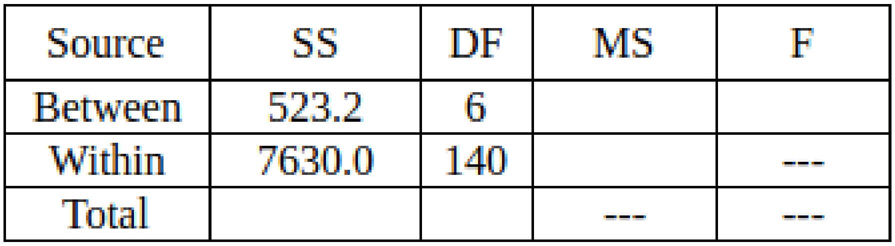
Data Science - Assignment

Amrutha E

25/03/2020

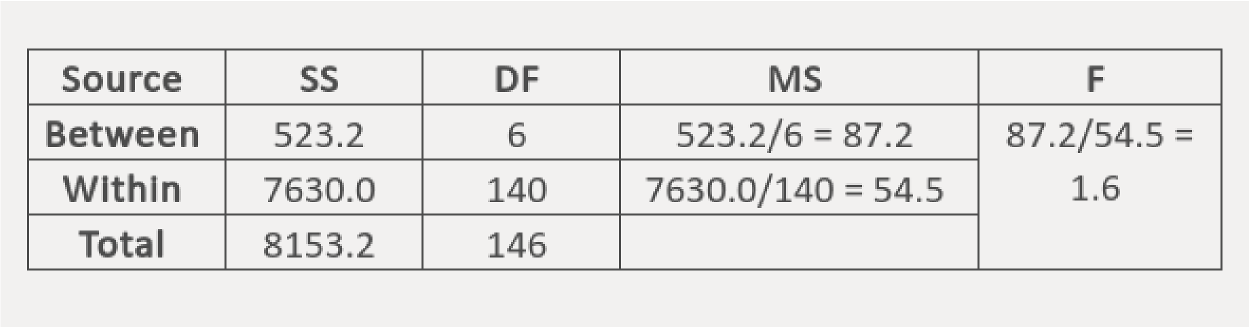
Que 1

Complete the following Generic ANOVA table:



Source	SS	DF	MS	F
Between	523.2	6		
Within	7630.0	140		---
Total			---	---

ans



Source	SS	DF	MS	F
Between	523.2	6	523.2/6 = 87.2	87.2/54.5 = 1.6
Within	7630.0	140	7630.0/140 = 54.5	
Total	8153.2	146		

Que 2

What is the difference between a linear model using glm( ) and a linear model using lm( ) in R?

ans

glm()

glm() is used to fit generalized linear models, specified by giving a symbolic description of the linear predictor and a description of the error distribution.

lm()

In R, the lm(), or "linear model," function can be used to create a simple regression model.

In R, using lm() is a special case of glm(). lm() fits models following the form Y = Xb + e, where e is Normal (0 , s^2). glm() fits models following the form f(Y) = Xb + e. However, in glm both the function f(Y) (the 'link function') and the distribution of the error term e can be specified. Hence the name - 'generalised linear model'. If you are getting the same results using both lm() and glm(), it is because for glm(), f(Y) defaults to Y, and e defaults to Normal (0, s^2). i.e. if you don't specify the link function and error distribution, the parameters that glm() uses produce the same effect as running lm().

Que 3

Solve the following Probability distribution problems using R:

1. Suppose widgit weights produced at Acme Widgit Works have weights that are normally distributed with mean 17.46 grams and variance 375.67 grams. What is the probability that a randomly chosen widgit weighs more than 19 grams?

ans

Question Rephrased : What is P(X > 19) when X has the N(17.46, 375.67) distribution? pnorm() is the R function that calculates the Cumulative distribution function R wants the s. d. as the parameter, not the variance. We'll need to take a square root!

```
1 - pnorm(19, mean=17.46, sd=sqrt(375.67))
```

```
## [1] 0.4683356
```

2. Suppose IQ scores are normally distributed with mean 100 and standard deviation 15. What is the 95th percentile of the distribution of IQ scores

ans

Question Rephrased : What is F^{-1}(0.95) when X has the N(100, 152) distribution? The qnorm function is simply the inverse of the Cumulative distribution function

```
qnorm(0.95, mean = 100, sd = 15)
```

```
## [1] 124.6728
```

Que 4

With the help of suitable examples explain attach and detach functions in R for data frames and packages

ans

The attach function allows to access variables of a data frame without calling the data frame. Create an example data frame

```
data <- data.frame(x1 = c(9, 8, 3, 4, 8),
                   x2 = c(5, 4, 7, 1, 1),
                   x3 = c(1, 2, 3, 4, 5))
data
```

```
##   x1 x2 x3
## 1  9  5  1
## 2  8  4  2
## 3  3  7  3
## 4  4  1  4
## 5  8  1  5
```

Let's assume that we want to work with the first column X1. If we try to call the column with the following code, R returns an error message:

```
# x1
# Error: object 'x1' not found
```

However, if we attach the data frame first…

```
attach(data)
x1
```

```
## [1] 9 8 3 4 8
```

```
# 9 8 3 4 8
```

After finishing the work on our data frame, it is advisable to detach the data. Otherwise, the R code might get difficult to use afterwards. We can detach our data with the following line of code:

```
detach(data)
```

After detaching, we cannot work with the X1 column as before anymore:

```
# x1
# Error: object 'x1' not found
```

Que 5

We have a sample of 30 tax accountants from all the states and territories of Australia and their individual state of origin is specified by a character vector of state mnemonics as

```
state <- c("tas", "sa", "qld", "nsw", "nsw", "nt", "wa", "wa", "qld", "vic",
           "nsw", "vic", "qld", "qld", "sa", "tas", "sa", "nt", "wa", "vic",
           "qld", "nsw", "nsw", "wa", "sa", "act", "nsw", "vic", "vic", "act")
```

We have the incomes of the same tax accountants in another vector (in suitably large units of money)

```
incomes <- c(60, 49, 40, 61, 64, 60, 59, 54, 62, 69,
             70, 42, 56, 61, 61, 61, 58, 51, 48, 65,
             49, 49, 41, 48, 52, 46, 59, 46, 58, 43)
```

ans

A factor is created using the factor() function

```
state_factor <- factor(state)
print(state_factor)
```

```
##  [1] tas sa  qld nsw nsw nt  wa  wa  qld vic nsw vic qld qld sa  tas sa  nt  wa
## [20] vic qld nsw nsw wa  sa  act nsw vic vic act
## Levels: act nsw nt qld sa tas vic wa
```

To calculate the sample mean income for each state we can now use the special function tapply()

```
income_means <- tapply(incomes, state_factor, mean)
print(income_means)
```

```
##      act      nsw       nt      qld       sa      tas      vic       wa
## 44.50000 57.33333 55.50000 53.60000 55.00000 60.50000 56.00000 52.25000
```

Que 6

Write short notes on cbind and rbind using suitable examples

ans

rbind()

The name of the rbind R function stands for row-bind. The rbind function can be used to combine several vectors, matrices and/or data frames by rows. The easiest way of using rbind in R is the combination of a vector and a data frame

```
x1 <- c(7, 4, 4, 9)                    # Column 1 of data frame
x2 <- c(5, 2, 8, 9)                    # Column 2 of data frame
x3 <- c(1, 2, 3, 4)                    # Column 3 of data frame
data_1 <- data.frame(x1, x2, x3)       # Create example data frame
print(data_1)
```

```
##   x1 x2 x3
## 1  7  5  1
## 2  4  2  2
## 3  4  8  3
## 4  9  9  4
```

and an example vector:

```
vector_1 <- c(9, 8, 7)                 # Create example vector
```

Now, let's rbind this vector to the data frame:

```
rbind(data_1, vector_1)                # rbind vector to data frame
```

```
##   x1 x2 x3
## 1  7  5  1
## 2  4  2  2
## 3  4  8  3
## 4  9  9  4
## 5  9  8  7
```

```
print(data_1)
```

```
##   x1 x2 x3
## 1  7  5  1
## 2  4  2  2
## 3  4  8  3
## 4  9  9  4
```

cbind()

The name of the cbind R function stands for column-bind. The cbind function is used to combine vectors, matrices and/or data frames by columns. A popular way of using the cbind command in the R programming language is the combination of a vector and a data.frame.

```
data_1 <- data.frame(x1 = c(7, 3, 2, 9, 0),           # Column 1 of data frame
                     x2 = c(4, 4, 1, 1, 8),           # Column 2 of data frame
                     x3 = c(5, 3, 9, 2, 4))           # Column 3 of data frame
```

and an example vector / column:

```
y1 <- c(9, 8, 7, 6, 5)                                # Create vector
```

Now, we can cbind this vector as new column to our example data frame:

```
data_new1 <- cbind(data_1, y1)                        # cbind vector to data frame
data_new1                                             # Print data to RStudio console
```

```
##   x1 x2 x3 y1
## 1  7  4  5  9
## 2  3  4  3  8
## 3  2  1  9  7
## 4  9  1  2  6
## 5  0  8  4  5
```