Week 2 Exercises

Patrick Sheehan

March 20, 2024

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.

For this exercise I specified the file path where I have the sales_pipe.txt and loaded it in using the read.delim function.

```r
# Your code here
sales <- read.delim("C:/DSE5002/Week_2/Data/sales_pipe.txt"
                     ,stringsAsFactors=FALSE
                     ,sep = "|"
                     ,fileEncoding="WINDOWS-1252"
 )
```

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.

Here I called the stringr library and then used colnames() to make the columns into a vector. I then pulled the first indice and renamed it to Row.ID.

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

```r
# Your code here
library(stringr)
colnames(sales)[1] <- 'Row.ID'
```

Exercise 3

Convert both Ship.Date 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?

Here I used as.Date to convert both the Ship.Date and Order.Date columns into date vectors. I then used the difftime function to subtract the min order date from the max order date to get 1457 days between the most recent and oldest order. I then took this and divided it by dyears/dweeks to get the number of days in years/weeks. 3.989 years and 208.143 weeks.

Note: Use lubridate

```
# Your code here
library(lubridate)
```

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

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

```
sales$Ship.Date <- as.Date(sales$Ship.Date
                           , format ='%B %d %Y')

sales$Order.Date <- as.Date(sales$Order.Date
                           , format= '%m/%d/%Y')

fulfillment_time <- difftime(max(sales$Order.Date),min(sales$Order.Date))


print(fulfillment_time) ## Difference in Days
```

```
## Time difference of 1457 days
```

```
fulfillment_time / dyears(1) ##Difference in Years
```

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

```
fulfillment_time / dweeks(1) ##Difference in Weeks
```

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

Exercise 4

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

Here I took difference between the time it takes to ship by subtracting the order date by the ship date and then took the average of that result. After that I divided it by ddays(1) to get the duration in days.

```
# Your code here
(mean(difftime((sales$Ship.Date),(sales$Order.Date))))/ddays(1)
```

```
## [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.

Here I created a dataframe that showed me all unique customer names. I then used str_split_fixed to split the unique_names and show me only the first name. I then used str_which to look only for Bill values and used length to determine the number of customers with Bill as their first name.

```
# Your code here
library(stringr)

unique_names <- unique(sales$Customer.Name)
```

```
first_name <- str_split_fixed(unique_names, pattern = ' ', n=3)
length(str_which(first_name, "Bill"))
```

```
## [1] 6
```

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**

Here I used str_which to find the indices of the word ‘table’ in the Product.Name column of sales. I then used length to count the total number of occurrences.

```
# Your code here

library(stringr)

length(str_which(sales$Product.Name, '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.

Here I used the table function to create a count for each state in the sales data. Before doing this I made the state column into factors so that it would be in alphabetical order.

```
# Your code here
sales$State <- factor(sales$State)
table(sales$State)
```

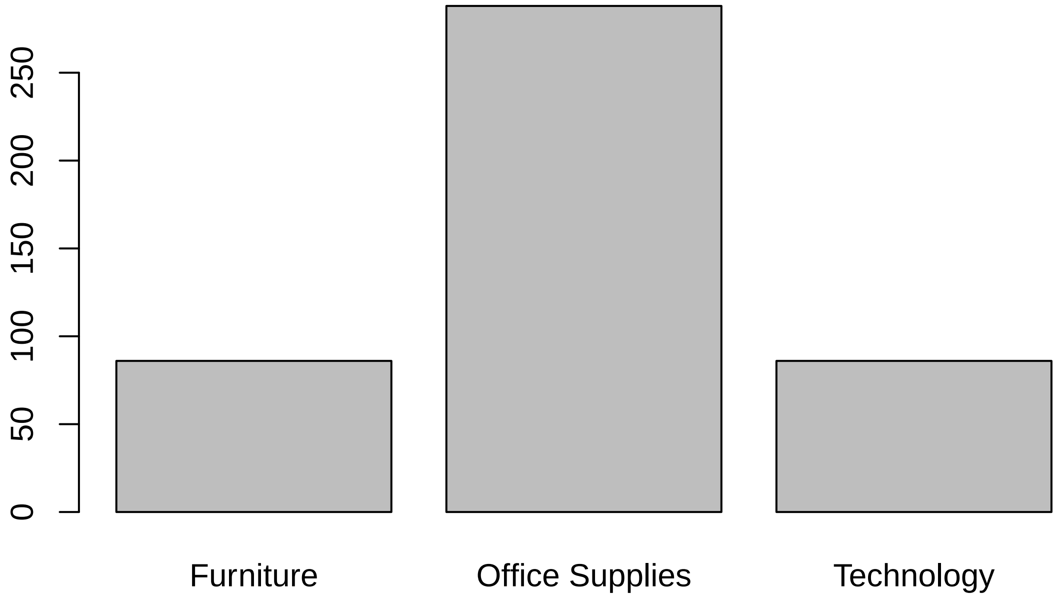
```
## 
##              Alabama              Arizona             Arkansas 
##                   28                  119                   22 
##           California             Colorado          Connecticut 
##                  993                   90                   50 
##             Delaware District 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.

Here I created a dataframe that only included the state of texas. I then turned the category column of the texas_df into a factor so that it would be ordered alphabetically. Lastly I used the table function on the texas_df category column to be able to use the barplot function and display the number of sales by category in Texas.

```r
# Your code here
texas_df <- sales[sales$State == 'Texas',]
texas_df$Category <- factor(texas_df$Category)
barplot(table(texas_df$Category))
```



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.**

Here I used the aggregate function to get the mean of the sales profit categorized by region.

```
# Your code here
aggregate(sales$Profit, list(sales$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.**

Here I used the aggregate function to get the mean of the sales profit categorized by year. I needed to wrap the order date in a year function to show the year and then make it into a list in order to work in the aggregate function.

```
# Your code here
aggregate(sales$Profit, list(year(sales$Order.Date)), FUN=mean)
```

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