---

title: "Practical 2"

author: "BHOJ BHAT CHHETRI"

date: "`r Sys.Date()`"

output: pdf\_document

---

```{r setup, include=FALSE}

knitr::opts\_chunk$set(echo = TRUE,

message = FALSE,

warning = FALSE)

# load libraries

library(dplyr)

library(ggplot2)

library(readr)

library(stringr)

library(tidyr)

library(tidyverse)

library(psych)

library(gsubfn)

# import the data file using the readr package

az\_data <- readr::read\_csv("az\_products\_practical.csv")

```

## Purpose

The purpose of this practical is to assess student mastery of verifying the correctness of their code.

## Instructions

For each of the following questions, provide your written answer in complete sentences in the text area of the R Markdown report and put your code in the code chunks. Unless otherwise specified, set your code chunk options to have the code and output shown but no messages or warnings. You may not ask anyone but Dr. Lapato for clarification or help. If you use any online resources, you must list those resources in a bulleted list beside the relevant question. Failure to do so may constitute an Honor code violation.

## Submission

Practical 2 is due by November 19th at 11:59PM. Submit your Rmd and rendered PDF as a zip file to Canvas. Late submissions will lose 1% credit for each hour late.

## Question 0

Please type out the VCU Honor Code and then type your name."On my honor, I have done these all the practial\_2 myself,neither given nor receive any help from others on this assignment. I pledge that I am in compliance with the VCU Honor system". Bhoj Bhat Chhetri

## Question 1 (20 points)

A csv file and a data dictionary have been provided by a collaborator. These files are from a dataset of Amazon product reviews. Import the dataset into R (show your code in this R Markdown report) and read over the data dictionary. As you know, data dictionaries give analysts information about the dataset that can be used during exploratory data analysis to assess data quality control.

Name two specific data quality checks you might perform on this dataset based on information provided in the data dictionary and what you see when you look at the dataset. Specify what information in the data dictionary prompted you to make this check and name the function(s) you would use to make the evaluation.

\*\*Specific data quality check# 1: The data dictionary states that, reviewer's ID (arin-variable ) is in unique form and it should not be repeated,So I would use the group\_by() and summarize() from the dplyr package to calculate the `n` number to check the non repetition value for the quality check.\*\*

```{r}

az\_data %>%

group\_by(arin) %>%

summarise(count = n())

```

\*\*Specific data quality check# 2:The data dictionary explained that the rating should be numeric(1-5),So I choose group\_by(),filter() and summarize (), functions from the dplyr package to count the total missing NA value in the rating variable.There should not be the missing value in rating variable in quality check.\*\*

```{r}

az\_data %>%

group\_by (rating) %>%

filter (is.na(rating)) %>%

summarise(count=n())

```

\*\*Rubric\*\*

\* Two correct specific data quality checks are mentioned (5 points)

\* Correct methods for performing these checks are provided (10 points)

\* Data is imported correctly (5 points)

```

Example of an answer using the `babynames` dataset:

```

The data dictionary states that the dataset only "includes names with at least 5 uses", so I would use `summarize()`from the dplyr package to calculate the minimum `n` for the dataset to verify that no values less than 5 are present for the count variable.

## Question 2 (30 points)

There are many ways to clean a character string variable. Below, there are four approaches one could use to clean a numeric variable that happens to include some string-like characters (e.g., $).

\*\*Part A (15 points)\*\*

Use `dplyr::mutate` to create four new variables in your combined dataset that clean the `price` variable (one new variable for each method). Please name your variables: `price\_numeric`, `price\_parse`, `price\_stringr`, and `price\_gsub`. Remember that proper tidyverse code does not use dollar signs for variable selection. One method has been coded for you as an example.

\* Method 1 (simple base R): `as.numeric(az\_data\_combined$price)`

\* Method 2 (tidyverse, readr): `readr::parse\_number(az\_data\_combined$price)`

\* Method 3 (tidyverse, stringr): `as.numeric(stringr::str\_match(az\_data\_combined$price, "[0-9]+"))`

\* Method 4 (base R, regex): `as.numeric(gsub('[$,]', '', az\_data\_combined$price))`.

\*\*(Part:A) Answer: Four new variables from the price variable giving new data name ie, az\_combined\_data is given below\*\*.

source: [dplyr.tidyverse](https://dplyr.tidyverse.org/reference/mutate.html)

```{r}

az\_data\_combined <- az\_data %>% # Creating the four new variables using the mutate()

select(price) %>%

mutate(price\_numeric = as.numeric(az\_data$price),

price\_parse = parse\_number(az\_data$price),

price\_stringr = as.numeric(stringr::str\_match(az\_data$price, "[0-9]+")),

price\_gsub = as.numeric(gsub('[$]','', az\_data$price)))

```

\*\*Part B (15 points)\*\*

Add code annotations that describe how each method is cleaning the `price` variable. One method has been annotated for you as an example.

\*\*Move the example code and annotation into a proper code chunk and adjust the code chunk options so that only the code shows in the knitted report (i.e., no dataset printout or summary).\*\*

\*\*Rubric\*\*

\* New variables are added correctly (15 points)

\* Code annotations are correct (5 points)

\* Code annotations are specific (5 points)

\* Code shows but the dataset does not (5 points)

```

<DATASET> %>%

mutate(

price\_gsub = as.numeric(gsub('[$,]', '', price)) # This method removes dollar

signs from a string by replacing the dollar sign with an empty string.

)

```

# Part B: Answer: The code and annotation of the different method is given below:

```

az\_data %>%

mutate(as.numeric(az\_data\_combined$price)) # This method convert the factors(categorical value)

# into numeric value present in the price column.

# [as-numeric](https://r-lang.com/as-numeric-r/)

```

```

az\_data %>%

mutate(readr::parse\_number(az\_data\_combined$price)) # This methods drops any non numeric characters before the

# first numeric and after first numeric like "",$ and % from all the price column values.

[parse\_number](https://readr.tidyverse.org/reference/parse\_number.html)

```

az\_data %>%

mutate (as.numeric(stringr::str\_match(az\_data\_combined$price, "[0-9]+"))) # This method abstract the first part of the string # values that matches the groups and patterns

# defined by the groups between the numeric values from the price column. [str\_tidyverse](https://stringr.tidyverse.org/reference/str\_match.html)

```

```

## Question 3 (20 points)

Create a table that summarizes key descriptive of each cleaning method. Your table must be created with a function (not manually) and should include the minimum value, maximum value, and mean value for each cleaning method. Round all numeric values to two decimal places. You do not have to use the tidyverse to earn full credit; however, folks who are able to create a table with the structure described below using only tidyverse functions will be eligible for up to 30 bonus points.

The table structure that must be used to earn full credit and any bonus points is 1) the cleaning methods as column names, and 2) the descriptive statistic as a row. The order of the summary statistics (rows) and cleaning methods (columns) do not matter. `999` was used below to illustrate where numeric values would

go.

```{r}

method\_summary <- function(x) {

x <- x %>%

select(-price)

x %>%

summarize\_at(vars(1:4),

funs(mean = round(mean(., na.rm= TRUE),2), min= round(min(., na.rm=TRUE),2),

max= round(max(., na.rm=TRUE),2))) %>%

pivot\_longer(1:12) %>%

mutate(Statistic = str\_extract(name,"\_mean|\_min|\_max"),

Statistic = str\_sub(Statistic, 2, -1),

name =str\_remove(name,"\_mean|\_min|\_max")) %>%

pivot\_wider(names\_from = name,

values\_from = value)

}

method\_summary(az\_data\_combined)

```

# resources:

1.[gtsummary](https://www.danieldsjoberg.com/gtsummary/reference/tbl\_summary.html)

2.[pivot table](https://rstudio-conf-2020.github.io/r-for-excel/pivot-tables.html)

3.[stackoverflow](https://stackoverflow.com/questions/49770222/gsub-inside-function-with-mutate-called-in-dplyr-chain-gives-error)

4.[satckoverflow](https://stackoverflow.com/questions/72300676/r-round-a-data-frame-with-a-specific-formula-across-all-columns-with-dplyr)

5.[satckoverflow](https://stackoverflow.com/questions/25759891/dplyr-summarise-each-with-na-rm)

6.[stackoverflow](https://stackoverflow.com/questions/26596540/functionx-in-r-writing-a-function-without-defining-a-function)

7.[stringrtidyverse](https://stringr.tidyverse.org/reference/str\_remove.html)

| Statistic | numeric | readr | stringr | gsub |

|----|----|----|----|

| mean | 999 | 999 | 999 | 999 |

| min | 999 | 999 | 999 | 999 |

| max | 999 | 999 | 999 | 999 |

Thirty (30) bonus points are available for correctly using the following functions in your calculation and table creation (partial extra credit is possible):

\* summarize\_at() [10 points]

\* pivot\_longer() [5 points]

\* pivot\_wider() [5 points]

\* names\_pattern argument in pivot\_longer() [10 points]

You have all of the tools you need to earn these bonus points in the examples provided in the class slides, the R help documentation, and the online [`dplyr` resource](https://dplyr.tidyverse.org/reference).

\*\*Rubric\*\*

\* Table structure is correct (10 points)

\* Table values are correct (10 points)

## Question 4 (20 points)

Explain why there are differences in the summary statistics for each method. To earn full credit, you must explicitly link how the method wrangles data to how that action impacts what values are altered or set to missing.

\*\*Answer: The first method produced invalid data with (inf)for each summary statistic, this is because the as.numeric function will only convert numeric values stored as strings ie (1-9) to numbers, because the original values contained dollars signs as well as commas, this function cannot convert those values to numerics and renders the result NA when non-numeric symbols are in the original values.\*\*

\*\*The next method parse number appeared to give the most robust summary statistics as each value had the correct decimals, and there were no cut off values as observed in the other methods (999), the parse\_numbers function is specifically written for dealing with this type of situation where original values may need to be cleaned in several ways to produce a final correct numeric value, in this case that involved dealing with both dollar signs and commas.\*\*

\*\*The string R method gives slightly different summary statistics because it appears to just be parsing that appear before the decimal point or a comma, this is why it is maxing out at 999 because original values with commas are converted to the number that appears before the comma, this combined with the lack of decimals slightly throws off the summary statistic calculation.\*\*

\*\*The final method returns correct minimum values but the max is thrown off because it has the same issue that it does not take commas into consideration and turns values with commas into NAs.\*\*

\*\*Rubric\*\*

\* At least three specific differences in output are discussed (5 points)

\* The discussion links method to impact (15 points)

<br>

## Question 5 (60 points)

\*\*Part A.\*\* (20 points)

Which method is best? Explain your rationale. Be sure to consider how well each method preserves the data (e.g., How much information does the method lose?).

\*\*Answer: The parse\_number method appears to be best method it is tied with stringr for least number of NAs returned (6,696) but we know that the data returned by parse\_numbers is of better quality because it preserves decimals and deals correctly with numbers with commas (numbers over 1000). The as.numeric method returns 100,000 NAs and the gsub method returns 7,836.\*\*

\*\*Rubric\*\*

\* Student selects a reasonable method as "best" (5 points)

\* Student provides specific evidence supporting their choice (15 points)

\*\*Part B.\*\* (30 points)

Were any raw data values lost (i.e., set to missing) by all four methods? If you believe that there were raw data values lost (i.e., set to missing) by all four methods, how do you know? What caused these values to be set to missing by all four methods? How problematic is this loss of data?

If you believe that there no raw data values were lost (i.e., set to missing) by all four methods, how do you know? Which method lost the least amount of data?

\*\*Answer:To figure out answer to this question, I filtered NA values across all four output columns. It appears that the only observations that were NA across all methods were ones that had original values as NA, which was the case in 6,696 observations. This mean that if values that were originally NA were filtered out, there were no observations and that were lost across all four methods. The parse\_numbers method is still the method that lost the least amount of data because its numerical summaries are correct whereas, and data related to decimals or numbers over 1,000 are not lost.\*\*

\*\*Rubric\*\*

\* Student correctly states if any raw data values or formats were lost (10 points)

\* Student provides a thorough explanation of their reasoning (15 points)

\* Student discusses either impact of data loss or correctly names which method lost the least amount of data.

\*\*Part C.\*\* (10 points)

Is there a method or a combination of methods that would preserve the data better? Explain your reasoning.

\*\*Answer: Among the four different methods,the parse\_number method appears to be the Method that best preserves the data, this is because summary statistics, an analysis of NA values and different values between the other methods shows that the output data retains a high fidelity to the original values and issues such as dollar signs, commas, and decimals are all cleaned and parsed correctly.\*\*

\*\*Rubric\*\*

\* Student provides a thoughtful, reasonable answer supported by evidence (10 points)