Week 2 Exercises

Hayley Morway

March 26, 2023

Please complete all exercises below. You may use stringr, lubridate, or the forcats library.

Place this at the top of your script: library(stringr) library(lubridate) library(forcats)

Exercise 1

Read the sales_pipe.txt file into an R data frame as sales.

```
#load libraries and create df from loading sales_pipe file
library(stringr)
library(lubridate)
```

```
## Warning: package 'lubridate' was built under R version 4.2.3
```

```
##
## Attaching package: 'lubridate'
```

```
## The following objects are masked from 'package:base':
##
##     date, intersect, setdiff, union
```

```
library(forcats)
```

```
## Warning: package 'forcats' was built under R version 4.2.3
```

```
sales_pipe_df <- read.delim(file= "Data/sales_pipe.txt", sep="|",
                            header=TRUE , stringsAsFactors = FALSE,
                            check.names = FALSE)
```

Exercise 2

You can extract a vector of columns names from a data frame using the colnames() function. Notice the first column has some odd characters. Change the column name for the FIRST column in the sales date frame to Row.ID.

Note: You will need to assign the first element of colnames to a single character.

```
#rename column 1 header to Row.ID
colnames(sales_pipe_df)[1] <- "Row.ID"
```

Exercise 3

Convert both Order.ID and Order.Date to date vectors within the sales data frame. What is the number of days between the most recent order and the oldest order? How many years is that? How many weeks?

Note: Use lubridate

```
#convert Order.Date to vector of Dates
sales_pipe_df$Order.Date.Format <- mdy(sales_pipe_df$Order.Date)

#create variables for minimum and maximum Order dates
order_max_date <- max(sales_pipe_df$Order.Date.Format)
order_min_date <- min(sales_pipe_df$Order.Date.Format)

#then subtract min from max to get the difference between the two dates
difference_dates <- order_max_date - order_min_date

#then convert date difference to number of days, save in variable and print.
#Repeat for weeks and years
days_between_all_orders<- time_length(difference_dates,"day")
days_between_all_orders
```

```
## [1] 1457
```

```
weeks_between_all_orders <- time_length(difference_dates, "week")
weeks_between_all_orders
```

```
## [1] 208.1429
```

```
years_between_all_orders <- time_length(difference_dates, "year")
years_between_all_orders
```

```
## [1] 3.989049
```

Exercise 4

What is the average number of days it takes to ship an order?

```
#convert Ship.Date to vector of Dates
sales_pipe_df$Ship.Date.Format <- mdy(sales_pipe_df$Ship.Date)

#find difference in days between order and shipment dates, then convert to #integers and store in df
sales_pipe_df$Shipping.Days <- time_length((sales_pipe_df$Ship.Date.Format
                                            - sales_pipe_df$Order.Date.Format),
                                           "day")

#find mean of vector
mean(sales_pipe_df$Shipping.Days)
```

```
## [1] 3.908482
```

Exercise 5

How many customers have the first name Bill? You will need to split the customer name into first and last name segments and then use a regular expression to match the first name bill. Use the length() function to determine the number of customers with the first name Bill in the sales data.

```
#split first and last name column into 2 columns, then paste them to exisitng
#df and rename them.
customer_name <- str_split_fixed(string=sales_pipe_df$Customer.Name, pattern=" ", n=2)
sales_pipe_df$Customer.First.Name <- paste(customer_name[,1])
```

```
sales_pipe_df$Customer.Last.Name <- paste(customer_name[,2])

#Then use regex to determine if first name is Bill.
#Then use logical indexing to find all the "Bill"s then use
#length to determine the number of elements in the vector.
matched_names <- str_match(sales_pipe_df$Customer.First.Name, pattern= "Bill")
length(which(matched_names == "Bill"))
```

```
## [1] 37
```

Exercise 6

How many mentions of the word 'table' are there in the Product.Name column? **Note you can do this in one line of code**

```
# find "table" in column using reg expression matching, and take length of
# which matches
length(which(str_match(sales_pipe_df$Product.Name, pattern = "table")== "table"))
```

```
## [1] 197
```

Exercise 7

Create a table of counts for each state in the sales data. The counts table should be ordered alphabetically from A to Z.

```
#create a table of states and their counts, and sort alphabetically by name
table(sort(sales_pipe_df$State, decreasing = FALSE))
```

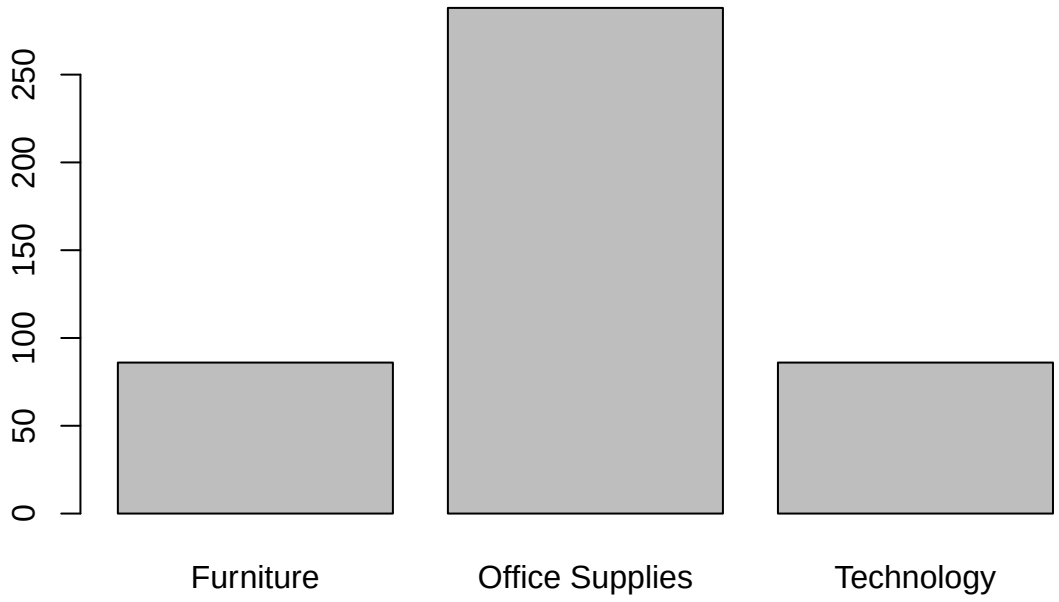
```
##
##              Alabama              Arizona             Arkansas
##                   28                  119                   22
##           California             Colorado          Connecticut
##                  993                   90                   50
##             Delaware District of Columbia              Florida
##                   47                    1                  186
##              Georgia                Idaho             Illinois
##                   79                    9                  286
##              Indiana                 Iowa               Kansas
##                   74                   11                   16
##             Kentucky            Louisiana                Maine
##                   64                   18                    4
##             Maryland        Massachusetts             Michigan
##                   63                   71                  142
##            Minnesota          Mississippi             Missouri
##                   41                   27                   37
##              Montana             Nebraska               Nevada
##                    2                   26                   24
##        New Hampshire           New Jersey           New Mexico
##                    9                   58                   11
##             New York       North Carolina         North Dakota
##                  555                  117                    7
##                 Ohio             Oklahoma               Oregon
##                  211                   38                   56
```

```
##         Pennsylvania         Rhode Island       South Carolina
##                  312                   25                   28
##         South Dakota            Tennessee                Texas
##                    9                   88                  460
##                 Utah              Vermont             Virginia
##                   27                   10                   80
##           Washington        West Virginia            Wisconsin
##                  254                    4                   38
##              Wyoming
##                    1
```

Exercise 8

Create an alphabetically ordered barplot for each sales Category in the State of Texas.

```
#create a df for sales in Texas, then a table for the categories listed
#alphabetically, and a barplot of the table
sales_catergoies_texas_df <- sales_pipe_df[(sales_pipe_df$State== "Texas") ,]
barplot(table(sort(sales_catergoies_texas_df$Category, decreasing = FALSE)))
```



Exercise 9

Find the average profit by region. **Note: You will need to use the aggregate() function to do this. To understand how the function works type ?aggregate in the console.**

```r
#aggregate the Profit column, listing by Region, using the mean function
aggregate(sales_pipe_df$Profit, list(sales_pipe_df$Region), FUN=mean)
```

```
##   Group.1        x
## 1 Central 20.46822
## 2    East 29.91937
## 3   South 11.27720
## 4    West 32.77000
```

Exercise 10

Find the average profit by order year. **Note: You will need to use the aggregate() function to do this. To understand how the function works type ?aggregate in the console.**

```r
#extract year from order date and add year to df as new column
Order.Year <- str_split_fixed(string=sales_pipe_df$Order.Date.Format, pattern="-", n=3)
sales_pipe_df$Order.Year <- paste(Order.Year[,1])

#then aggregate profits by year, using the mean function
aggregate(sales_pipe_df$Profit, list(sales_pipe_df$Order.Year), FUN=mean)
```

```
##   Group.1        x
## 1    2014 32.24582
## 2    2015 21.58676
## 3    2016 30.10960
## 4    2017 21.31825
```