Daniel Schumacher VisCom HW 1

AUTHOR
HDD249 Dan Schumacher

Homework 1

Set up

```
library(tidyverse)
library(scales)
```

Read `tech_stocks_csv.zip` into your R session using `read_csv()` function from `readr` package. Store the resulting object in `d1`.

```
Rows: 93 Columns: 981
── Column specification ────────────────────────────────────────────────────────
Delimiter: ","
chr   (34): gvkey, indfmt, consol, popsrc, datafmt, tic, cusip, conm, acctch...
dbl  (436): fyear, ajex, ajp, currtr, fyr, ismod, ltcm, pddur, scf, src, upd...
lgl  (506): adrr, bspr, curuscn, ogm, stalt, udpl, acco, accrt, acoxar, acql...
date   (5): datadate, apdedate, fdate, pdate, ipodate

ℹ Use `spec()` to retrieve the full column specification for this data.
ℹ Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

Q1

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

```
d1 %>%
  select(at,emp,xrd) %>%
  summarize(
    median_at = median(at),
    median_emp = median(emp),
    median_xrd = median(xrd)
  )
```

```
# A tibble: 1 × 3
  median_at median_emp median_xrd
      <dbl>      <dbl>      <dbl>
1     93798       72.0       7754
```

Q2

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

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

```
d1 %>%
  filter(
    conm %in% c('APPLE INC',
                    'META PLATFORMS INC',
                    'TESLA INC')
    ) %>%
  group_by(conm) %>%
  summarise(
    mean_sale = mean(sale),
    mean_oibdp = mean(oibdp),
    mean_xrd = mean(xrd)
  ) %>%
  as.data.frame()
```

```
                conm mean_sale mean_oibdp  mean_xrd
1          APPLE INC 224763.08  71468.923 11167.000
2 META PLATFORMS INC  43413.00  20110.923  9754.923
3          TESLA INC  18585.48   2672.776  1094.280
```

```
### same thing but using accross
(d1means <- d1 %>%
  filter(
    conm %in% c('APPLE INC',
                    'META PLATFORMS INC',
                    'TESLA INC')
    ) %>%
  group_by(conm) %>%
  summarise(
    across(c(sale,oibdp,xrd), mean)
  ) %>%
  as.data.frame())
```

```
                conm      sale     oibdp       xrd
1          APPLE INC 224763.08 71468.923 11167.000
```

```
2 META PLATFORMS INC  43413.00 20110.923  9754.923
3          TESLA INC  18585.48  2672.776  1094.280
```

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

```
d1means %>%
  mutate(
    across(
      where(is.numeric),
      round, 1)
    ) %>%
  as.data.frame()
```

```
Warning: There was 1 warning in `mutate()`.
ℹ In argument: `across(where(is.numeric), round, 1)`.
Caused by warning:
! The `...` argument of `across()` is deprecated as of dplyr 1.1.0.
Supply arguments directly to `.fns` through an anonymous function instead.

  # Previously
  across(a:b, mean, na.rm = TRUE)

  # Now
  across(a:b, \(x) mean(x, na.rm = TRUE))

                conm     sale   oibdp     xrd
1          APPLE INC 224763.1 71468.9 11167.0
2 META PLATFORMS INC  43413.0 20110.9  9754.9
3          TESLA INC  18585.5  2672.8  1094.3
```

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: `sum(is.na(d1$xad))`

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 `d2`.

```
#how many NAs do we have? ----> 13
sum(is.na(d1$xad))
```

```
[1] 13
```

```
#lets replace them with 0s
d2 <- d1 %>%
    #           ifelse(test,    yes-val, no-val)
    mutate(adv = ifelse(is.na(xad), 0, xad))

# we should have 0 NAs now let's check
sum(is.na(d2$adv))
```

```
[1] 0
```

Q5

Using `d2`, create the following variables and print first 8 rows for NVidia 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`

```
d2 <- d2 %>%
  mutate(
    roa = oibdp/at,
    fcf = oancf/che,
    strat_emph = (adv - xrd) / at
  )


d2 %>%
  select(conm, datadate, roa, fcf, strat_emph) %>%
  filter(conm == 'NVIDIA CORP')
```

```
# A tibble: 14 × 5
   conm        datadate      roa   fcf strat_emph
   <chr>       <date>      <dbl> <dbl>      <dbl>
 1 NVIDIA CORP 2010-01-31 0.0926 0.282     -0.249
 2 NVIDIA CORP 2011-01-31 0.129  0.271     -0.187
 3 NVIDIA CORP 2012-01-31 0.158  0.291     -0.179
 4 NVIDIA CORP 2013-01-31 0.142  0.221     -0.177
 5 NVIDIA CORP 2014-01-31 0.107  0.179     -0.182
 6 NVIDIA CORP 2015-01-31 0.136  0.196     -0.187
 7 NVIDIA CORP 2016-01-31 0.149  0.233     -0.178
 8 NVIDIA CORP 2017-01-31 0.217  0.246     -0.147
 9 NVIDIA CORP 2018-01-31 0.303  0.493     -0.158
10 NVIDIA CORP 2019-01-31 0.309  0.504     -0.177
```

```
11 NVIDIA CORP 2020-01-31 0.187  0.437     -0.163
12 NVIDIA CORP 2021-01-31 0.205  0.504     -0.136
13 NVIDIA CORP 2022-01-31 0.254  0.429     -0.119
14 NVIDIA CORP 2023-01-31 0.174  0.424     -0.178
```

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.

```
d2 <- d2 %>%
  mutate(
    profit_ind = ifelse(oibdp >0, 1, 0)
  )

#let's check it
d2$profit_ind
```

```
 [1] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
[39] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
[77] 1 1 1 1 0 0 0 1 1 0 1 1 1 1 1 1 1
```

2. Group by company names

3. Summarize `profit_ind` by taking its sum. Also, get the total number of observations for each company.

```
d2 %>%
  group_by(conm) %>%
  summarise(
     profit_years  = sum(profit_ind),
     # I found this answer on Stack Overflow.
     # why doesn't total_years = count(profit_ind) work?
     total_years = n(), .groups = 'drop')
```

```
# A tibble: 7 × 3
  conm               profit_years total_years
  <chr>                     <dbl>       <int>
1 ALPHABET INC                 13          13
2 AMAZON.COM INC               13          13
3 APPLE INC                    13          13
4 META PLATFORMS INC           13          13
5 MICROSOFT CORP               14          14
6 NVIDIA CORP                  14          14
7 TESLA INC                     9          13
```

Q7

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

```
d2 %>%
# Arrange the data set by conm and datadate.
  arrange(conm, datadate) %>%

# Group by conm
  group_by(conm) %>%

# Calculate stock return stk_ret by taking the difference between mkvalt (Market value of equity)
  mutate(
    stk_ret =
      (mkvalt - lag(mkvalt, n = 1)) / lag(mkvalt, n = 1)
  ) %>%

# Calculate stock return stk_ret2 by taking the difference between prcc_f (Stock price at the end
  mutate(
    stk_ret2 =
      (prcc_f - lag(prcc_f, n = 1)) / lag(prcc_f, n = 1)
  ) %>%

# Summarize to get the mean of the stock returns stk_ret_mean and stk_ret2_mean.

  summarize(
    stk_ret_mean = percent(mean(stk_ret, na.rm = T), 0.01),
    stk_ret_mean2 =percent(mean(stk_ret2, na.rm = T), 0.01)
  )
```

```
# A tibble: 7 × 3
  conm              stk_ret_mean stk_ret_mean2
  <chr>             <chr>        <chr>
1 ALPHABET INC      19.87%       11.00%
2 AMAZON.COM INC    29.98%       24.60%
3 APPLE INC         24.31%       5.62%
4 META PLATFORMS INC 29.45%      26.84%
5 MICROSOFT CORP    22.61%       24.01%
6 NVIDIA CORP       57.76%       43.89%
7 TESLA INC         115.37%      43.60%
```

```
# Display the average stock returns in percentage format.
```

Not graded: The average stock returns calculated using these two measures are very different. Which of these is correct?

1. My guess is stk_ret because stk_ret2 is yearly whereas stk_ret is on a more regular interval.

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 dataframe to `d2_sum`
4. Join `d2` and `d2_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 Tesla with `conm`, `sale`, `sale_scaled`

```
#steps 1-3
(d2_sum <- d2 %>%
  group_by(conm) %>%
  summarize(
    sale_mean = mean(sale, na.rm = T),
    sale_sd = sd(sale, na.rm = T)
  ))
```

```
# A tibble: 7 × 3
  conm               sale_mean sale_sd
  <chr>                  <dbl>   <dbl>
1 ALPHABET INC         118540.  82029.
2 AMAZON.COM INC       200842. 164507.
3 APPLE INC            224763.  92452.
4 META PLATFORMS INC    43413   42337.
5 MICROSOFT CORP       114056.  48604.
6 NVIDIA CORP            9914.   8203.
7 TESLA INC             18585.  24698.
```

```
# step 4-6
left_join(d2, d2_sum, by = 'conm') %>%

  mutate(
```

```
  sale_scaled = (sale - sale_mean)/sale_sd
) %>%

filter(conm == 'TESLA INC') %>%

select(conm, sale, sale_scaled) %>%

as.data.frame() %>%
head(10)
```

```
        conm      sale sale_scaled
1  TESLA INC   116.744  -0.7477853
2  TESLA INC   204.242  -0.7442426
3  TESLA INC   413.256  -0.7357798
4  TESLA INC  2013.496  -0.6709872
5  TESLA INC  3198.356  -0.6230132
6  TESLA INC  4046.025  -0.5886917
7  TESLA INC  7000.132  -0.4690821
8  TESLA INC 11758.751  -0.2764092
9  TESLA INC 21461.268   0.1164384
10 TESLA INC 24578.000   0.2426325
```