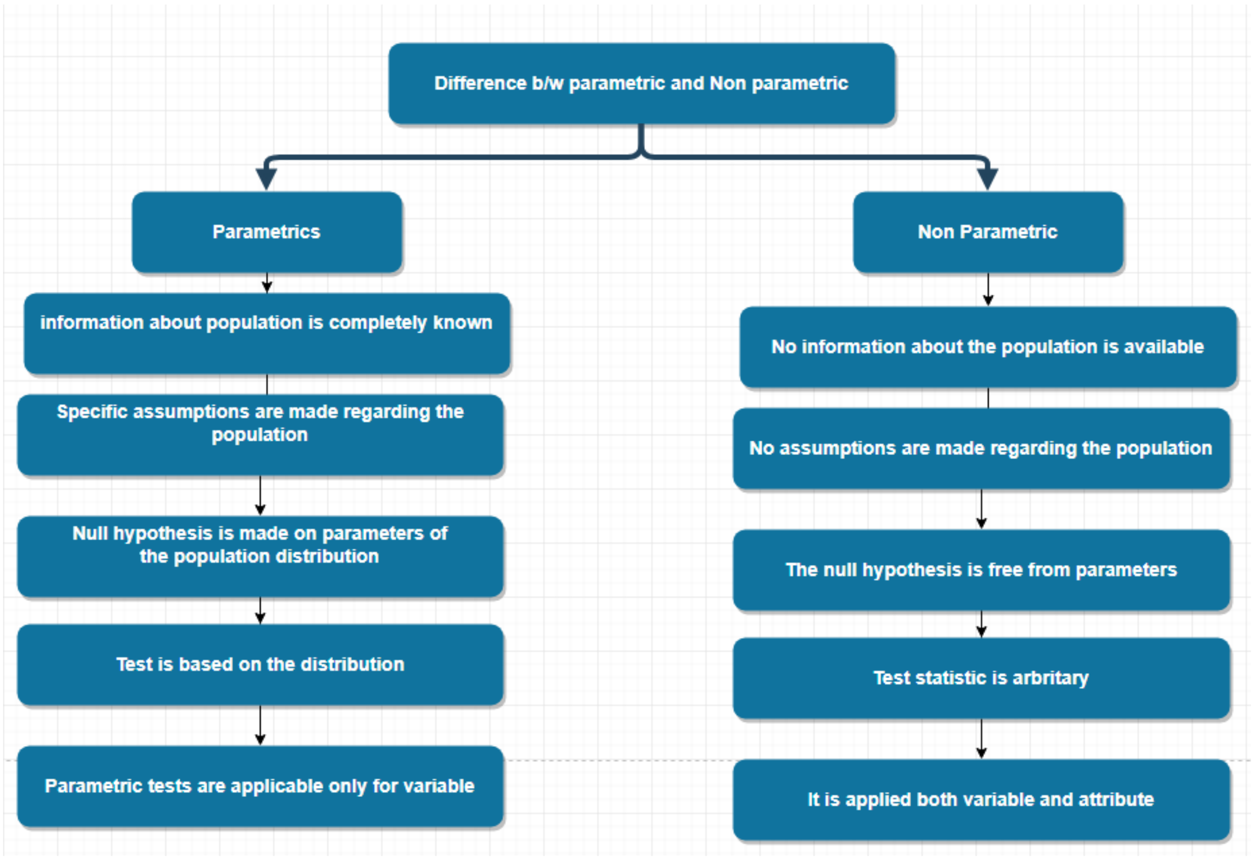
```
## [1] 2
```

Parametric and Non parametric classification


Classification of Hypothesis
Parametric Test
Non Parametric Test
t-test, f-test,z-test,ANOVA
mann-whitney,rank sum test,kruskal-Eallis test

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

Difference between Parametric and Non parametric test


Difference b/w parametric and Non parametric
Parametrics
Non Parametric
information about population is completely known
No information about the population is available
Specific assumptions are made regarding the population
No assumptions are made regarding the population
Null hypothesis is made on parameters of the population distribution
The null hypothesis is free from parameters
Test is based on the distribution
Test statistic is arbritary
Parametric tests are applicable only for variable
It is applied both variable and attribute

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

Advantages of Non - parametric Test

- Non Parametric test are simple and easy to understatnd
- It will not involve complicated sampling theory
- No assumption is made regarding the parent population
- This method is only available for nominal scale data
- This method is easy applicable for attribute dates

Disadvantages of non-parametric test

- It can be applied only for nominal or ordinal scale
- for any problem, if any parametric test exist it is highly powerful
- Non parametric methods are not so efficient as of parametric test
- No nonparametric test available for testing the interaction in analysis of variance model

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

Chapter 1 : One-Sample T-test

Read Data

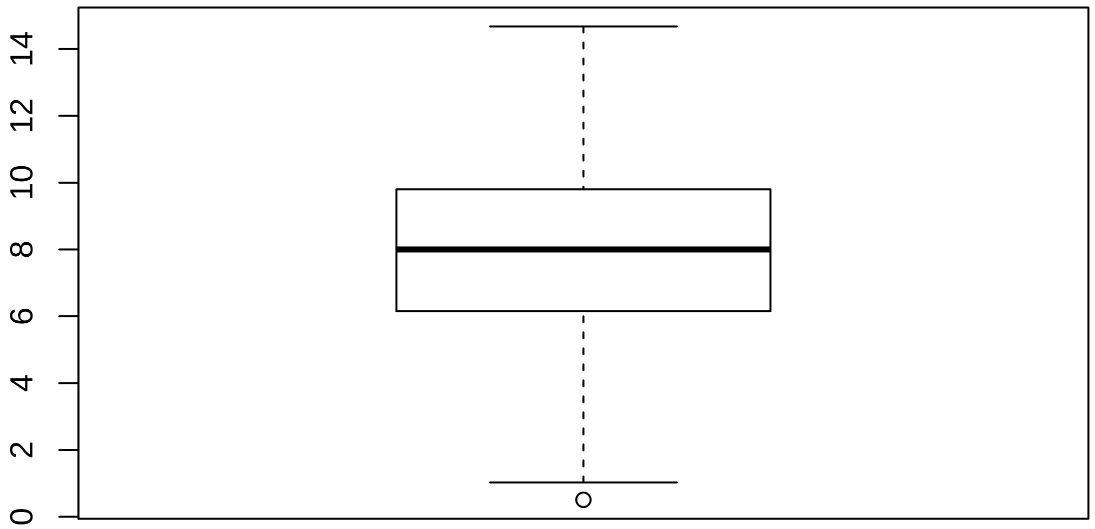
```
LungCapData <- read.csv("LungCapData.txt",header = T,sep = "\t")
attach(LungCapData)
names(LungCapData)
```

```
## [1] "LungCap"   "Age"        "Height"    "Smoke"     "Gender"    "Caesarean"
```

We can conduct the t-test and confidence interval using t.test command

Before beginning any analysis it is useful examine the plot of the data.

```
boxplot(LungCap)
```



## H0: mu < 8 ## one-sided 95% confidence interval for mu

```
t.test(LungCap, mu=8,alternative = "less",conf.level = 0.95)
```

```
##
##  One Sample t-test
##
## data:  LungCap
## t = -1.3842, df = 724, p-value = 0.08336
## alternative hypothesis: true mean is less than 8
## 95 percent confidence interval:
##      -Inf 8.025974
## sample estimates:
## mean of x
##  7.863148
```

```r
t.test(LungCap, mu=8,alt = "less",conf = 0.95)
```

```
##
##  One Sample t-test
##
## data:  LungCap
## t = -1.3842, df = 724, p-value = 0.08336
## alternative hypothesis: true mean is less than 8
## 95 percent confidence interval:
##      -Inf 8.025974
## sample estimates:
## mean of x
##  7.863148
```

#two sided

```r
t.test(LungCap, mu=8,alternative = "two.sided",conf.level = 0.95)
```

```
##
##  One Sample t-test
##
## data:  LungCap
## t = -1.3842, df = 724, p-value = 0.1667
## alternative hypothesis: true mean is not equal to 8
## 95 percent confidence interval:
##  7.669052 8.057243
## sample estimates:
## mean of x
##  7.863148
```

two sided test is default in r

```r
TEST <- t.test(LungCap, mu=8,conf.level = 0.95)
```

attributes of object

```r
attributes(TEST)
```

```
## $names
## [1] "statistic"   "parameter"   "p.value"     "conf.int"    "estimate"
## [6] "null.value"  "alternative" "method"      "data.name"
##
## $class
## [1] "htest"
```

```r
TEST$p.value
```

```
## [1] 0.1667108
## [1] 2
```

Chapter 2 : Two - Sample t Test

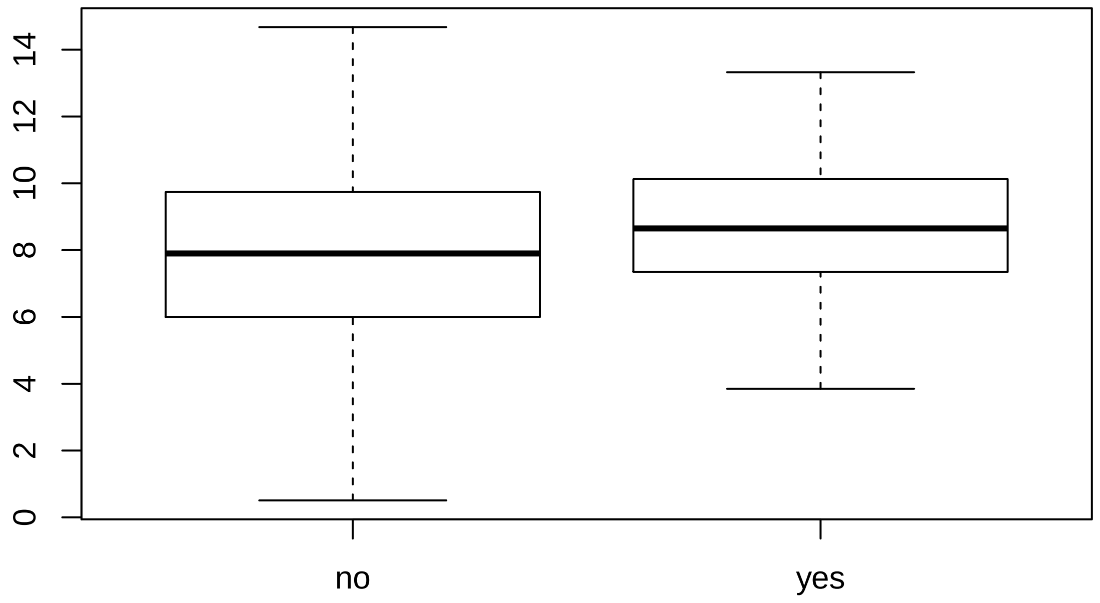
Conducting the Independent 2 - sample t-test and confidence interval.

These are parametric methods appropriate for examining the difference in Means of for 2 population.

These are ways of examining the relationship between a numeric outcome variable (Y) and a categorical explanatory variable (X, with 2 levels)

Working with LungCapData

```
boxplot(LungCap ~ Smoke)
```



### H0 : mean Lung cap of smoker = of non smokers

```
t.test(LungCap ~ Smoke,mu=0,alt="two.sided",conf=0.95,var.eq=F,paired=F)
```

```
##
##  Welch Two Sample t-test
##
## data:  LungCap by Smoke
## t = -3.6498, df = 117.72, p-value = 0.0003927
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -1.3501778 -0.4003548
## sample estimates:
##  mean in group no mean in group yes
##          7.770188          8.645455
```

Subsetting t.test

```
t.test(LungCap[Smoke == "no"],LungCap[Smoke == "yes"])
```

```
##
##  Welch Two Sample t-test
##
## data:  LungCap[Smoke == "no"] and LungCap[Smoke == "yes"]
## t = -3.6498, df = 117.72, p-value = 0.0003927
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -1.3501778 -0.4003548
## sample estimates:
## mean of x mean of y
##  7.770188  8.645455
```

```
t.test(LungCap ~ Smoke,mu=0,alt="two.sided",conf=0.95,var.eq=TRUE,paired=F)
```

```
##
##  Two Sample t-test
##
## data:  LungCap by Smoke
## t = -2.7399, df = 723, p-value = 0.006297
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -1.5024262 -0.2481063
## sample estimates:
##  mean in group no mean in group yes
##          7.770188          8.645455
```

Equal or non Equal variances

```
var(LungCap[Smoke == "yes"])
```

```
## [1] 3.545292
```

```
var(LungCap[Smoke == "no"])
```

```
## [1] 7.431694
```

Levene's Test

H0: population variances are equal.

```
library(car)
```

```
## Loading required package: carData
```

```
leveneTest(LungCap ~ Smoke)
```

```
## Levene's Test for Homogeneity of Variance (center = median)
##        Df F value    Pr(>F)
## group   1  12.955 0.0003408 ***
##       723
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## [1] 2
```

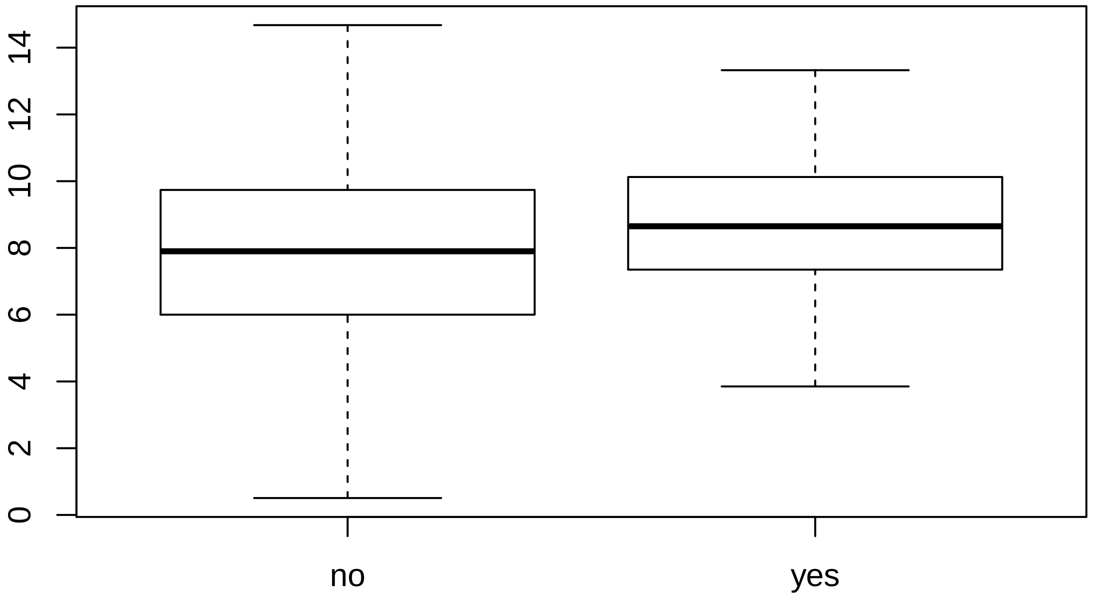
Chapter 3 : Mann Whitney U Test

Conducting the Mann - Whitney U Test A.K.A Wilcoxon Rank - Sum Test

This is a non - parametric method appropriate for examining the difference in Median for 2 independent populations.

It is a way of examining the relationship between a numeric autcome variable(Y) and a categorical explanatory variable (x, with 2 levels) when groups are independent!

```
boxplot(LungCap ~ Smoke)
```



## Ho : Median Lung Capacity of smokers = that of non smokers ## two sided test

```
wilcox.test(LungCap ~ Smoke, mu=0, alt="two.sided",conf.int=T,conf.level=0.95,exact = T,correct=T)
```

```
## Warning in wilcox.test.default(x = c(6.475, 9.55, 11.125, 4.8, 6.225,
## 4.95, : cannot compute exact p-value with ties

## Warning in wilcox.test.default(x = c(6.475, 9.55, 11.125, 4.8, 6.225,
## 4.95, : cannot compute exact confidence intervals with ties

##
##  Wilcoxon rank sum test with continuity correction
##
## data:  LungCap by Smoke
## W = 20128, p-value = 0.005538
## alternative hypothesis: true location shift is not equal to 0
## 95 percent confidence interval:
##  -1.3999731 -0.2499419
## sample estimates:
## difference in location
##             -0.8000564
```

```r
wilcox.test(LungCap ~ Smoke, mu=0, alt="two.sided",conf.int=T,conf.level=0.95,exact = F,correct=T)
```

```
##
##  Wilcoxon rank sum test with continuity correction
##
## data:  LungCap by Smoke
## W = 20128, p-value = 0.005538
## alternative hypothesis: true location shift is not equal to 0
## 95 percent confidence interval:
##  -1.3999731 -0.2499419
## sample estimates:
## difference in location
##             -0.8000564
## [1] 2
```

Chapter 4 : Boostrap Hypothesis Testing

Comparing Numeric variable for 2 Groups

An alternative approach to two-sample t-test comparing independent Groups.

load in the chicken diet data (save it in "d)

```
d <- read.csv("ChickData.csv",header=T,sep=",")
```

This data is a subset of the "chickwts" data in the "R datasets package"

```
names(d)
```

```
## [1] "weight" "feed"
```

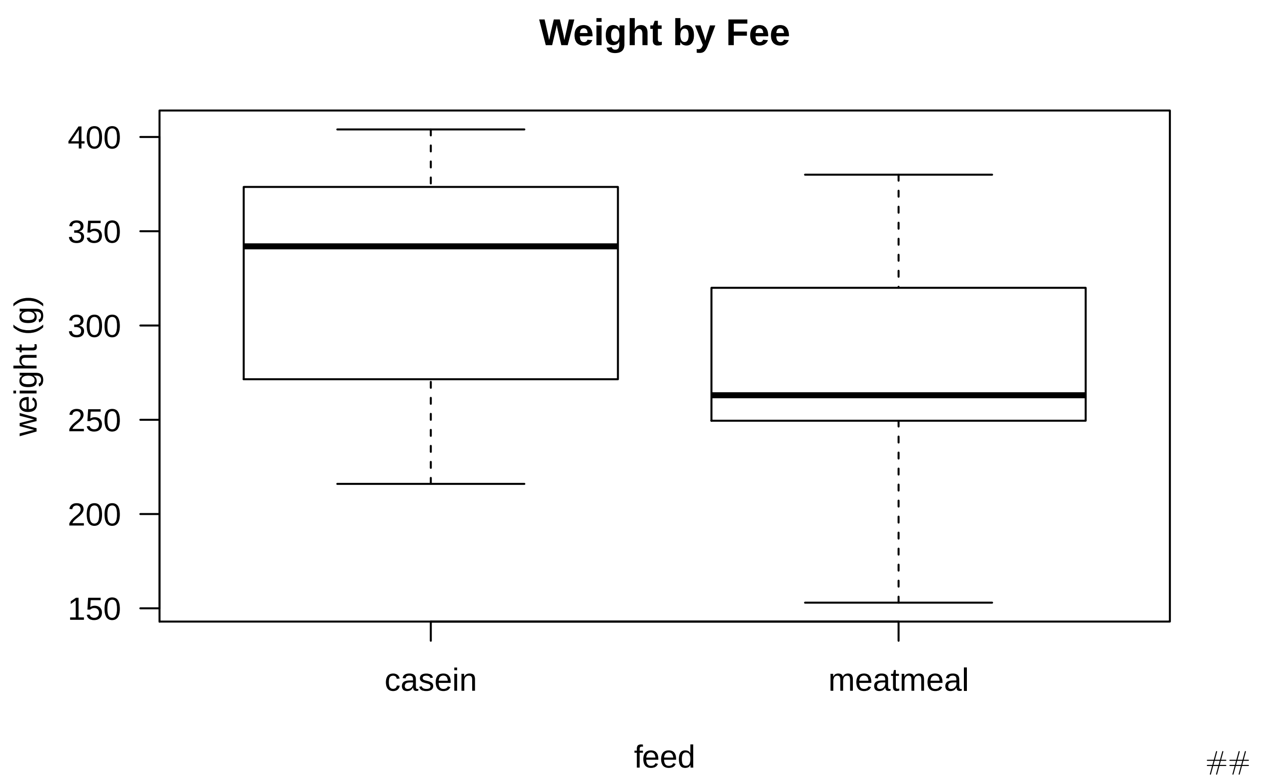
how many observations in each diet?

```
table(d$feed)
```

```
##
##   casein meatmeal
##       12       11
```

Let's look at a boxplot of weight again by those 2 diets

```
boxplot(d$weight ~ d$feed,las=1,ylab = "weight (g)",xlab="feed",main="Weight by Fee")
```



##

Calculate the difference in sample means

```
mean(d$weight[d$feed == "casein"]) # mean for casein
```

```
## [1] 323.5833
```

```
mean(d$weight[d$feed == "meatmeal"]) # mean for meatmeal
```

```
## [1] 276.9091
```

and, a fancier way to do that ..

```
with(d,tapply(weight, feed, mean))
```

```
##   casein meatmeal
## 323.5833 276.9091
```

lets calculate the absolute diff in means

```
test.stat1 <- abs(mean(d$weight[d$feed == "casein"])- mean(d$weight[d$feed =="meatmeal"])) #diff in mea
test.stat1
```

```
## [1] 46.67424
```

and, a fancier way to do that . . .

```
  abs(diff(with(d,tapply(weight, feed, mean))))
```

```
## meatmeal
## 46.67424
```

and, the same for the medians

```
median(d$weight[d$feed == "casein"]) # median for casein
```

```
## [1] 342
```

```
median(d$weight[d$feed == "meatmeal"]) # median for meatmeal
```

```
## [1] 263
```

and , a fancier way to do that . . .

```
with(d,tapply(weight, feed, median))
```

```
##   casein meatmeal
##      342      263
```

lets calculate the absolute diff in medians

```
test.stat2 <- abs(median(d$weight[d$feed == "casein"] - median(d$weight[d$feed == "meatmeal"])))
test.stat2
```

```
## [1] 79
```

and, a fancier way to do that . . . .

```
abs(diff(with(d,tapply(weight, feed, median))))
```

```
## meatmeal
##       79
```

let's take a look at the 3 "Classic hyp tests we could consider (ech of which comes with their own limitations ..)

let's look at the Independent 2-sample t-test

```
t.test(d$weight ~ d$feed,paired=F) # tests H0: medians are equal
```

```
##
##  Welch Two Sample t-test
##
## data:  d$weight by d$feed
## t = 1.7288, df = 20.799, p-value = 0.09866
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##   -9.504377 102.852861
## sample estimates:
##   mean in group casein mean in group meatmeal 
##               323.5833               276.9091 
```

let's look at the Wilcoxon aka Mann-Whitney U

```r
wilcox.test(d$weight ~ d$feed,paired=F)
```

```
## 
##  Wilcoxon rank sum test
## 
## data:  d$weight by d$feed
## W = 94, p-value = 0.09084
## alternative hypothesis: true location shift is not equal to 0
```

let's look at the Kolmogorov-Smirnov 2-Sample test

```r
ks.test(d$weight[d$feed == "casein"],d$weight[d$feed == "meatmeal"],paired=F)
```

```
## 
##  Two-sample Kolmogorov-Smirnov test
## 
## data:  d$weight[d$feed == "casein"] and d$weight[d$feed == "meatmeal"]
## D = 0.40909, p-value = 0.1957
## alternative hypothesis: two-sided
```

setting the seed Produces the exact same random data every time you run the code

Not setting the seed ,every time you run the code you will get slightly different data and results.

```r
set.seed(112358) # for reproducibility
n <- length(d$feed) # the number of observations to sample
n
```

```
## [1] 23
```

```r
B <- 10000 # the number of bootstrap samples
B
```

```
## [1] 10000
```

```r
variable <- d$weight # the variable we will resample from
```

now, get those bootstrap samples (without loops!)

```r
BootstrapSamples <- matrix(sample(variable,size = n*B,replace = TRUE),nrow = n,ncol = B)
```

let's take a moment to discuss what that code is doing ..

```
dim(BootstrapSamples)
```

```
## [1]    23 10000
```

now, calculate the means (Yc and Ym) for each of the bootstrap samples (the inefficeint, but transparent way . . . best to start simple, and once working well, then make code more efficent)

initalize the vector to store the TEST-STATS

```
Boot.test.stat1 <- rep(0,B)
```

```
Boot.test.stat2 <- rep(0,B)
```

run through a loop, each time calculating the bootstrap test.stat

Note: could make this faster by writing a "Function" and then using "apply" to apply it to columns of the "BootSamples"

```
for (i in 1:B) {
  #Calculate the boot-test-stat1 and save it
  Boot.test.stat1[i] <- abs(mean(BootstrapSamples[1:12,i])- mean(BootstrapSamples[13:23,i]))
  #Calculate the boot-test-stat2 and save it
  Boot.test.stat2[i] <- abs(median(BootstrapSamples[1:12,i])-median(BootstrapSamples[13:23,i]))
 }
```

before going too far, let's remind ourselves of the OBSERVED TEST STATS

```
test.stat1;test.stat2
```

```
## [1] 46.67424
```

```
## [1] 79
```

and, take a look at the first 20 Bootstrap- TEST STATS for 1 and 2

```
round(Boot.test.stat1[1:20],1)
```

```
##  [1] 14.7  5.5 13.2 21.1 10.0 48.0 15.7 27.5 13.7 30.1  8.3 23.8 25.0 19.6
## [15] 68.2 20.1  1.4 45.6 22.4 34.3
```

```
round(Boot.test.stat2[1:20],1)
```

```
##  [1] 15.0  8.5  6.5 63.5 55.0 78.5 53.5 39.0 46.0 21.0  6.0 41.0 52.5 33.5
## [15] 73.5 60.5 32.0 79.0 18.5 52.0
```

remind ourselves what is p-values?

P- value tells us under set of assumptions, what is the probability of getting the observed test statistic or one more extreme, if the null hypothesis is True

What is the probability of getting a (test statistic >= 46.67) if the null hypothesis were true and we'd expect:(test statistic = zero)

P-value formula

p-value = the number of bootstrap test statistics that are greater than the observed test statistic/B(the total number of bootstrap test statistics)

and, let's calculate the boostrap p-value ..

notice how we can ask R a true/false question .. (for the first 20)

```
(Boot.test.stat1 >= test.stat1)[1:20]
```

```
##  [1] FALSE FALSE FALSE FALSE FALSE  TRUE FALSE FALSE FALSE FALSE FALSE
## [12] FALSE FALSE FALSE  TRUE FALSE FALSE FALSE FALSE FALSE
```

and if we ask for the mean of all of those, it treats 0=FALSE ,1=TRUE and calculate the p-value.

```
mean(Boot.test.stat1 >= test.stat1)
```

```
## [1] 0.0832
```

P-value = 8.32% out of the 10,000 bootstrap test statistics calculated, 832 of them had test statistics greater than the observed one.

P-value =8% if there is no difference in the mean weights, we would see a test stastic of 46.67 or more by chance roughly 8% of the time.

let's calculate the p-value for test statistic 2 (abs diff in medians)

```
mean(Boot.test.stat2 >= test.stat2)
```

```
## [1] 0.063
```

now, recall the difference between "Statisitcal significace" and "scientific significance"

in a "real_world" what would you wnat to conclude here

```
table(d$feed)
```

```
##
##   casein meatmeal
##       12       11
## [1] 2
```

Chapter 5 : BOOTSTRAP CONFIDENCE INTERVAL

(comparing 2 numeric variables)

load in the chicken diet data (save it in "d")

```
d <- read.table(file="ChickData.csv", header=T, sep=",")
```

this data is a subset of the "chickwts" data in the

"R datasets package"

let's add the data into the "data view"

check the names, etc

```
names(d)
```

```
## [1] "weight" "feed"
```

```
levels(d$feed)
```

```
## [1] "casein"   "meatmeal"
```

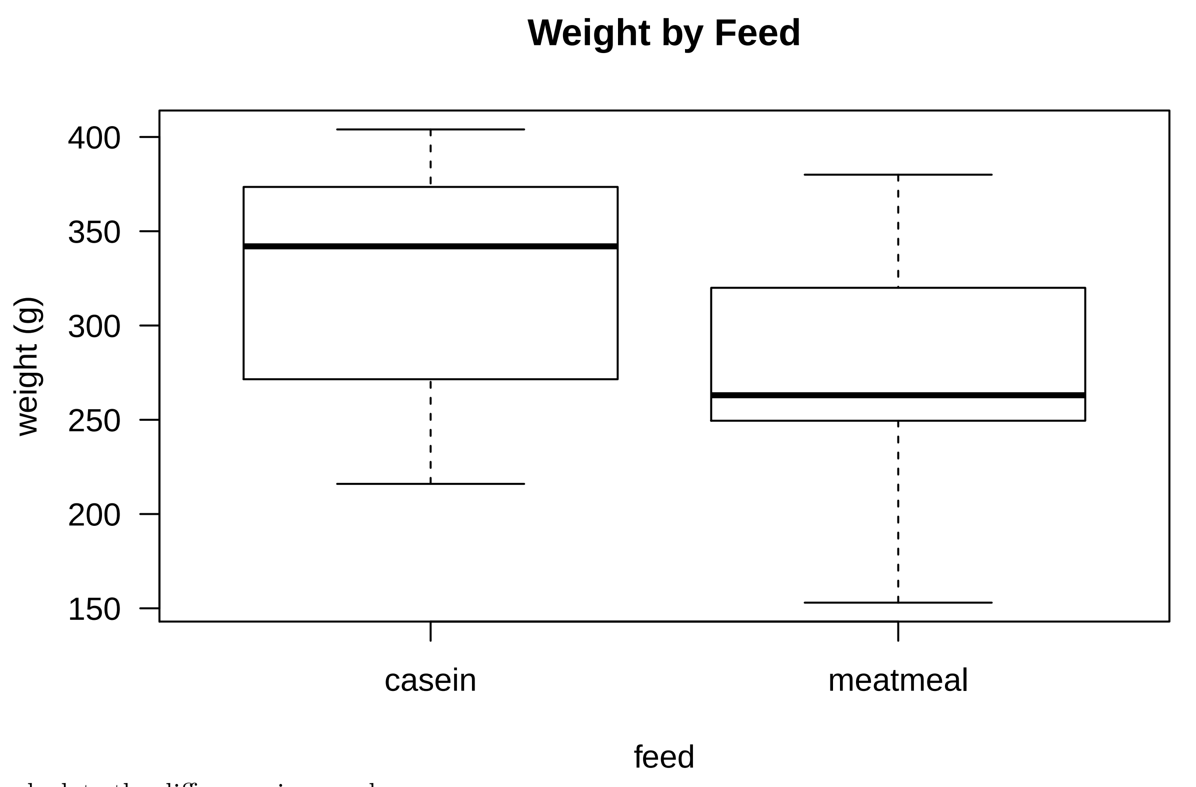
how many observations in each diet?

```
table(d$feed)
```

```
##
##   casein meatmeal
##       12       11
```

let's look at a boxplot of weight gain by those 2 diets

```
boxplot(d$weight~d$feed, las=1, ylab="weight (g)",
        xlab="feed",main="Weight by Feed")
```



calculate the difference in sample means

```
mean(d$weight[d$feed=="casein"])  # mean for casein
```

```
## [1] 323.5833
```

```
mean(d$weight[d$feed=="meatmeal"])  # mean for meatmeal
```

```
## [1] 276.9091
```

and, a fancier way to do that...

```
with(d, tapply(weight, feed, mean))
```

```
##   casein meatmeal
## 323.5833 276.9091
```

lets calculate the diff in means: (casein - meatmeal)

```
Obs.Diff.In.Means <- (mean(d$weight[d$feed=="casein"]) - mean(d$weight[d$feed=="meatmeal"]))  #diff in
Obs.Diff.In.Means
```

```
## [1] 46.67424
```

and, a fanceir way to do that... (- to have it be casein-meatmeal)

```
-diff( with(d, tapply(weight, feed, mean)) )
```

```
## meatmeal
## 46.67424
```

and, the same for the medians

```
median(d$weight[d$feed=="casein"])  # median for casein
```

```
## [1] 342
```

```
median(d$weight[d$feed=="meatmeal"])  # median for meatmeal
```

```
## [1] 263
```

and, a fancier way to do that...

```
with(d, tapply(weight, feed, median))
```

```
##   casein meatmeal
##      342      263
```

lets calculate the diff in medians: (casein - meatmeal)

```
Obs.Diff.In.Medians <- (median(d$weight[d$feed=="casein"]) - median(d$weight[d$feed=="meatmeal"]))  #di
Obs.Diff.In.Medians
```

```
## [1] 79
```

and, a fanceir way to do that... (- to have it be casein-meatmeal)

```
-diff( with(d, tapply(weight, feed, median)) )
```

```
## meatmeal
##       79
```

BOOTSTRAP CONFIDENCE INTERVAL

let's run through making conf ints for the difference in means and medians

let's bootstrap...

```
set.seed(13579)   # set a seed for consistency/reproducability
n.c <- 12  # the number of observations to sample from casein
n.m <- 11  # the number of observations to sample from meatmeal
B <- 100000  # the number of bootstrap samples...go big or go home right?
```

now, get those bootstrap samples (without loops!)

stick each Boot-sample in a column...

```
Boot.casein <- matrix( sample(d$weight[d$feed=="casein"], size= B*n.c,
                              replace=TRUE), ncol=B, nrow=n.c)
Boot.meatmeal <- matrix( sample(d$weight[d$feed=="meatmeal"], size= B*n.m,
                                replace=TRUE), nrow=n.m, ncol=B)
```

check those

```
dim(Boot.casein); dim(Boot.meatmeal)
```

```
## [1]     12 100000
```

```
## [1]     11 100000
```

check to make sure they are not empty!

```
Boot.casein[1:5,1:5]
```

```
##      [,1] [,2] [,3] [,4] [,5]
## [1,]  390  222  379  379  222
## [2,]  283  359  260  404  260
## [3,]  390  352  260  216  318
## [4,]  283  222  222  379  283
## [5,]  404  390  260  318  379
```

```
Boot.meatmeal[1:5,1:5]
```

```
##      [,1] [,2] [,3] [,4] [,5]
## [1,]  242  344  344  153  303
## [2,]  344  303  315  315  206
## [3,]  153  242  263  380  242
## [4,]  303  242  258  344  344
## [5,]  344  325  315  344  263
```

calculate the difference in MEANS for each of the bootsamples

```
Boot.Diff.In.Means <- colMeans(Boot.casein) - colMeans(Boot.meatmeal)
```

check that

```
length(Boot.Diff.In.Means)
```

```
## [1] 100000
```

and, look at the first 10 diff in means

```
Boot.Diff.In.Means[1:10]
```

```
##  [1] 75.43939 23.03030 33.71970 54.30303 34.84848 40.58333 45.36364
##  [8] 79.65909 82.45455 19.20455
```

calculate the difference in MEDIANS for each of the bootsamples

```
Boot.Diff.In.Medians <- apply(Boot.casein, MARGIN=2, FUN=median) -
  apply(Boot.meatmeal, MARGIN=2, FUN=median)
```

check that

```
length(Boot.Diff.In.Medians)
```

```
## [1] 100000
```

and, look at the first 10 diff in medians

```
Boot.Diff.In.Medians[1:10]
```

```
##  [1]  90.0  29.0  79.0  53.0  55.0  52.5  82.5 103.0  94.0  69.0
```

MAKE THE CONFIDENCE INTERVALS (using 95% confidence)

let's look at the PERCENTILE METHOD

the "PERCENTILE" bootstrap confidence interval

first, for the difference in MEANS

```
quantile(Boot.Diff.In.Means, prob=0.025)
```

```
##      2.5%
## -4.007955
```

```
quantile(Boot.Diff.In.Means, prob=0.975)
```

```
##    97.5%
## 96.82595
```

and then, the difference in MEDIANS

```
quantile(Boot.Diff.In.Medians, prob=0.025)
```

```
##  2.5%
## -24.5
```

```r
quantile(Boot.Diff.In.Medians, prob=0.975)
```

```
## 97.5%
##   116
```

What do you make of the fact that these both cross 0?

Apart from "statistical significance", what do you think about

"scientific significance" here?

below is code to calculate confidence interval using the BASIC method

let's look at the BASIC METHOD

first, for the difference in MEANS

```r
2*Obs.Diff.In.Means - quantile(Boot.Diff.In.Means, prob=0.975)
```

```
##     97.5%
## -3.477462
```

```r
2*Obs.Diff.In.Means - quantile(Boot.Diff.In.Means, prob=0.025)
```

```
##     2.5%
## 97.35644
```

and then, the difference in MEDIANS

```r
2*Obs.Diff.In.Medians - quantile(Boot.Diff.In.Medians, prob=0.975)
```

```
## 97.5%
##    42
```

```r
2*Obs.Diff.In.Medians - quantile(Boot.Diff.In.Medians, prob=0.025)
```

```
##  2.5%
## 182.5
```

Code for confidence interval for difference in 80th percentiles

calculate the observed difference in 80th percentiles

```r
Obs.Diff.In.80per <- (quantile(d$weight[d$feed=="casein"], prob=0.80) - quantile(d$weight[d$feed=="meatm
Obs.Diff.In.80per
```

```
##  80%
## 51.8
```

calculate the difference in 80th percentile for each of the bootsamples

```
Boot.Diff.In.80per <- apply(Boot.casein, MARGIN=2, FUN=quantile, prob=0.80) - apply(Boot.meatmeal, MARG
```

let's look at the PERCENTILE METHOD for the difference in 80th percentile

```
quantile(Boot.Diff.In.80per, prob=0.025)
```

```
## 2.5%
##  -12
```

```
quantile(Boot.Diff.In.80per, prob=0.975)
```

```
## 97.5%
##   116
## [1] 2
```

Chapter 6 : Paired t-Test

Conducting the paired t-test and confidence Interval

These are parametric methods appropriate for examining the difference in Means for 2 populations that are paired or dependent o one another...

```
BloodPressure <- read.csv("BloodPressure.txt",header = T,sep = "\t")
```

```
names(BloodPressure)
```

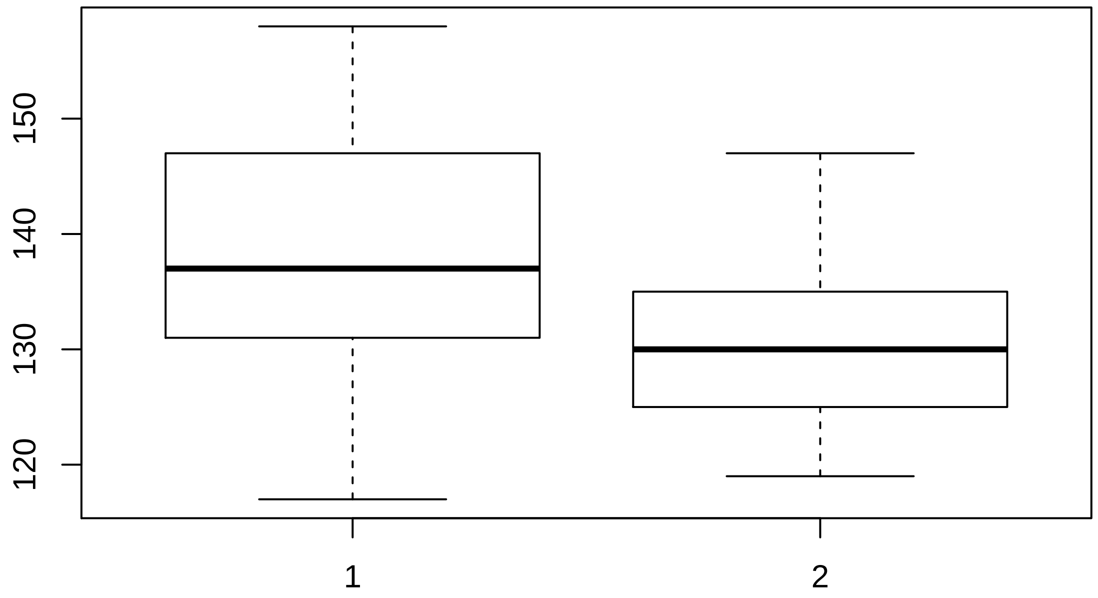
```
## [1] "Subject" "Before"  "After"
```

```
attach(BloodPressure)
```

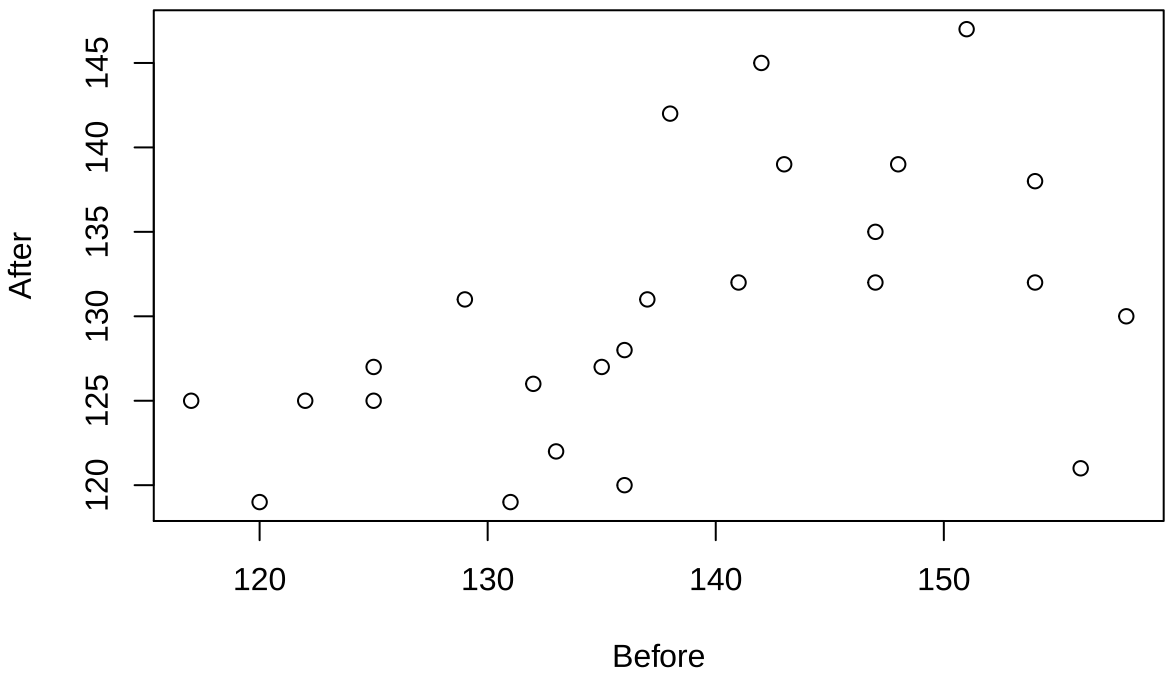
```
BloodPressure[1:3,]
```

```
##   Subject Before After
## 1       1    135   127
## 2       2    142   145
## 3       3    137   131
```

```
boxplot(Before,After)
```



```
plot(Before,After)
```



abline(a=0,b=1)

Ho: mean difference in SBP is 0 two-sided test

```
t.test(Before,After,mu=0,alt="two.side",paired = T,conf.level = 0.99)
```

```
##
##  Paired t-test
##
## data:  Before and After
## t = 3.8882, df = 24, p-value = 0.0006986
## alternative hypothesis: true difference in means is not equal to 0
## 99 percent confidence interval:
##   2.245279 13.754721
## sample estimates:
## mean of the differences
##                       8
```

```
t.test(After,Before,mu=0,alt="two.side",paired = T,conf.level = 0.99)
```

```
##
##  Paired t-test
##
## data:  After and Before
## t = -3.8882, df = 24, p-value = 0.0006986
## alternative hypothesis: true difference in means is not equal to 0
## 99 percent confidence interval:
##  -13.754721  -2.245279
## sample estimates:
## mean of the differences
##                      -8
## [1] 2
```

Chapter 7 : Wilcoxon Singed Rank Test

Conducting the Wilcoxon Signed Rank Test

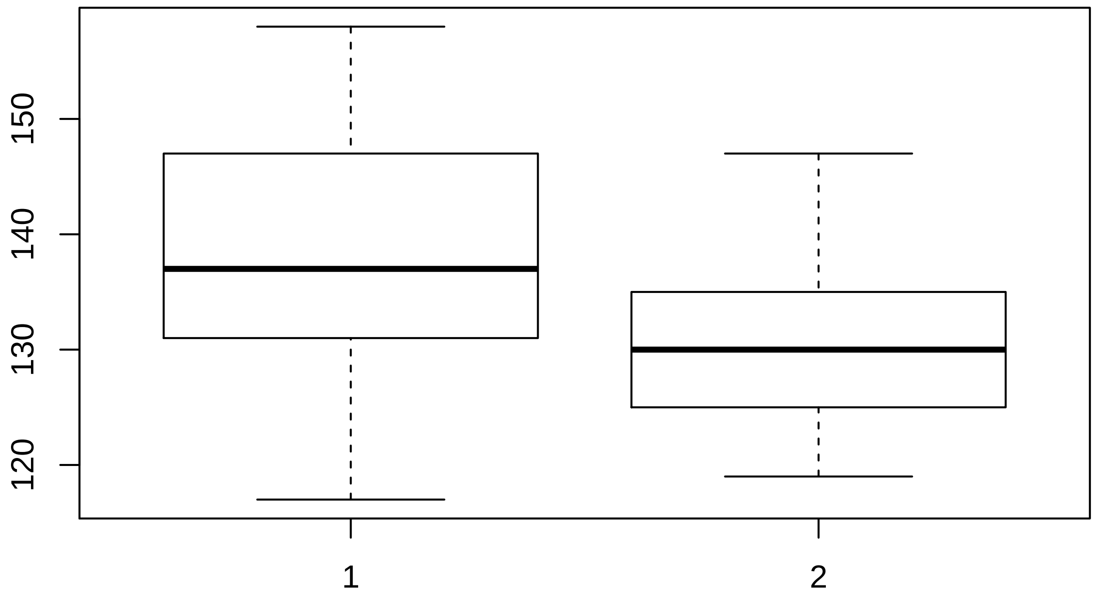
This is a non-parametric method appropriate for examining the Median difference in observations for 2 populations that paired or dependent on one another ….

```
BloodPressure <- read.csv("BloodPressure.txt",header = T,sep = "\t")
```

```
BloodPressure[c(1,3,5),]
```

```
##   Subject Before After
## 1       1    135   127
## 3       3    137   131
## 5       5    147   132
```

```
boxplot(Before,After)
```



## H0 : Median Change in SBP is 0 two.sided test

```
wilcox.test(Before,After,mu=0,alt="two.sided",paired=T,conf.int=T,conf.level=0.99)
```

```
## Warning in wilcox.test.default(Before, After, mu = 0, alt = "two.sided", :
## cannot compute exact p-value with ties

## Warning in wilcox.test.default(Before, After, mu = 0, alt = "two.sided", :
## cannot compute exact confidence interval with ties

## Warning in wilcox.test.default(Before, After, mu = 0, alt = "two.sided", :
## cannot compute exact p-value with zeroes

## Warning in wilcox.test.default(Before, After, mu = 0, alt = "two.sided", :
## cannot compute exact confidence interval with zeroes
```

```
##
##  Wilcoxon signed rank test with continuity correction
##
## data:  Before and After
## V = 267, p-value = 0.0008655
## alternative hypothesis: true location shift is not equal to 0
## 99 percent confidence interval:
##   2.000012 14.000038
## sample estimates:
## (pseudo)median
##       7.500019
```

Approximate p-value and approximate confidence interval we can use exact = F or correct = F

```r
wilcox.test(Before,After,mu=0,alt="two.sided",paired=T,conf.int=T,conf.level=0.99,exact = F,correct = F
```

```
##
##  Wilcoxon signed rank test
##
## data:  Before and After
## V = 267, p-value = 0.0008221
## alternative hypothesis: true location shift is not equal to 0
## 99 percent confidence interval:
##   2.000083 14.000027
## sample estimates:
## (pseudo)median
##       7.500019
## [1] 2
```

Chapter 7 : ANOVA, Multiple Comparisons & Kruskal Wallis

Conducting One-Way Analysis of Variance (ANOVA) and Kruskal Wallis One-Way Analysis of Variance

ANOVA is a parametric method appropriate for comparing the Means 2 or more Independent Populations.

Read Data

```
DietData <- read.csv("DietWeigthLoss.txt",header = T,sep = "\t")
attach(DietData)
```

```
## The following object is masked from package:carData:
##
##     WeightLoss
```

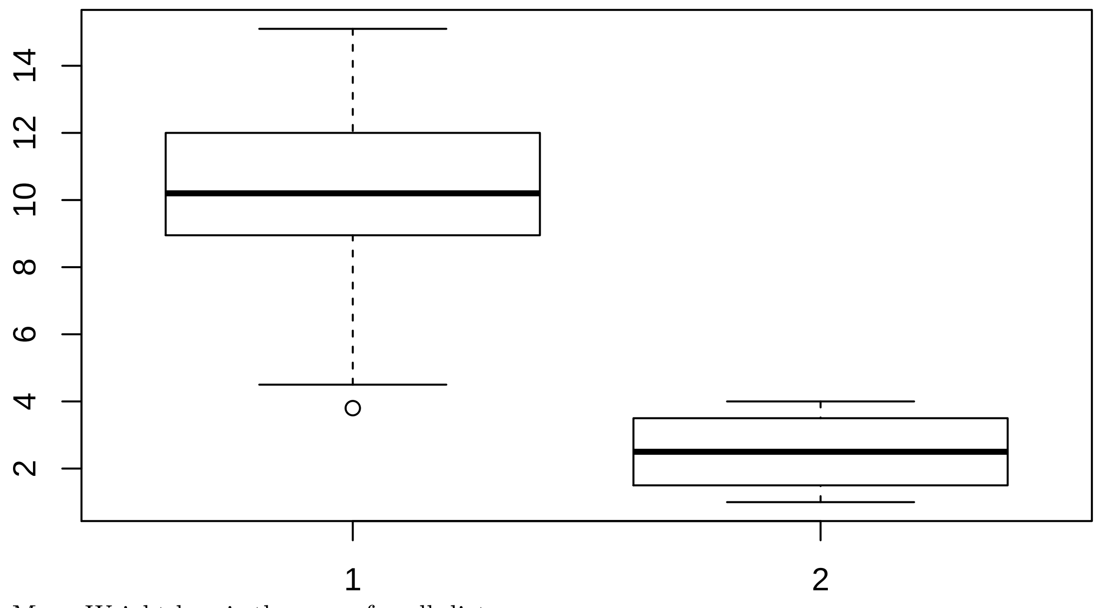
```
names(DietData)
```

```
## [1] "WeightLoss" "Diet"
```

```
class(WeightLoss)
```

```
## [1] "numeric"
```

```
boxplot(WeightLoss,Diet)
```



H0 :
Mean Weight loss is the same for all diets

```
ANOVA1 <- aov(WeightLoss ~ Diet)
```

```
summary(ANOVA1)
```

```
##             Df Sum Sq Mean Sq F value  Pr(>F)
## Diet         3  97.33   32.44   6.118 0.00113 **
## Residuals   56 296.99    5.30
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
attributes(ANOVA1)
```

```
## $names
##  [1] "coefficients"  "residuals"     "effects"       "rank"
##  [5] "fitted.values" "assign"        "qr"            "df.residual"
##  [9] "contrasts"     "xlevels"       "call"          "terms"
## [13] "model"
##
## $class
## [1] "aov" "lm"
```

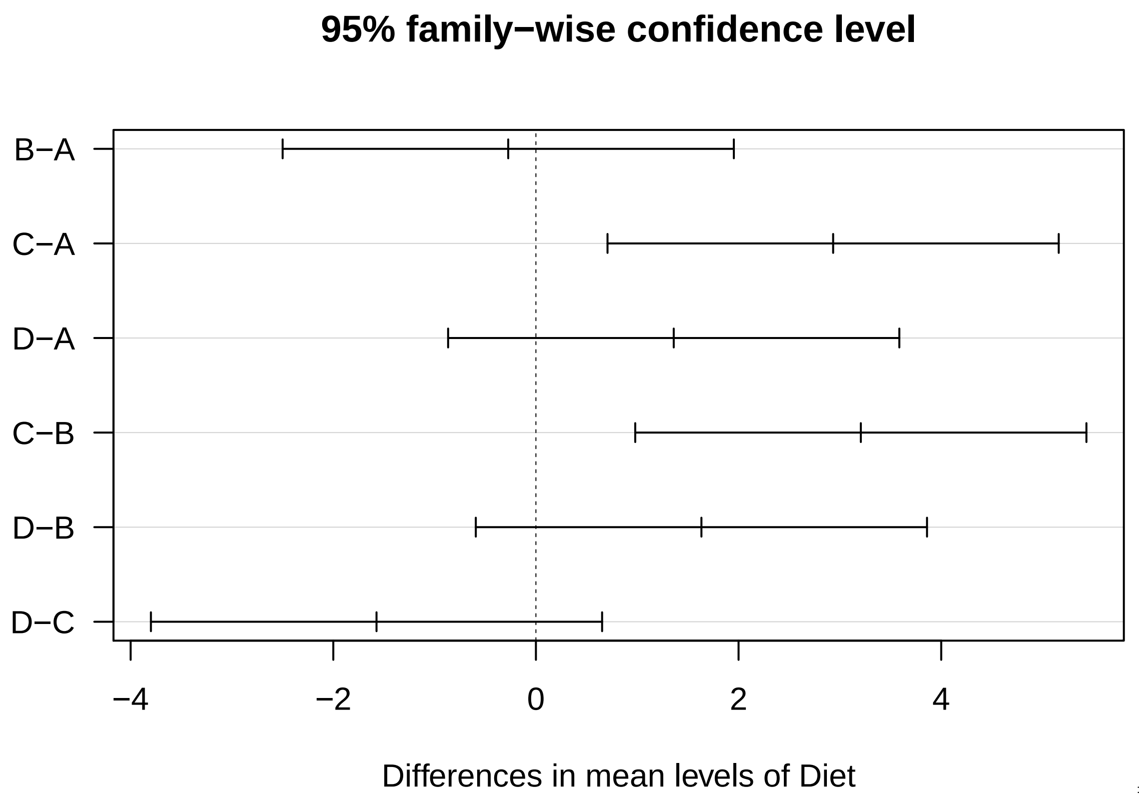
```
ANOVA1$coefficients
```

```
## (Intercept)       DietB       DietC       DietD
##   9.1800000  -0.2733333   2.9333333   1.3600000
```

```
TukeyHSD(ANOVA1)
```

```
##   Tukey multiple comparisons of means
##     95% family-wise confidence level
##
## Fit: aov(formula = WeightLoss ~ Diet)
##
## $Diet
##           diff        lwr       upr     p adj
## B-A -0.2733333 -2.4999391 1.9532725 0.9880087
## C-A  2.9333333  0.7067275 5.1599391 0.0051336
## D-A  1.3600000 -0.8666058 3.5866058 0.3773706
## C-B  3.2066667  0.9800609 5.4332725 0.0019015
## D-B  1.6333333 -0.5932725 3.8599391 0.2224287
## D-C -1.5733333 -3.7999391 0.6532725 0.2521236
```

```
plot(TukeyHSD(ANOVA1),las = 1)
```



# Kruskal Wallis One-Way Analysis of Variance is a non-parametric equivalent to the one-way Analysis of Variance.

```r
kruskal.test(WeightLoss~Diet)
```

```
##
##  Kruskal-Wallis rank sum test
##
## data:  WeightLoss by Diet
## Kruskal-Wallis chi-squared = 15.902, df = 3, p-value = 0.001188
## [1] 2
```

Chapter 8 : Chi-Square Test, Fisher's Exact Test, Cross Tabulations

Conducting the Chi-square test of Independence and Fisher's Test

The Chi-square test of independence is a method appropriate for testing independence between two categorical varialbles.

```
names(LungCapData)
```

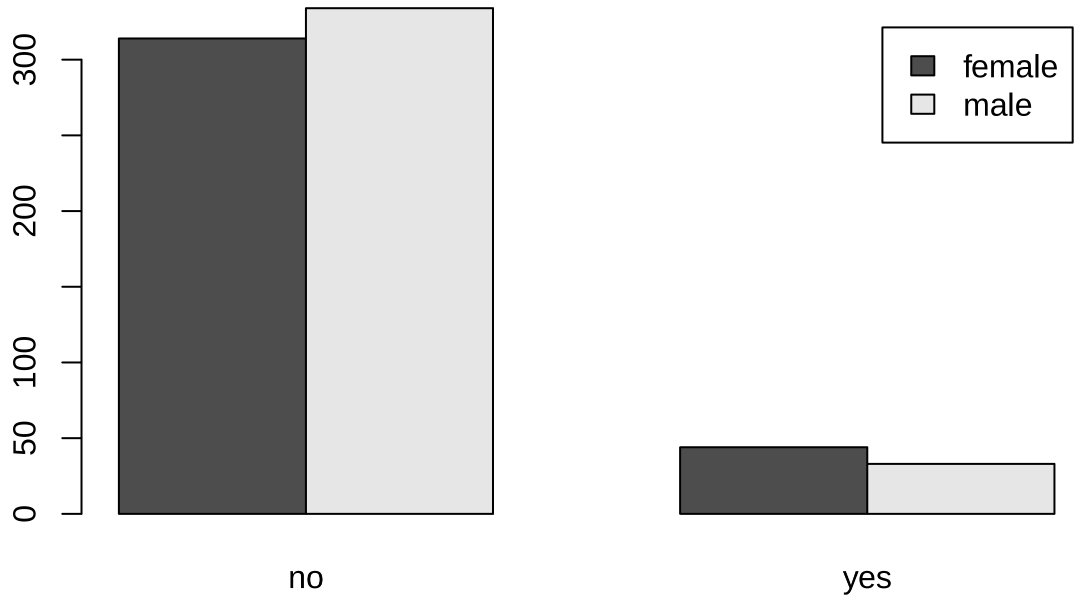
```
## [1] "LungCap"   "Age"        "Height"    "Smoke"     "Gender"    "Caesarean"
```

very first produce a contingency table

```
tab <- table(Gender,Smoke)
```

produce a barplot to examine replationship visually.

```
barplot(tab,beside = T,legend =T)
```



# Produce a chi-square test

```
CHI <- chisq.test(tab,correct = T)
```

```
attributes(CHI)
```

```
## $names
## [1] "statistic" "parameter" "p.value"   "method"    "data.name" "observed"
## [7] "expected"  "residuals" "stdres"
```

```
##
## $class
## [1] "htest"
```

```r
CHI$expected
```

```
##         Smoke
## Gender         no      yes
##   female 319.9779 38.02207
##   male   328.0221 38.97793
```

if the assumptions of chi-square test not met, we may consider using Fisher's Exact Test.

Fisher's Exact test is a non-parametric alternative to the chi-square test.

```r
fisher.test(tab,conf.int = T,conf.level = 0.99)
```

```
##
##  Fisher's Exact Test for Count Data
##
## data:  tab
## p-value = 0.1845
## alternative hypothesis: true odds ratio is not equal to 1
## 99 percent confidence interval:
##  0.3625381 1.3521266
## sample estimates:
## odds ratio
##  0.7054345
## [1] 2
```

Chapter 9 : Odds Ratio, Relative Risk and Risk Difference

Calculating Relative Risk, odds Ratio, and attributable Risk or Risk Difference

These are all measures of the direction and the strength of the association between two categorical variables.

```
names(LungCapData)
```

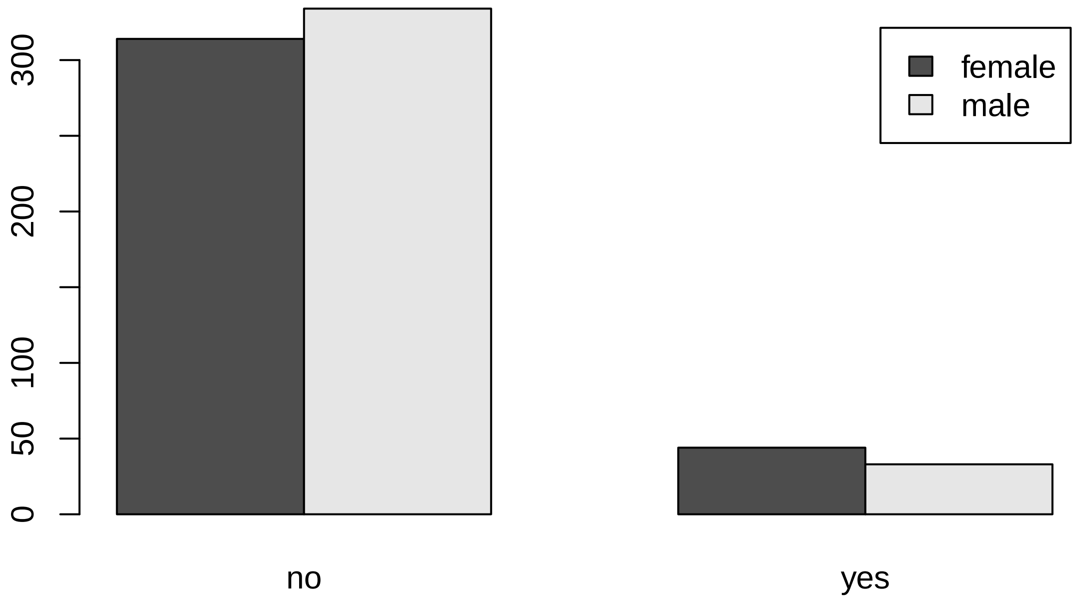
```
## [1] "LungCap"   "Age"        "Height"     "Smoke"      "Gender"     "Caesarean"
```

Exploare the relationship between Gender and Smoking

```
TAB <- table(Gender,Smoke)
TAB
```

```
##         Smoke
## Gender    no yes
##   female 314  44
##   male   334  33
```

```
barplot(TAB,beside = T,legend=T)
```



```
library(epiR)
```

```
## Loading required package: survival

## Package epiR 0.9-99 is loaded

## Type help(epi.about) for summary information

##
```

```
epi.2by2(TAB,method = "cohort.count",conf.level = 0.95)
```

```
##              Outcome +    Outcome -      Total        Inc risk *
## Exposed +          314           44        358              87.7
## Exposed -          334           33        367              91.0
## Total              648           77        725              89.4
##                  Odds
## Exposed +        7.14
## Exposed -       10.12
## Total            8.42
##
## Point estimates and 95 % CIs:
## -------------------------------------------------------------------
## Inc risk ratio                               0.96 (0.92, 1.01)
## Odds ratio                                   0.71 (0.44, 1.14)
## Attrib risk *                                -3.30 (-7.79, 1.19)
## Attrib risk in population *                  -1.63 (-5.32, 2.06)
## Attrib fraction in exposed (%)               -3.76 (-9.12, 1.34)
## Attrib fraction in population (%)            -1.82 (-4.34, 0.64)
## -------------------------------------------------------------------
##  X2 test statistic: 2.077 p-value: 0.15
##  Wald confidence limits
##  * Outcomes per 100 population units
```

The Odds of a female not smoking are 0.71 times the Odds of a male not smoking!

revers 1/0.71 The Odds of a male not smoking are 1.4 times the Odss of a female not smoking.

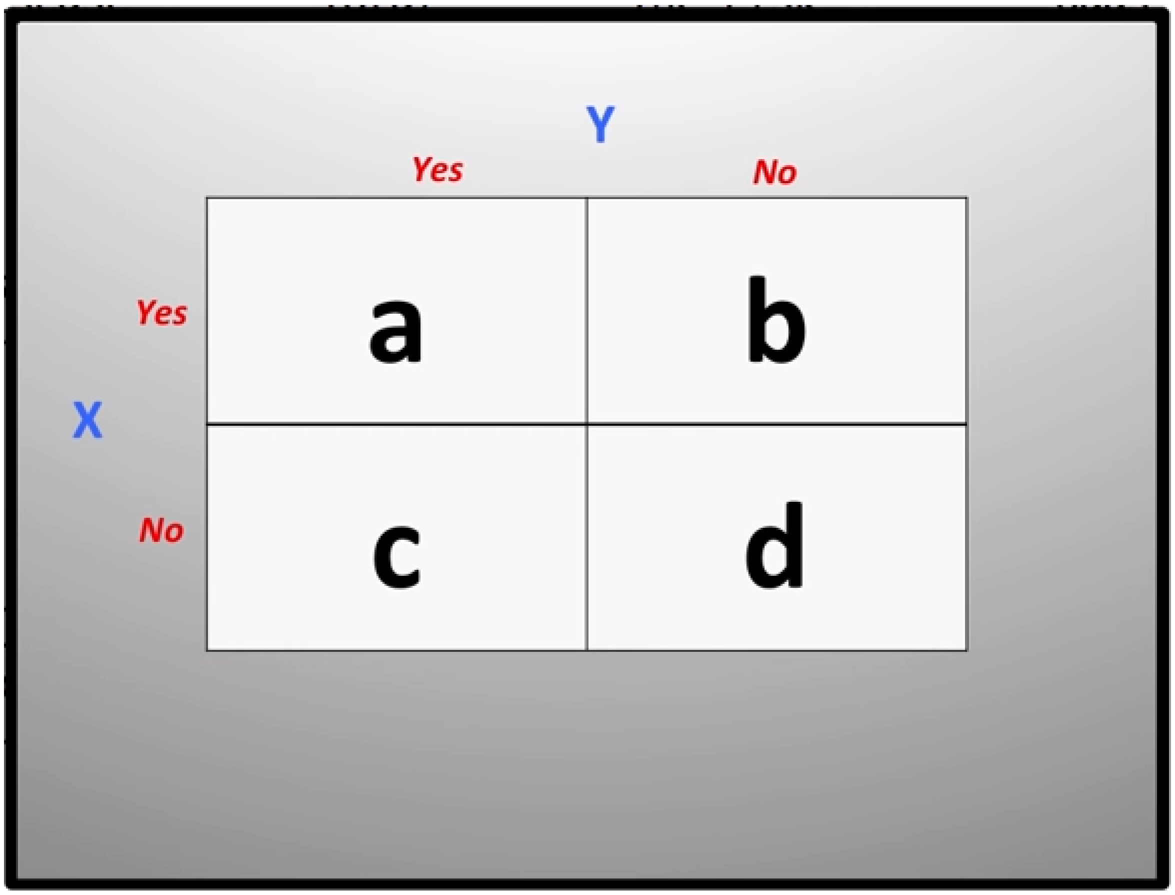


Figure 1: Odds Ratio

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

```
TAB2 <- matrix(c(44,314,33,334),nrow = 2,byrow = T)
TAB2
```

```
##      [,1] [,2]
## [1,]   44  314
## [2,]   33  334
```

```
TAB3 <- cbind(TAB[,2],TAB[,1])
TAB3
```

```
##        [,1] [,2]
## female   44  314
## male     33  334
```

```
colnames(TAB3) <- c("yes","no")
```

```
TAB3
```

```
##        yes  no
## female  44 314
## male    33 334
```

```
epi.2by2(TAB3,method = "cohort.count")
```

```
##              Outcome +    Outcome -      Total        Inc risk *
## Exposed +           44          314        358             12.29
## Exposed -           33          334        367              8.99
## Total               77          648        725             10.62
##                   Odds
## Exposed +       0.1401
## Exposed -       0.0988
## Total           0.1188
##
## Point estimates and 95 % CIs:
## -------------------------------------------------------------------
## Inc risk ratio                               1.37 (0.89, 2.10)
## Odds ratio                                   1.42 (0.88, 2.28)
## Attrib risk *                                3.30 (-1.19, 7.79)
## Attrib risk in population *                  1.63 (-2.06, 5.32)
## Attrib fraction in exposed (%)               26.84 (-12.15, 52.28)
## Attrib fraction in population (%)            15.34 (-8.10, 33.69)
## -------------------------------------------------------------------
##  X2 test statistic: 2.077 p-value: 0.15
##  Wald confidence limits
##  * Outcomes per 100 population units
## [1] 2
```

Chapter 9 : Correlations and Covariance

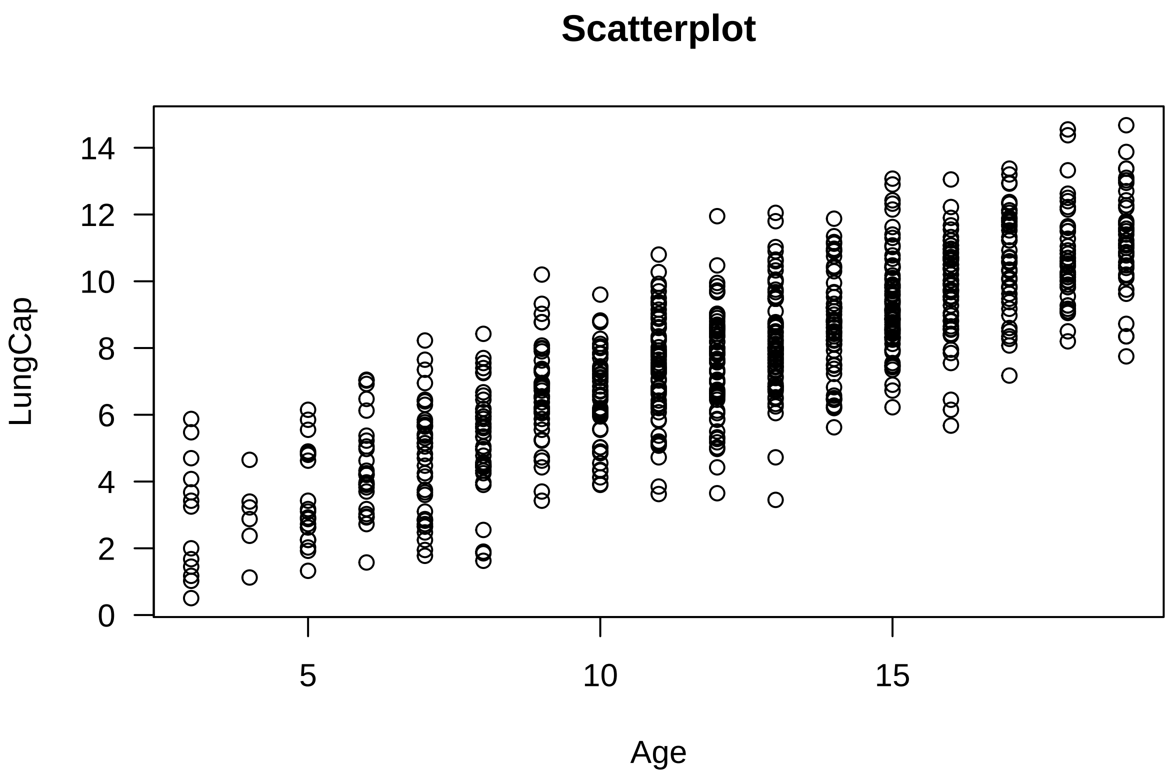
Pearson's correlation is a parametric measure of the linear association between 2 numeric variables..

Spearman's rank correlation is a non-parametric measure of the monotonic association between 2 numeric variables.

Kendall's rank correlation is another non-parametric measure of the association, based on concordance or discordance of x-y pairs..

step 1 :Explore the relationship between Age and Lung Capacity using scatterplot

```r
plot(Age,LungCap,main = "Scatterplot",las =1)
```



##

Step 2 : Calculate the correlation using cor command method= pearson is default.

```r
cor(Age,LungCap,method = "pearson")
```

```
## [1] 0.8196749
```

```r
cor(Age,LungCap,method = "spearman")
```

```
## [1] 0.8172464
```

```r
cor(Age,LungCap,method = "kendall")
```

```
## [1] 0.639576
```

```r
cor.test(Age,LungCap,method = "pearson")
```

```
##
##  Pearson's product-moment correlation
##
## data:  Age and LungCap
## t = 38.476, df = 723, p-value < 2.2e-16
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  0.7942660 0.8422217
## sample estimates:
##       cor
## 0.8196749
```

```r
cor.test(Age,LungCap,method = "spearman")
```

```
## Warning in cor.test.default(Age, LungCap, method = "spearman"): Cannot
## compute exact p-value with ties
```

```
##
##  Spearman's rank correlation rho
##
## data:  Age and LungCap
## S = 11607000, p-value < 2.2e-16
## alternative hypothesis: true rho is not equal to 0
## sample estimates:
##       rho
## 0.8172464
```

```r
cor.test(Age,LungCap,method = "spearman",exact = F)
```

```
##
##  Spearman's rank correlation rho
##
## data:  Age and LungCap
## S = 11607000, p-value < 2.2e-16
## alternative hypothesis: true rho is not equal to 0
## sample estimates:
##       rho
## 0.8172464
```

```r
cor.test(Age,LungCap,method = "pearson",alt="greater",conf.level = 0.99)
```

```
##
##  Pearson's product-moment correlation
##
## data:  Age and LungCap
## t = 38.476, df = 723, p-value < 2.2e-16
## alternative hypothesis: true correlation is greater than 0
## 99 percent confidence interval:
##  0.7891778 1.0000000
## sample estimates:
##       cor
## 0.8196749
```

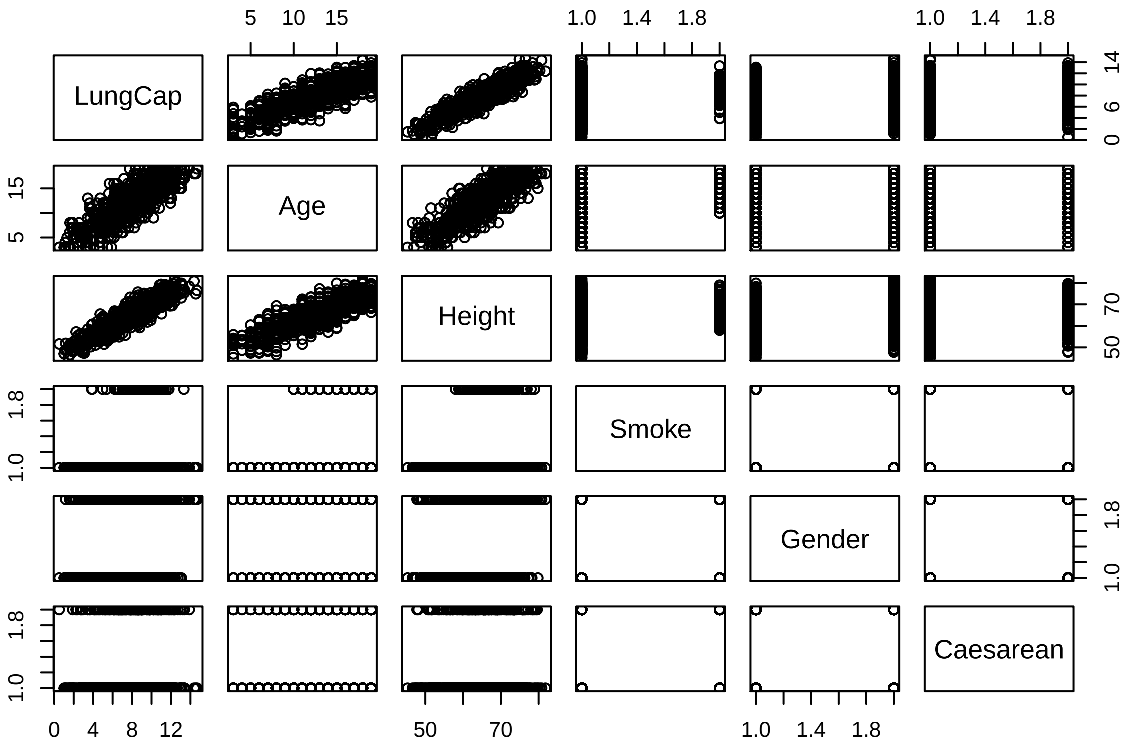
Covariance is option of less interest in applied statistics

```
cov(Age,LungCap)
```

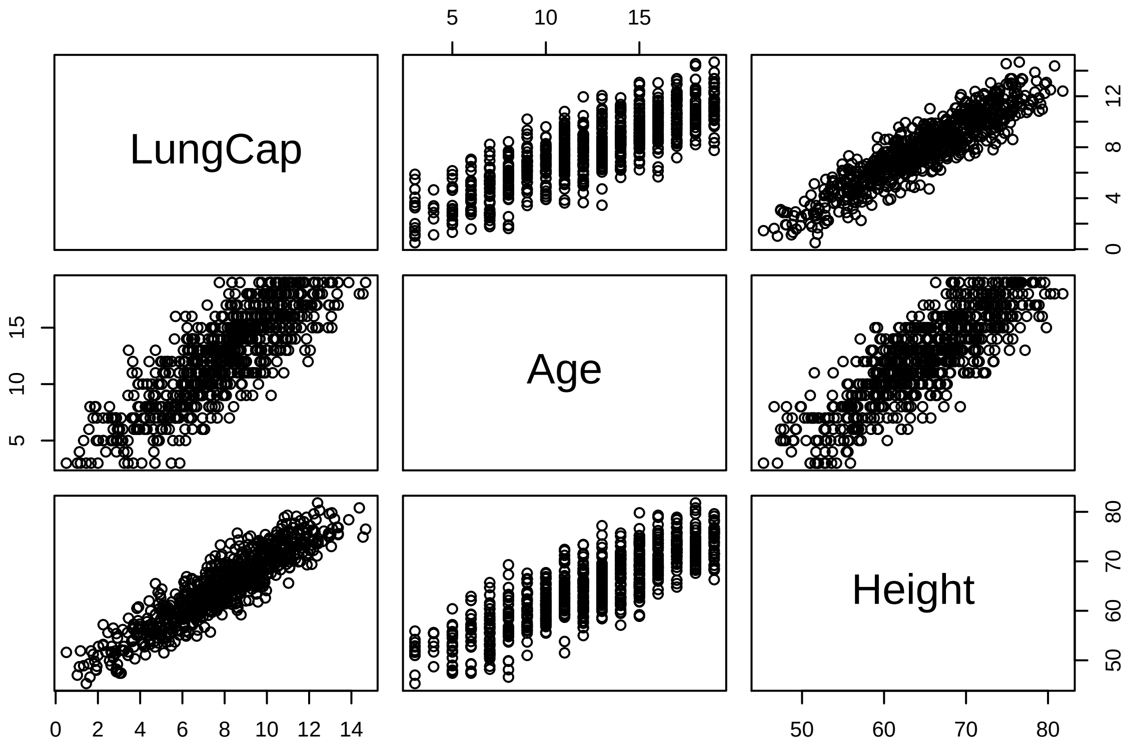
```
## [1] 8.738289
```

Producing pairs plot

```
pairs(LungCapData)
```



```
pairs(LungCapData[,1:3])
```



```r
cor(LungCapData[,1:3])
```

```
##           LungCap       Age    Height
## LungCap 1.0000000 0.8196749 0.9121873
## Age     0.8196749 1.0000000 0.8357368
## Height  0.9121873 0.8357368 1.0000000
## [1] 2
```

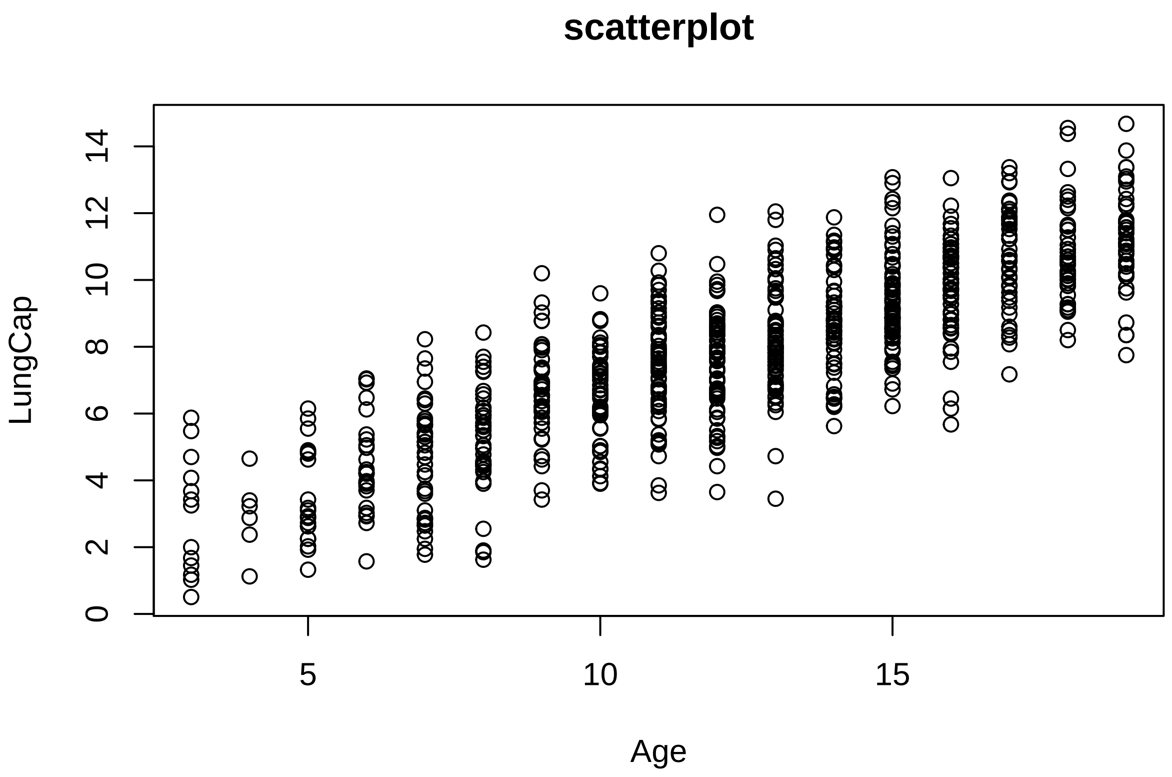
Chapter 9 : Simple Linear Regression

Simple Linear Regression is usefull for examining or modelling the relationship between 2 numeric variables.

in fact, we can also fit a simple linear regression using a categorical explanatory (x) variable. . .

STEP 1 : Model the relationship between Age and Lung Capacity, Lung Capacity is the outcome or dependent (y) variables.

```
plot(Age,LungCap,main = "scatterplot")
```



```
cor(Age,LungCap)
```

```
## [1] 0.8196749
```

Fit a linear model

```
mod <- lm(LungCap ~ Age)
```

```
summary(mod)
```

```
##
```

```
## Call:
## lm(formula = LungCap ~ Age)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -4.7799 -1.0203 -0.0005  0.9789  4.2650
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  1.14686    0.18353   6.249 7.06e-10 ***
## Age          0.54485    0.01416  38.476  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.526 on 723 degrees of freedom
## Multiple R-squared:  0.6719, Adjusted R-squared:  0.6714
## F-statistic:  1480 on 1 and 723 DF,  p-value: < 2.2e-16
```

Stars are used to identify significant coefficient.

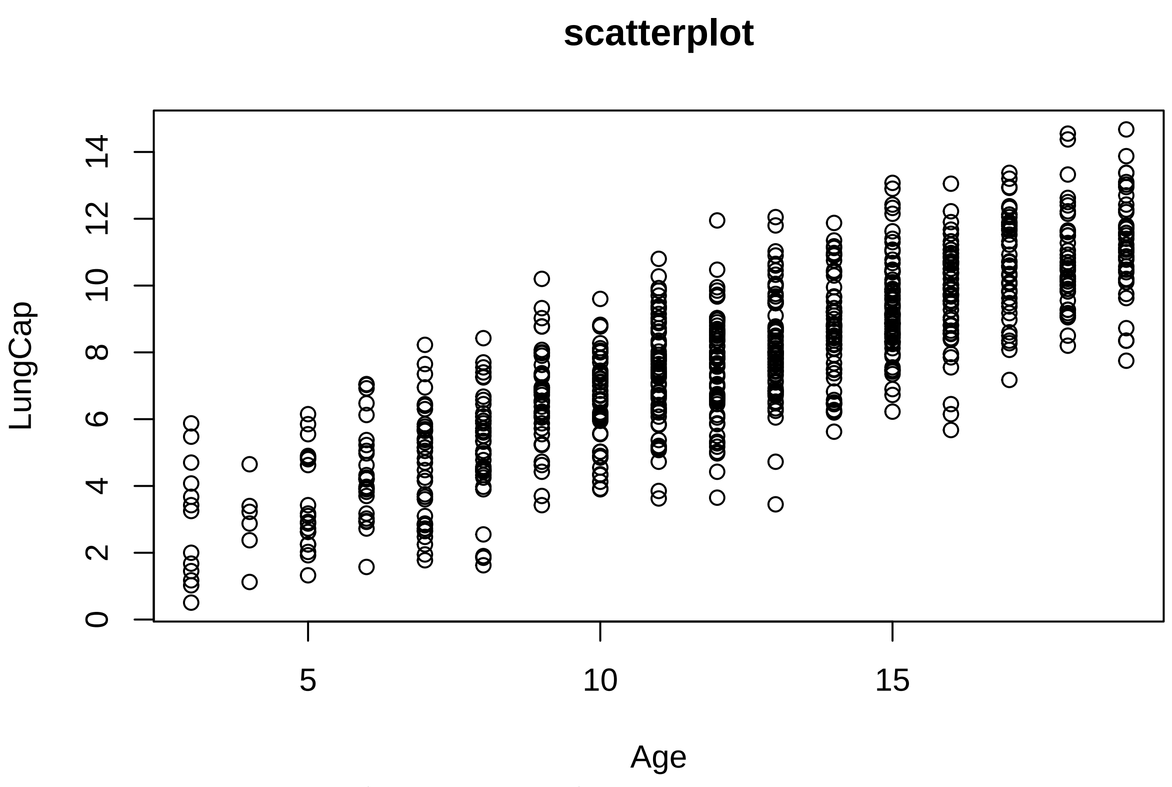
```r
attributes(mod)
```

```
## $names
##  [1] "coefficients"  "residuals"     "effects"       "rank"
##  [5] "fitted.values" "assign"        "qr"            "df.residual"
##  [9] "xlevels"       "call"          "terms"         "model"
##
## $class
## [1] "lm"
```

```r
mod$coefficients
```

```
## (Intercept)         Age
##   1.1468578   0.5448484
```

```r
plot(Age,LungCap,main = "scatterplot")
```



#regression line to the plot abline(mod,col=2,lwd=3)

```
confint(mod)
```

```
##                 2.5 %    97.5 %
## (Intercept) 0.7865454 1.5071702
## Age         0.5170471 0.5726497
```

```
anova(mod)
```

```
## Analysis of Variance Table
##
## Response: LungCap
##            Df Sum Sq Mean Sq F value    Pr(>F)
## Age         1 3447.0  3447.0  1480.4 < 2.2e-16 ***
## Residuals 723 1683.5     2.3
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
sqrt(2.3)
```

```
## [1] 1.516575
## [1] 2
```

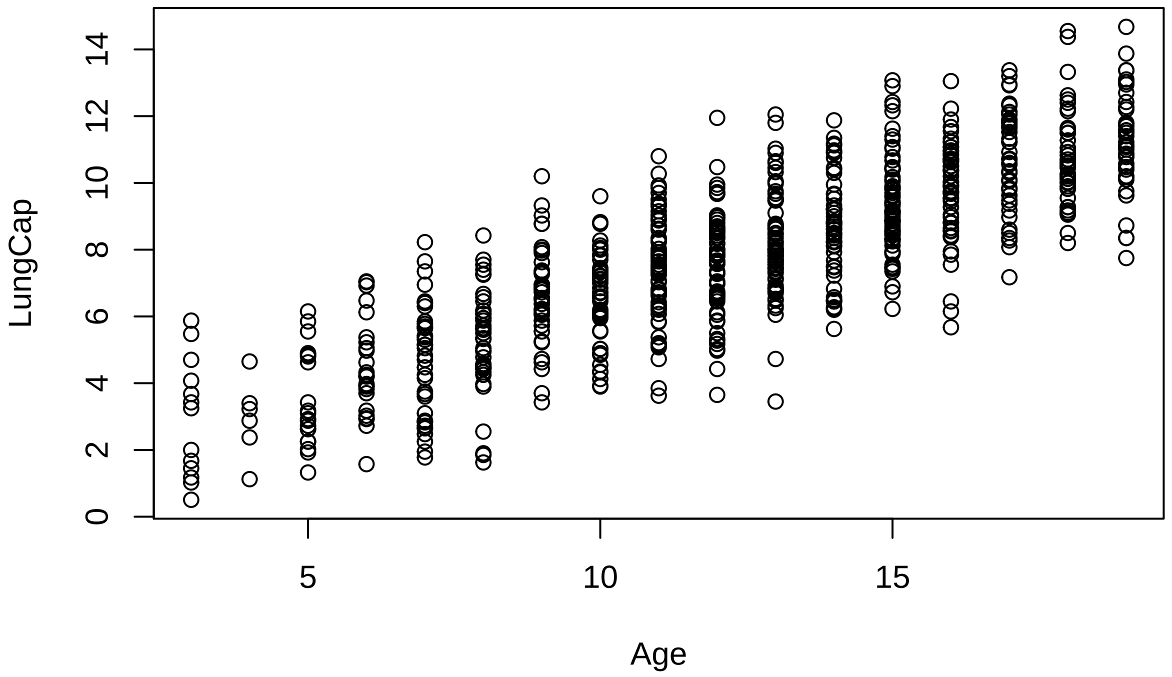
Chapter 10 : Checking Linear Regresssion Assumptions

While the assumptions of a linear model are never perfectly met, we must still check if they are reasonable assumptions to work with!

Examine the assumptions for a model of the relationship between Age and Lung Capacity !

Lung Capacity = Y, outcome, dependent Variable.

```r
plot(Age,LungCap)
```

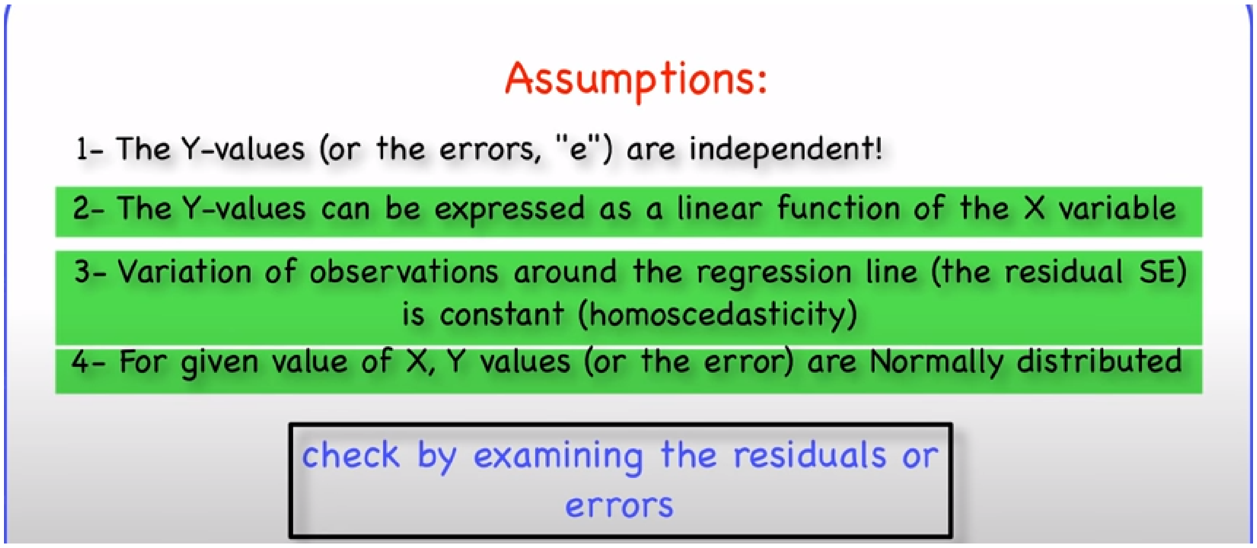


```r
mod <- lm(LungCap ~ Age)
```

```r
summary(mod)
```

```
##
## Call:
## lm(formula = LungCap ~ Age)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -4.7799 -1.0203 -0.0005  0.9789  4.2650
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  1.14686    0.18353   6.249 7.06e-10 ***
## Age          0.54485    0.01416  38.476  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.526 on 723 degrees of freedom
## Multiple R-squared:  0.6719, Adjusted R-squared:  0.6714
## F-statistic:  1480 on 1 and 723 DF,  p-value: < 2.2e-16
```

Here is the assumptions image



Assumptions:

1- The Y-values (or the errors, "e") are independent!

2- The Y-values can be expressed as a linear function of the X variable

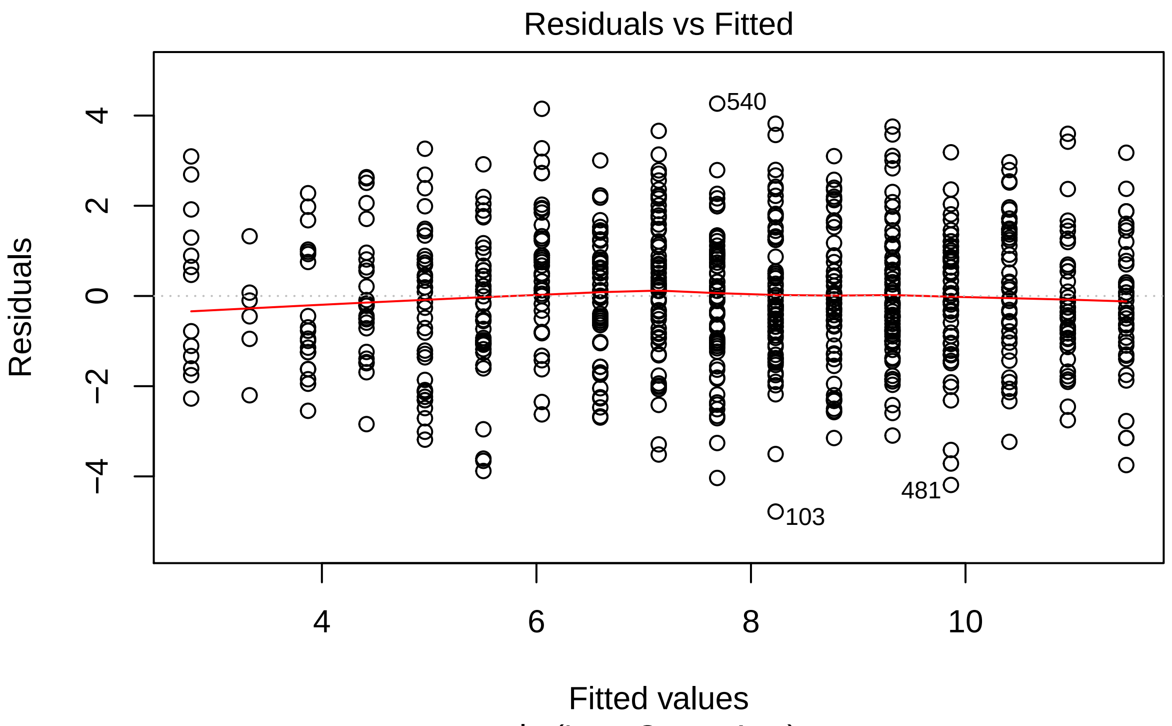
3- Variation of observations around the regression line (the residual SE) is constant (homoscedasticity)

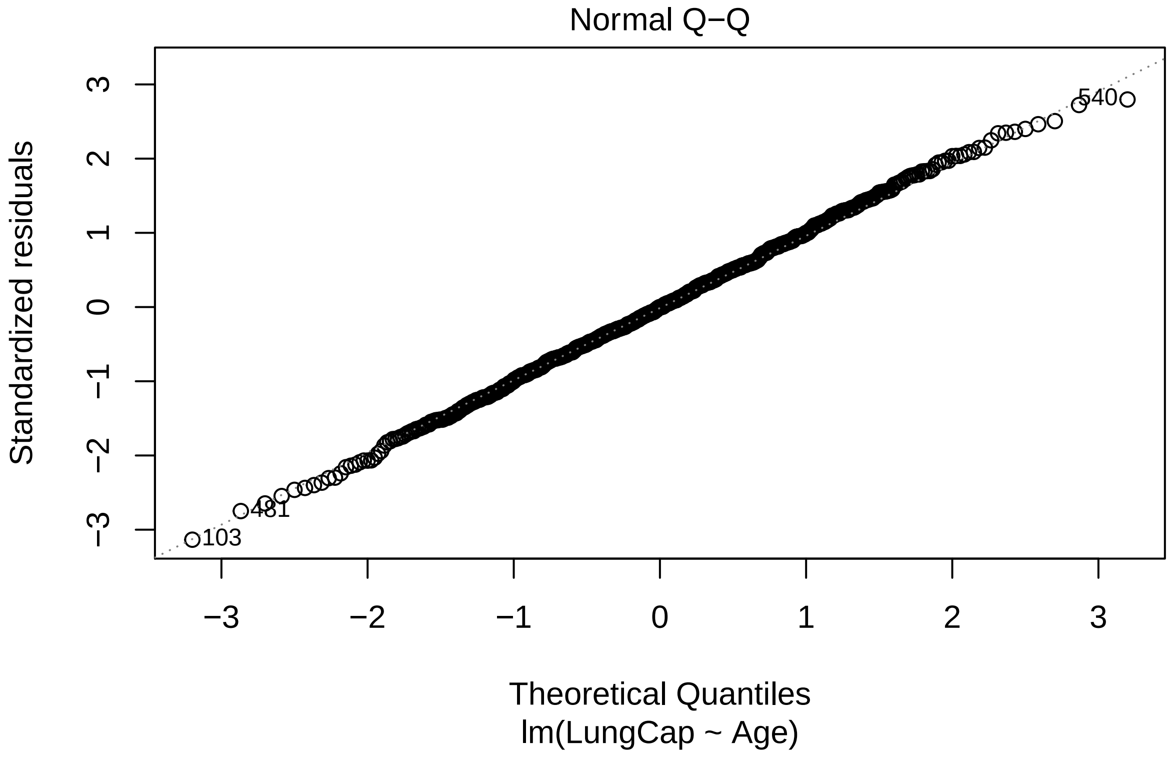
4- For given value of X, Y values (or the error) are Normally distributed

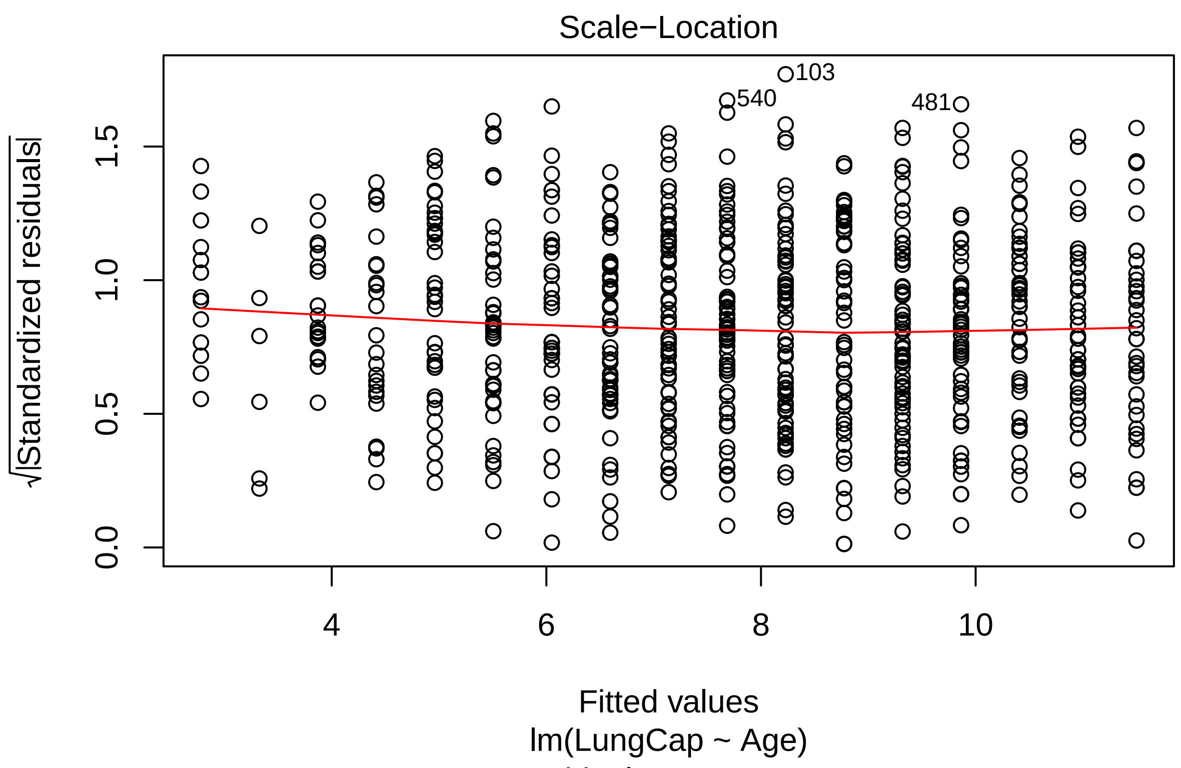
check by examining the residuals or errors

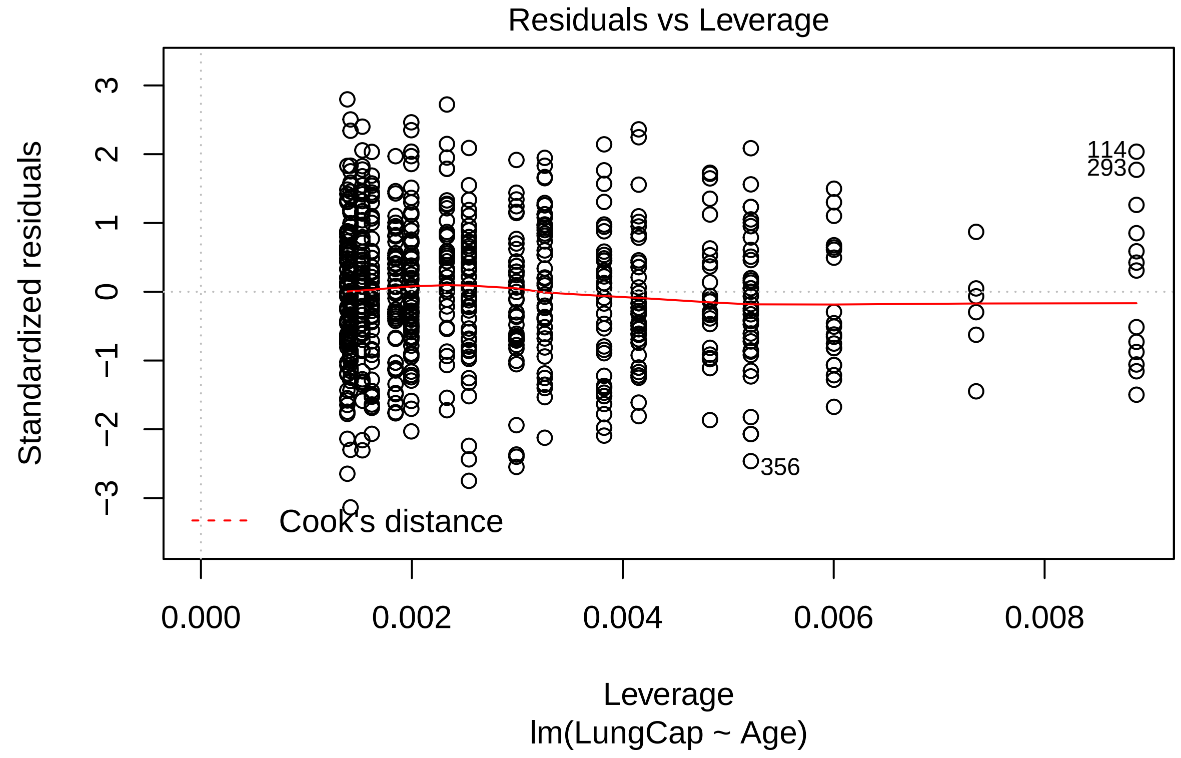
```
## [1] 2
```

```
plot(mod)
```


Residuals vs Fitted
Residuals
Fitted values
lm(LungCap ~ Age)
540
103
481
4
2
0
−2
−4
4
6
8
10


Normal Q−Q
Standardized residuals
Theoretical Quantiles
lm(LungCap ~ Age)
540
481
103
3
2
1
0
−1
−2
−3
−3
−2
−1
0
1
2
3


Scale−Location
√|Standardized residuals|
0.0
0.5
1.0
1.5
4
6
8
10
103
540
481
Fitted values
lm(LungCap ~ Age)


Residuals vs Leverage
Standardized residuals
−3
−2
−1
0
1
2
3
114
293
356
Cook's distance
0.000
0.002
0.004
0.006
0.008
Leverage
lm(LungCap ~ Age)