Homework 1

Ashwin Malshe

07 September 2021

This homework uses `tech_co_cstat.dta` which we used in weeks 2 and 3.
The data set consists of the following variables:

```
##  [1] "gvkey"    "datadate" "fyear"    "indfmt"   "consol"   "popsrc"
##  [7] "datafmt"  "tic"      "cusip"    "conm"     "curcd"    "fyr"
## [13] "at"       "capx"     "che"      "cogs"     "csho"     "cshpri"
## [19] "cshr"     "dlc"      "dltt"     "dvc"      "dvt"      "ebit"
## [25] "ebitda"   "emp"      "gdwl"     "lct"      "ni"       "oancf"
## [31] "oiadp"    "oibdp"    "opiti"    "pi"       "ppent"    "re"
## [37] "rect"     "revt"     "sale"     "seq"      "xad"      "xlr"
## [43] "xrd"      "xsga"     "exchg"    "cik"      "costat"   "naicsh"
## [49] "sich"     "prcc_c"   "mkvalt"   "prcc_f"   "add1"     "add2"
## [55] "add3"     "add4"     "addzip"   "busdesc"  "city"     "conml"
## [61] "county"   "ggroup"   "gind"     "gsector"  "gsubind"  "naics"
## [67] "sic"      "state"    "ipodate"
```

Read `tech_co_cstat_dta.zip` into your R session using `read_dta()` function from `haven` package. Store the resulting object in `d1`.

Before you start working on this homework, study the variables in `tech_co_cstat_dta.zip` as well as the structure of the dataset by typing these commands in your console:

```
psych::describe(d1)
dplyr::glimpse(d1)
```

Take a peek at the data by typing:

```
head(d1)
```

Read the attributes of any variable from this data set using `attributes` function. For example, the attributes of `gvkey` can be printed using:

```
attributes(d1$gvkey)
```

```
## $label
## [1] "Global Company Key"
##
## $format.stata
## [1] "%6s"
```

Finally, before you begin, include only the rows with `sale > 0`.

```
d2 <- filter(d1, sale > 0)
```

This homework consists of 8 questions. Q1 carries 1 point. Q2 through Q8 carry two points each. We use `d2` as the initial input. Your objective is to reproduce the output shown in the HTML file for Q1 through Q9.

Q1

Print a data frame with the medians of `cogs`, `emp`, and `xrd`.

```
## # A tibble: 1 x 3
##    cogs   emp   xrd
##   <dbl> <dbl> <dbl>
## 1  9162  36.3  4475
```

Q2

Print a data frame with the means of `sale`, `oibdp`, and `xrd` for Apple, Facebook, and Tesla. For this, you will need to follow these steps:

1. Filter only the observations pertaining to Apple, Facebook, and Tesla
2. Group by `conm`
3. Summarize across `sale`, `oibdp`, and `xrd` to get their means
4. Output it as data frame by using `as.data.frame()` function.

```
##           conm       sale      oibdp       xrd
## 1    APPLE INC 196525.000 62428.7273 8818.7273
## 2 FACEBOOK INC  29984.636 14952.8182 6074.6364
## 3    TESLA INC   9666.025   705.4621  777.9678
```

Q3

Round all the numeric variables in the above data frame to 1 decimal place. Output as a data frame using `as.data.frame()` function.

For rounding, you will have to use `mutate`, `across`, and `where` functions from `dplyr` package. Check https://www.tidyverse.org/blog/2020/04/dplyr-1-0-0-colwise/ for more information.

```
##           conm     sale   oibdp    xrd
## 1    APPLE INC 196525.0 62428.7 8818.7
## 2 FACEBOOK INC  29984.6 14952.8 6074.6
## 3    TESLA INC   9666.0   705.5  778.0
```

Q4

Many advertising values are missing. The missing code in R is `NA`. We can get the total number of missing values for advertising quite easily by running the following function:

I urge you to understand this piece of code. It is a quick way to get a sense of number of missing values for any given variable in the data frame.

```
sum(is.na(d2$xad))
```

```
## [1] 19
```

In the finance literature, a common (but incorrect) practice is to assume that the missing advertising is 0. We will use this adjustment to `xad` and create a new variable `adv` and save it in a new object `d3`.

My own research shows that this is highly misleading. However, my solution to this issue is complex and requires application of machine learning.

The first six values of `d3` when `xad` is NA are as follows:

```
## # A tibble: 6 x 4
##   conm        datadate     xad   adv
##   <chr>       <date>     <dbl> <dbl>
## 1 APPLE INC   2016-09-30    NA     0
## 2 APPLE INC   2017-09-30    NA     0
## 3 APPLE INC   2018-09-30    NA     0
## 4 APPLE INC   2019-09-30    NA     0
## 5 APPLE INC   2020-09-30    NA     0
## 6 TWITTER INC 2011-12-31    NA     0
```

Q5

Using `d3`, create the following variables and print first 8 rows for Netflix and the new columns along with `conm` and `datadate`:

1. Return on assets (`roa`) = `oibdp` / `at`
2. Free cash flow (`fcf`) = `oancf` / `che`
3. Strategic emphasis (`strat_emph`) = (`adv` - `xrd`) / `at`

```
## # A tibble: 8 x 5
##   conm        datadate      roa     fcf strat_emph
##   <chr>       <date>      <dbl>   <dbl>      <dbl>
## 1 NETFLIX INC 2010-12-31 0.321   0.789    0.0500
## 2 NETFLIX INC 2011-12-31 0.140   0.398    0.0131
## 3 NETFLIX INC 2012-12-31 0.0241  0.0304   0.0121
## 4 NETFLIX INC 2013-12-31 0.0511  0.0815   0.0109
## 5 NETFLIX INC 2014-12-31 0.0647  0.0102   0.00861
## 6 NETFLIX INC 2015-12-31 0.0361 -0.324    0.00622
## 7 NETFLIX INC 2016-12-31 0.0322 -0.850   -0.000714
## 8 NETFLIX INC 2017-12-31 0.0479 -0.633    0.00202
```

Q6

You want to know how many profitable years each of the sample company experienced. For this follow these steps:

1. Create an indicator variable (dummy variable) called `profit_ind` such that when `oibdp > 0` this variable is 1. Otherwise it is 0.
2. Group by company names
3. Summarize `profit_ind` by taking its sum. Also, get the total number of observations for each company.

```
## # A tibble: 10 x 3
##    conm                          profit_years total_years
##  * <chr>                                <dbl>       <int>
##  1 AMAZON.COM INC                          11          11
##  2 APPLE INC                               11          11
##  3 FACEBOOK INC                            11          11
##  4 INTL BUSINESS MACHINES CORP             11          11
##  5 MICROSOFT CORP                          12          12
##  6 NETFLIX INC                             11          11
##  7 PAYPAL HOLDINGS INC                      8           8
##  8 QUALCOMM INC                            11          11
##  9 TESLA INC                                7          11
## 10 TWITTER INC                              5          10
```

Q7

Find the average annual stock returns of all the companies. Follow these steps:

1. Arrange the data set by `conm` and `datadate`.
2. Group by `conm`
3. Calculate stock return `stk_ret` by taking the difference between `prcc_f` and its lag and then divide the difference by the lag of `prcc_f`
4. Summarize to get the mean of the stock returns `stk_ret_mean`.
5. Display the average stock returns in percentage format.

Hint: This is exactly the same procedure we used to calculate sales growth.

Hint: learn more about percent() function from scales package, which is already installed with tidyverse

```
## # A tibble: 10 x 2
##    conm                          stk_ret_mean
##  * <chr>                         <chr>
##  1 AMAZON.COM INC                39.028%
##  2 APPLE INC                     4.763%
##  3 FACEBOOK INC                  38.682%
##  4 INTL BUSINESS MACHINES CORP -0.234%
##  5 MICROSOFT CORP                25.883%
##  6 NETFLIX INC                   38.770%
```

```
##  7 PAYPAL HOLDINGS INC         50.985%
##  8 QUALCOMM INC                12.919%
##  9 TESLA INC                   56.175%
## 10 TWITTER INC                 5.540%
```

Q8

In many statistical and machine learning applications, we use scaled variables instead of the original variables. A scaled variable is typically created by subtracting the sample mean of the variable from the variable and dividing it by its standard deviation. There is a `scale()` function in base R which can directly do it.

You want to create a scaled variable for `sale` but separately for each company. Therefore, you can't use the mean and standard deviation of `sale` for the entire sample. Instead, you have to calculate these statistics for each company separately and then create a scaled variable. Follow these steps:

1. Group by `conm`
2. Summarize `sale` to get the mean (`sale_mean`) and the standard deviation (`sale_sd`)
3. Assign this data frame to `d3_sum`
4. Join `d3` and `d3_sum` by `conm`
5. Create `sale_scaled` by subtracting `sale_mean` from `sale` and dividing this difference by `sale_sd`

Print the first 10 rows for Twitter with `conm`, `sale`, `sale_scaled`, `sale_mean`, and `sale_sd` using `as.data.frame()`

```
##           conm     sale sale_scaled sale_mean sale_sd
## 1  TWITTER INC  106.313  -1.4474185  1990.013 1301.42
## 2  TWITTER INC  316.933  -1.2855799  1990.013 1301.42
## 3  TWITTER INC  664.890  -1.0182127  1990.013 1301.42
## 4  TWITTER INC 1403.002  -0.4510538  1990.013 1301.42
## 5  TWITTER INC 2218.032   0.1752082  1990.013 1301.42
## 6  TWITTER INC 2529.619   0.4146290  1990.013 1301.42
## 7  TWITTER INC 2443.299   0.3483014  1990.013 1301.42
## 8  TWITTER INC 3042.359   0.8086140  1990.013 1301.42
## 9  TWITTER INC 3459.329   1.1290102  1990.013 1301.42
## 10 TWITTER INC 3716.349   1.3265021  1990.013 1301.42
```