جلسه ی دوم:

سپیده کاردان

نرجس امین

ابتدا از قسمت open فایل R script مربوط به جلسه ی دوم را باز می‏کنیم.

فایلی که در این جلسه مورد بررسی قرار می‏گیرد یک table است که با پسوند .csv ذخیره شده است. این فایل را می‏توان با excel باز کرد. این فایل دارای ستون‏هایی نظیر شماره‏ی دانش‏آموران، رشته‎ی آن‏ها و نمرات پنج تا از درسای آن‏ها می‎باشد.

حال می‏خواهیم این داده‏‏ها را در R ایمپورت کنیم. فرمت‏های مختلفی را می‏توان در R ایمپورت کرد که متداول‏ترین آن‏ها .csv[[1]](#footnote-1) می‏باشد( البته به جز کاما می‏تواند seperator‏های دیگر نیز داشته باشد).

بهتر است فایل‏های excel را با فرمت .csv ذخیره کنیم تا بتوانیم آن‏ها را بعدا به راحتی در R باز کنیم.

دیتاها را باید از یک سری مسیرهایی(directory) بخوانیم و در یک directory هم ذخیره می‏کنیم.

با تایپ دستور getwd() در قسمت console می‏توانیم working directory را پیدا کنیم.

بهتر است در ابتدا working directory را در پوشه‏ای که در آن محتوا را داریم set کنیم. این کار را با دستور setwd(" ") می‏توان انجام داد و در داخل پرانتز بایدdirectory را که محتوای فایل .csv قرار دارد بنویسیم. به عنوان مثال:

#setting path

setwd("/Users/majid/Projects/StatsCourse/Session2") #change it

برای نوشتن دایرکتوری از علامت / و کلید Tab کمک می‏‏گیریم. بعد از هر بار قرار دادن / و زدن Tab، R به ما در تکمیل کردن دستور کمک می‏کند. یک راه ساده این است که فایل .csv را در دسکتاپ قرار داده تا به راحتی بتوانیم دایرکتوری را set کنیم.

به منظور import فایل، با توجه به این که .csv است، از دستور read.csv(" ") استفاده می‏کنیم و در داخل پرانتز نام فایل را می‏نویسیم.

importing csv table

data <- read.csv("Scores.csv",sep = ","row.names = NULL) #error

این دستور فاکتور مهمی به نام sep دارد که نشان می‏دهد عناصر این جدول با چه separator جدا شده است. فاکتور دیگر دستور فوق row.names می‏باشد که بدین معنی است که آیا می‏خواهیم یک ردیف را به عنوان اسم اختصاص دهیم یا خیر. در این جلسه برابر با NULL قرار دهیم.

در دستور فوق هنگام اجرا با خطایی مواجه می‏شویم. به منظور رفع این خطا باید پیش از فاکتور row.names یک کاما قرار دهیم. فاکتورهای مختلف هر دستور را باید با کاما از یکدیگر جدا کنیم:

#importing csv table

> data <- read.csv("Scores.csv",sep = ",",row.names = NULL) #error

به منظور مشاهده‏ی دیتا:

می‏توانیم بر روی data در environment کلیک کرده

در console، data تایپ کنیم

در قسمت script، data را انتخاب کرده و run کنیم

با دستور class(data) می‏توانیم class داده را مشخص کنیم.

class(data)

[1] "data.frame"

با دستور dim(data) می‏توان dimension داده را به دست آورد:

dim(data)

[1] 115 7

با کمک $ می‏توانیم انتخاب انجام دهیم. در این‏جا به منظور تهیه‏ی بردار از ستون Fields ستون را با دستور:

field <- data$Field

انتخاب کرده و بردار را می‏سازیم. در این‏جا محتویات بردار character هستند با دستور زیر به factor تبدیل می‏شود.

as.factor(field)

با کمک دستور tabulate می‏توانیم تعداد محتویات هر یک از رشته‏ها را بدانیم:

tabulate(as.factor(field))

[1] 56 59

انتخاب subset:

برای انتخاب از [] استفاده می‏کنیم. در این‏جا یک جدول داریم. به منظور انتخاب یک بخش از جدول باید ردیف و ستون مد نظر را داخل براکت مشخص کنیم. به این منظور از کاما استفاده می‏کنیم. به این صورت که هر چه قبل از کاما می‏آید ردیف و هر چه بعد از کاما می‏آید ستون می‏باشد. اگر بخواهیم چند ردیف یا ستون را انتخاب کنیم در قسمت مخصوص خودشان از اندیس c(combine) استفاده می‏کنیم:

‏data1 <- data[c(1,2,10:20),]

به منظور حذف تعدادی ردیف یا ستون می‏توان از علامت c– استفاده کرد:

data2 <- data[-c(1,2,10:25),]

برای انتخاب تعدادی از ستون‏ها می‏توان از c(combine) استفاده کرد و اندیس ستون‏های مدنظر را در آن قرار داد:

data3 <- data[,c(1,2,3)]

یک مدل دیگر انتخاب بر اساس اسم ستون است. در این حالت یک بردار می‏سازیم و نام ستون‏های مد نظر را با c کامباین می‏کنیم:

data3 <- data[,c("Stud","Field","Course1")]

از فانکشن subset نیز می‏توان به منظور انتخاب کردن ردیف‏های بخشی از داده‏ها استفاده کرد. به عنوان مثال برای انتخاب Psych :

data4 <- subset(data,data$Field == "Psych")

در دستور فوق می‎توان شروط مختلفی را وارد کرد. مثلا می‏توان یک subset گرفت که در آن داده‏های NA را مشخص کرد:

data7 <- subset(data,is.na(data$Course3) )

class سابست‏های گرفته شده نیز با دستور class() به دست‏ می‏آوریم و مشاهده می‏کنیم که class آن‏ها data.frame می‏باشد. به منظور ذخیره دیتافریم از دستور زیر استفاده می‏کنیم:

write.csv()

به عنوان مثال به منظور ذخیره‏ی data7 به این صورت عمل می‏کنیم:

write.csv(data7,"data7.csv")

**NA Handeling:**

برای به دست آوردن تعداد داده‏های not available می‏توان از دستور sum() کمک گرفت.

sum( is.na(data) )

برای این‏که مشخص کنیم کدام یک از داده‏ها not available می‏باشد، از دستور which کمک می‏گیریم( که اندیس داده را از بالا به پایین می‏شمارد و مشخص می‏کند.)

which( is.na(data) )

با استفاده از آرگومان arr.ind=T می‏توان ردیف و ستون داده‏ی not available را مشخص کرد.

which( is.na(data) ,arr.ind = T)

برای حذف داده‏ی not available می‏توان ابتدا آن‏ها را به یک موجودی assign کنیم و سپس آن‏ها را حذف کنیم.

به عنوان مثال برای حذف دانشجویانی که در این دیتاست داده‏ی NA داشتند از دستور زیر استفاده می‏کنیم:

cond <- which( is.na(data) ,arr.ind = T)

which( is.na(data) ,arr.ind = T)

row col

[1,] 10 3

[2,] 87 3

[3,] 78 4

[4,] 99 4

[5,] 4 5

[6,] 29 5

[7,] 78 5

[8,] 102 5

[9,] 78 7

data1 <- data[-cond[,1],]

که در این‏جا با توجه به این‏که هدف حذف کامل دانشجو با داده‏ی NA است، باید row را که ستون اول cond است مد نظر قرار دهیم و -cond[,1] بنویسیم.

1:01:40

**"Narjes Amin”**

میتوانیم دیتا دیگری بسازیم که مثل دیتا یک هست و هیچ فرقی ندارد، حالا ما می‌تونیم دیتا ۲ یا تعدادی از ردیف‌ها را حذف کنیم و یا اینکه یک مقدار به na بدهیم، مثلاً فرض می‌کنیم دانش آموزان غیبت کرده اند و برای غیبت به آن ها صفر میدهیم. (کافیه سلکشن رو داشته باشیم و داخلش یک شرط بنویسیم مثلاً بگیم دیتای ۲ همه جاهایی که na هست را برابرشون قرار می‌دیم با صفر)  
#52،53

data2 <- data

data2[ is.na(data2) ] <- 0 #change it

وقتی قرار است تک تک کورس‌ها را ببینیم و آن ها را بررسی کنیم و هیستگرام آن ها را رسم کنیم. از دستور 55# استفاده میکنیم.

#histogram inspection

hist(data$Course1)

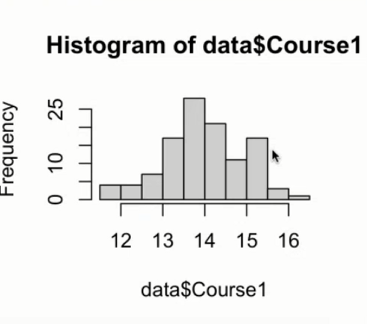
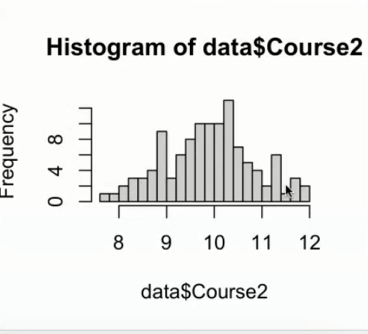
تعریف هیستوگرام: در نموداری که محور افقی نشان دهنده نمرات است، هیستوگرام نشان می دهد که چه تعداد از افراد در آن بازه نمره کسب کرده اند. که تعداد افراد بر روی نمودار Y قابل مشاهده است.

هیستوگرام با تعدادی مستطیل خاص نمایش داده می‌شود که ما می‌توانیم تعداد این مستطیل‌ها را زیاد و یا کم کنیم، با این کار ما می‌توانیم رزولوشن بیشتر یا کمتری از نمایش دیتا داشته باشیم.

hist(data$Course2, breaks = 20) #change it

#descriptive statistics

hist(data$Course1)



برای به دست آوردن میانگین در یک major ما همه نمرات را با هم جمع می‌کنیم و تقسیم بر تعداد می‌کنیم تا میانگین نمرات به دست بیاید. برای این هدف از دستور 63 استفاده میکنیم.

mean(data$Course1)

اگردر اجرای دستور mean با پاسخ na مواجه شدیم، بهترین استراتژی اسفاده از کد ۶۵ است که در واقع از این طریق naها را remove می‌کند.

mean(data$Course1,na.rm = True) #error

#sum(is.na(data$Course1))

#sum(data$Course1,na.rm = T) / 113

از طریق دستور median شما می‌توانید اعداد میانی تمام ستون‌های مختلف دیتایتان را حساب کنید.

median(data$Course1,na.rm = T)

در دستور ۷۲ ما دیتاهایمان را در کورس ۳ sort می‌کنیم، و اعداد وسط یعنی مدین را به دست می‌آوریم.

sort(data$Course3)

hist(data$Course1)

mean(data$Course1,na.rm = T)

median(data$Course1,na.rm = T)

hist(data$Course2) #change it

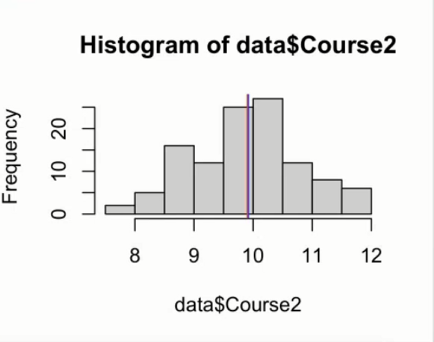
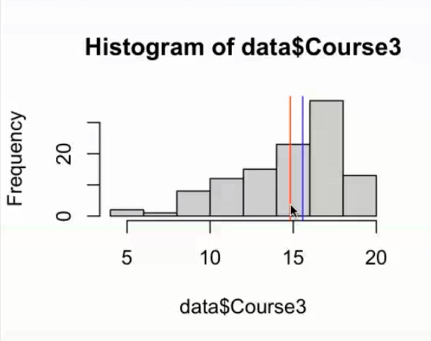
mean(data$Course2,na.rm = T) #change it

median(data$Course2,na.rm = T) #change it

81، 82، دستور abline می تواند خطی افقی یا عمودی بر روی نمودار رسم کند، (به طور مثال در دستور عنوان می کنیم که جایی که میانگین من قرار دارد با رنگ قرمز و جایی که median من قرار دارد با رنگ آبی یک خط عمودی رسم کن)

abline(v=mean(data$Course2,na.rm = T),col="red") #change it

abline(v=median(data$Course2,na.rm = T),col="blue") #change it



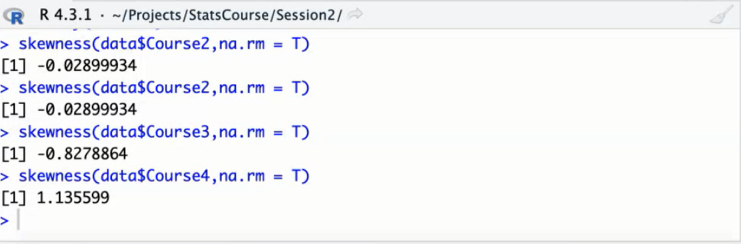
در کورس ۲ خط‌های قرمز و آبی تقریباً روی هم افتادند ولی در کورس ۳ آنها با فاصله از هم قرار دارند این نشان دهنده این است که در کورس ۲ ما یک توزیع نرمال داریم اما در کورس ۳ skewness و یا کشیدگی منفی است این یعنی اینکه نمرات بالا هستند.

skewness(data$Course2,na.rm = T)

86 87 برای محاسبه skewness ما نیاز داریم تا پکیج moments را نصب کنیم.

library(moments)

skewness(data$Course2,na.rm = T)



دستور 93 مربوط به معیار واریانس یا معیار پراکندگی می‌باشد.

#

hist(data$Course1) #change it

var(data$Course1,na.rm = T) #change it

#mu <- mean(data$Course1,na.rm = T)

#sum((data$Course1 - mu)^2,na.rm = T) / 112

بر اساس دستور standard deviation، می‌توانیم پراکندگی دیتا را محاسبه کنیم چرا که برای محاسبه معیار پراکندگی بهتر از واریانس عمل می‌کند، به این دلیل که ابتدا داده را به توان ۲ می‌رساند و بعد از آن جذر می‌گیرد. هر چقدر مقدار پراکندگی یا standard deviation و یا واریانس بیشتر باشد نشان دهنده این است که پراکندگی دیتا بیشتر است.

sd(data$Course1,na.rm = T) #change it

#mu <- mean(data$Course1,na.rm = T)

#sqrt(sum((data$Course1 - mu)^2,na.rm = T)) / sqrt(112)

در این دستور ما می‌توانیم ببینیم که رنج اعدادمان از چند شروع می‌شود و تا چند ادامه دارد. (کورس ۱ از ۱۱ و ۸ هست ۱۶ و ۱۹ )

range(data$Course1,na.rm = T) #change it

دستور ۱۰۷ برای اجرای quantile می‌باشد. quantile دستوری است که دیتاهای ما را به ۲۵ درصد کوچکتر و ۷۵ درصد بزرگتر، یا ۵۰ درصد کوچکتر و ۵۰ درصد بزرگتر (که median است)، و یا 75 درصد کوچکتر و ۲۵ درصد بزرگتر تقسیم می‌کند.

quantile(data$Course1,0.5,na.rm = T)

median(data$Course1,na.rm = T)

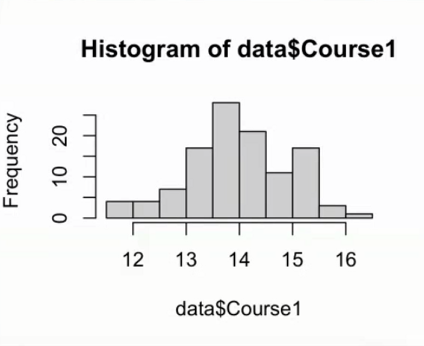
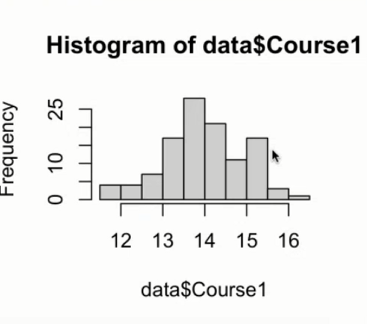
quantile(data$Course1,0.25,na.rm = T)

quantile(data$Course1,0.75,na.rm = T)

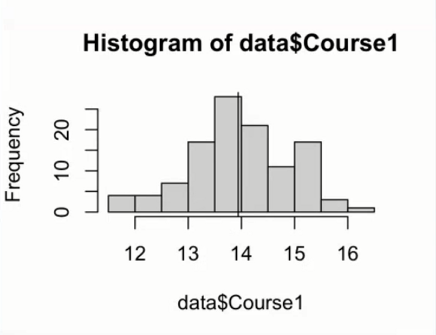
117 دستور ۱۱۷ برای اجرای اینتر quantileرنج است. (به طور مثال اختلاف ۷۵ درصد از ۲۵ درصد)

IQR(data$Course1,na.rm = T) #change it

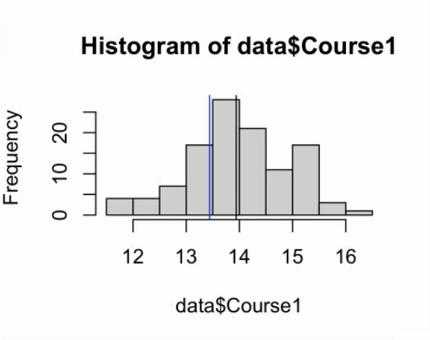
* **نمودارها:**

112 هیستوگرام  


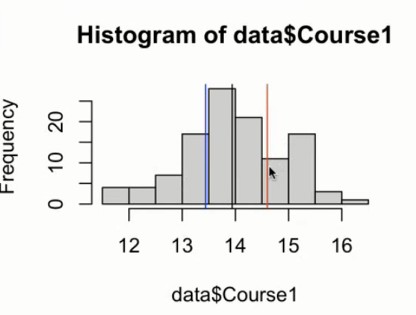
113 median



114 کوانتایل 25 صدم



115 کوانتایل 75 صدم



دستور kurtosis مقدار شارپ بودن نمودار را نشان می‌دهد، هر چقدر تابع توزیع شارپ‌تر باشد مثبت و هر چقدر تخت تر بشه منفی است.

kurtosis(data$Course5,na.rm = T) #change it

**پایان**

1. Comma separated value [↑](#footnote-ref-1)