Untitled

STAT613\_HW#2

J Dickens

2/13/2021

library(tidyverse)

## -- Attaching packages --------------------------------------- tidyverse 1.3.0 --

## v ggplot2 3.3.2 v purrr 0.3.4  
## v tibble 3.0.1 v dplyr 1.0.0  
## v tidyr 1.1.2 v stringr 1.4.0  
## v readr 1.4.0 v forcats 0.5.0

## -- Conflicts ------------------------------------------ tidyverse\_conflicts() --  
## x dplyr::filter() masks stats::filter()  
## x dplyr::lag() masks stats::lag()

library(dplyr)  
library(purrr)

#QUES 1): #Enter and run USArrests. What type of information is shown in the data table USArrests? (Three or four sentences) #Answer 1): #The data table consists of 50 rows and 4 columns. #Columns (Murder, Rape) type is ‘dbl’. #Columns (Assault, UrbanPop) type is ‘int’. #Both formats ‘dbl’ and ‘int’ are for real numbers

USArrests

## Murder Assault UrbanPop Rape  
## Alabama 13.2 236 58 21.2  
## Alaska 10.0 263 48 44.5  
## Arizona 8.1 294 80 31.0  
## Arkansas 8.8 190 50 19.5  
## California 9.0 276 91 40.6  
## Colorado 7.9 204 78 38.7  
## Connecticut 3.3 110 77 11.1  
## Delaware 5.9 238 72 15.8  
## Florida 15.4 335 80 31.9  
## Georgia 17.4 211 60 25.8  
## Hawaii 5.3 46 83 20.2  
## Idaho 2.6 120 54 14.2  
## Illinois 10.4 249 83 24.0  
## Indiana 7.2 113 65 21.0  
## Iowa 2.2 56 57 11.3  
## Kansas 6.0 115 66 18.0  
## Kentucky 9.7 109 52 16.3  
## Louisiana 15.4 249 66 22.2  
## Maine 2.1 83 51 7.8  
## Maryland 11.3 300 67 27.8  
## Massachusetts 4.4 149 85 16.3  
## Michigan 12.1 255 74 35.1  
## Minnesota 2.7 72 66 14.9  
## Mississippi 16.1 259 44 17.1  
## Missouri 9.0 178 70 28.2  
## Montana 6.0 109 53 16.4  
## Nebraska 4.3 102 62 16.5  
## Nevada 12.2 252 81 46.0  
## New Hampshire 2.1 57 56 9.5  
## New Jersey 7.4 159 89 18.8  
## New Mexico 11.4 285 70 32.1  
## New York 11.1 254 86 26.1  
## North Carolina 13.0 337 45 16.1  
## North Dakota 0.8 45 44 7.3  
## Ohio 7.3 120 75 21.4  
## Oklahoma 6.6 151 68 20.0  
## Oregon 4.9 159 67 29.3  
## Pennsylvania 6.3 106 72 14.9  
## Rhode Island 3.4 174 87 8.3  
## South Carolina 14.4 279 48 22.5  
## South Dakota 3.8 86 45 12.8  
## Tennessee 13.2 188 59 26.9  
## Texas 12.7 201 80 25.5  
## Utah 3.2 120 80 22.9  
## Vermont 2.2 48 32 11.2  
## Virginia 8.5 156 63 20.7  
## Washington 4.0 145 73 26.2  
## West Virginia 5.7 81 39 9.3  
## Wisconsin 2.6 53 66 10.8  
## Wyoming 6.8 161 60 15.6

#QUES 2): #Use and show R coding that features a map function to show maximum values for all variables of the USArerests data frame. Which State has the largest number of Assaults according to the USAressts data frame ? #Answer 2):  
#Maximum values for all variables:

map\_dbl(USArrests, max)

## Murder Assault UrbanPop Rape   
## 17.4 337.0 91.0 46.0

#North Carolina (Row 33) displays 337 (maximum number of Assaults)

which.max(USArrests$Assault)

## [1] 33

USArrests %>%   
 slice\_max(Assault)

## Murder Assault UrbanPop Rape  
## North Carolina 13 337 45 16.1

#QUES 3): #Install the nycflights13 package: install.packages(“nycflights13”) , call the following library: library(nycflights13), and then enter flights (this will produce the flights data table)

library(nycflights13)  
flights

## Warning: `...` is not empty.  
##   
## We detected these problematic arguments:  
## \* `needs\_dots`  
##   
## These dots only exist to allow future extensions and should be empty.  
## Did you misspecify an argument?

## # A tibble: 336,776 x 19  
## year month day dep\_time sched\_dep\_time dep\_delay arr\_time sched\_arr\_time  
## <int> <int> <int> <int> <int> <dbl> <int> <int>  
## 1 2013 1 1 517 515 2 830 819  
## 2 2013 1 1 533 529 4 850 830  
## 3 2013 1 1 542 540 2 923 850  
## 4 2013 1 1 544 545 -1 1004 1022  
## 5 2013 1 1 554 600 -6 812 837  
## 6 2013 1 1 554 558 -4 740 728  
## 7 2013 1 1 555 600 -5 913 854  
## 8 2013 1 1 557 600 -3 709 723  
## 9 2013 1 1 557 600 -3 838 846  
## 10 2013 1 1 558 600 -2 753 745  
## # ... with 336,766 more rows, and 11 more variables: arr\_delay <dbl>,  
## # carrier <chr>, flight <int>, tailnum <chr>, origin <chr>, dest <chr>,  
## # air\_time <dbl>, distance <dbl>, hour <dbl>, minute <dbl>, time\_hour <dttm>

#QUES 4): #Use and show R code that will indicate how many rows and how many columns the flights data has. Review and revisit your notes from STAT 412/612. Describe a tibble (two or three sentences). Now use and show R code that verifies that flights is a tibble. #Answer 4): #No. of Rows= 336776; No. of Columns= 19

ncol(flights)

## [1] 19

nrow(flights)

## [1] 336776

#Answer: #Tibbles are a type of data frames/data set representation #characterized by the display(print) of the first ten rows and all #of the variables that can be displayed in the console with each #column types. #They are widely used in Tidyverse and preferred as they #make data cleaner and tidier.

is\_tibble(flights)

## [1] TRUE

#Yes, the result “True” indicates that the table is #a tibble.

#QUES 5): #Now Use and show R code (featuring a map function) that will output the type of each column of the flights tibble.

str(flights)

## tibble [336,776 x 19] (S3: tbl\_df/tbl/data.frame)  
## $ year : int [1:336776] 2013 2013 2013 2013 2013 2013 2013 2013 2013 2013 ...  
## $ month : int [1:336776] 1 1 1 1 1 1 1 1 1 1 ...  
## $ day : int [1:336776] 1 1 1 1 1 1 1 1 1 1 ...  
## $ dep\_time : int [1:336776] 517 533 542 544 554 554 555 557 557 558 ...  
## $ sched\_dep\_time: int [1:336776] 515 529 540 545 600 558 600 600 600 600 ...  
## $ dep\_delay : num [1:336776] 2 4 2 -1 -6 -4 -5 -3 -3 -2 ...  
## $ arr\_time : int [1:336776] 830 850 923 1004 812 740 913 709 838 753 ...  
## $ sched\_arr\_time: int [1:336776] 819 830 850 1022 837 728 854 723 846 745 ...  
## $ arr\_delay : num [1:336776] 11 20 33 -18 -25 12 19 -14 -8 8 ...  
## $ carrier : chr [1:336776] "UA" "UA" "AA" "B6" ...  
## $ flight : int [1:336776] 1545 1714 1141 725 461 1696 507 5708 79 301 ...  
## $ tailnum : chr [1:336776] "N14228" "N24211" "N619AA" "N804JB" ...  
## $ origin : chr [1:336776] "EWR" "LGA" "JFK" "JFK" ...  
## $ dest : chr [1:336776] "IAH" "IAH" "MIA" "BQN" ...  
## $ air\_time : num [1:336776] 227 227 160 183 116 150 158 53 140 138 ...  
## $ distance : num [1:336776] 1400 1416 1089 1576 762 ...  
## $ hour : num [1:336776] 5 5 5 5 6 5 6 6 6 6 ...  
## $ minute : num [1:336776] 15 29 40 45 0 58 0 0 0 0 ...  
## $ time\_hour : POSIXct[1:336776], format: "2013-01-01 05:00:00" "2013-01-01 05:00:00" ...

#QUES 6):  
#Use and show R coding that features usage of a map function to find the slope and the intercept of models for the different levels of the Species variable of the iris data frame. For each model, Sepal.Width predicts Sepal.Length.

#Level of Species variable (setosa): #Intercept: 2.6390 #Slope: 0.6905

#Level of Species variable (versicolor): #Intercept: 3.5397 #Slope: 0.8651

#Level of Species variable (virginica): #Intercept: 3.9068 #Slope: 0.9015

model1 <- iris %>%  
 split(.$Species) %>%  
 map(~lm(Sepal.Length ~ Sepal.Width, data = .))  
model1

## $setosa  
##   
## Call:  
## lm(formula = Sepal.Length ~ Sepal.Width, data = .)  
##   
## Coefficients:  
## (Intercept) Sepal.Width   
## 2.6390 0.6905   
##   
##   
## $versicolor  
##   
## Call:  
## lm(formula = Sepal.Length ~ Sepal.Width, data = .)  
##   
## Coefficients:  
## (Intercept) Sepal.Width   
## 3.5397 0.8651   
##   
##   
## $virginica  
##   
## Call:  
## lm(formula = Sepal.Length ~ Sepal.Width, data = .)  
##   
## Coefficients:  
## (Intercept) Sepal.Width   
## 3.9068 0.9015

#QUES 7):

V <- list(12, 22, 27, 31.5, NA, 39, "east")  
V

## [[1]]  
## [1] 12  
##   
## [[2]]  
## [1] 22  
##   
## [[3]]  
## [1] 27  
##   
## [[4]]  
## [1] 31.5  
##   
## [[5]]  
## [1] NA  
##   
## [[6]]  
## [1] 39  
##   
## [[7]]  
## [1] "east"

#For the given list above; #7a) Use and show R code to find the length of the list

length(V)

## [1] 7

#7b) Use and show R code that will extract the missing value

V[is.na(V)]

## [[1]]  
## [1] NA

#7c) Use and show R code that will extract the third object

V[3]

## [[1]]  
## [1] 27

#7d) Use and show R code that will extract the character string and the minimum number.

V[c(7,1)]

## [[1]]  
## [1] "east"  
##   
## [[2]]  
## [1] 12

#7e) Use one line of code to show that the seventh object is a character object.

str(V[7])

## List of 1  
## $ : chr "east"