Data Wrangling - 3

COMM 205 - Lecture 21 - R6

Hasan Cavusoglu

2020

Agenda

- `group_by()` with **multiple** columns
  - `group_by()` with `summarise()`
  - `group_by()` with `mutate()`
- `if_else()`
  - used in `mutate()`
  - used in `summarise()`
- `arrange()`function

group_by() with Multiple Variables

- Recall that in the last R class, we used the group_by funtion to repeat computations on subsets of a data frame.
- the syntax for the group_by() function was:

```
group_by(group_by_variable(s)) %>%
  [mutate() or summarise()]
```

- Last time, we use a single variable (i.e., column) to group observations. group_by(), however, has the ability to group a dataset by more than just one variable.

group_by() based on columns with summarise()

Question

Suppose now we want R to create a new data frame with the summary statistics of **assets** (i.e., *at*) based on **industry** (i.e., *naicsh*) and **fiscal year** (i.e., *fyear*).

- So, all firms in the *same* **naicsh** and the *same* **fyear** will be grouped as one group, and R will output the *summary statistics* of **at** for each group.
- Run the following command on R to create the summary data frame:

```
library(tidyverse)
companies <- readRDS("data/North_American_Stock_Market_1994-2013.rds")
at_summary <- companies %>%
  group_by(naicsh, fyear) %>%
  summarise(min_at = min(at, na.rm = TRUE),
            avg_at = mean(at, na.rm = TRUE),
            max_at = max(at, na.rm = TRUE))
```

Result

	naicsh	fyear	min_at	avg_at	max_at
1	21	1994	0.060	25.27150	50.483
2	21	2006	12.681	85.35037	223.181
3	21	*NA*	Inf	NaN	-Inf
4	23	2000	2161.130	2161.13000	2161.130
5	33	1996	140.736	140.73600	140.736
6	42	1994	27.346	781.04850	1534.751
7	42	1995	7.717	838.48000	1669.243
8	42	1996	2119.021	2119.02100	2119.021
9	42	1997	1997.821	1997.82100	1997.821
10	42	1998	2103.902	29289.45100	56475.000
11	42	1999	2564.826	32330.91300	62097.000
12	42	2000	2.593	18714.35000	53680.856
13	42	2001	1.912	17490.41600	50138.090
14	42	2002	2437.448	28916.36100	55395.274
15	42	2003	2624.678	33545.14750	64465.617
16	42	2004	2809.573	36815.09900	70820.625
17	42	2005	3107.921	38043.48250	72979.044
18	42	2006	3046.088	43260.50600	83474.924
19	42	2007	0.623	33383.00733	97054.371
20	42	2008	3515.417	44001.36000	84487.303

NAs in the group_by() variables

- You might be wondering why we had a strange 3rd row in `at_summary` (refer to previous slide). The reason is that there are observations whose `fyear` is `NA` in our original data frame (remember we have also seen that last time).

You can confirm that there are `NA`s in `fyear` by typing:

```
> sum(is.na(companies$fyear)) # # of NAs in fyear
[1] 477
```

- Therefore, it is always a **good** practice to ***remove*** observations with `NA` values in the columns that will be used as **group_by variables**. The code should be then:

```
at_summary <- companies %>%
  filter(!is.na(fyear)) %>%
  group_by(naicsh, fyear) %>%
  summarise(min_at = min(at, na.rm = TRUE), avg_at = mean(at, na.rm = TRUE),
            max_at = max(at, na.rm = TRUE))
```

Since gvkey column has **no** `NA`, `!is.na(gvkey)` is **not** included in the **filter** above.

group_by() based on multiple columns with mutate()

Question

Suppose you are asked to calculate the *sum* of **total assets** (*at*) of all firms by **country of headquarters** (*loc*) and **fiscal year** (*fyear*). Let's call this new column **tot_at**. You are also asked to retain all the observations in the original data.

Since we are doing the calculation based on `loc` and `fyear`, we need to use `group_by` with two variables (`loc` and `fyear`). However, we first need to drop observations that have missing `loc` or `fyear` by using `filter(!is.na(loc), !is.na(fyear))`. Then, we can pipe the screened data to `mutate()` where we sum all `at` in each group created by `group_by`.

The code should accomplish what we are interested in:

```
total_at <- companies %>%
  filter(!is.na(loc), !is.na(fyear)) %>%
  group_by(loc, fyear) %>%
  mutate(total_at = sum(at, na.rm = TRUE))
```

Result

You can confirm the output (by just looking at a few columns)

```
View(total_at %>% select(gvkey, fyear, loc,conm, at, total_at))
```

You can see that all firms located in the same `loc` and the same `fyear` are group together as a subset, and then R returns the total of `at` for each subset.

	gvkey	fyear	loc	conm	at	total_at
1	012071	1994	ANT	SUNRESORTS LTD -CL A	25.959	1389.199
2	024368	1994	ANT	RETAIL HOLDINGS NV	1306.500	1389.199
3	028380	1994	ANT	ORTHOFIX INTERNATIONAL NV	56.740	1389.199
4	119993	1994	ANT	STATIA TERMINALS GROUP NV	*NA*	1389.199

The `gvkey` `119993` in `loc` `ANT` (i.e., *Netherlands Antilles*) in `fyear` `1994` has a missing `at`. In the calculation of `tot_at`, R ignored this missing `at` as `na.rm = TRUE` is set.

if_else()

Similar to IF function in Excel, `if_else()` makes conditional assignment depending in the logical comparison provided.

if_else() Syntax

`if_else(condition, true_v, false_v)`
where **condition** is a logical vector (the comparison goes here!)
true_v, false_v are values to be returned when the condition is `TRUE` or `FALSE` respectively.

- **true_v, false_v** must be either the same length as condition, or length 1. They must also be the same type: `if_else()` checks that they have the same type and same class.
- If the condition is NA, it returns NA by default.

- `if_else()` can be used within `mutate()` and `summarise()` *with* or *without* `group_by()`.

Using if_else() in mutate()

Question

Using **companies** data frame, create a new data frame with **gvkey**, **fyear**, **conm**, **loc**, **ch**, and *two new* columns (**p_ch** and **nrth_loc**).
where

- **p_ch** is **ch** if **ch** is positive; otherwise, **0**.
- **nrth_loc** is **"North America"** if **loc** is **"USA"** or **"CAN"**; otherwise, **"other"**.

Since we need to retain the observations, `mutate()` is used. The entire code to generate the new data frame is:

```
north_companies <- companies %>%
  mutate(p_ch = if_else(ch>0, ch,0),
         nrth_loc = if_else(loc=="CAN"|loc=="USA",
                            "North America",
                            "other")) %>%
  select(gvkey,fyear,conm,loc,ch, p_ch,nrth_loc)
```

Partial Result

Let's look at a particular gvkey == "009652".

```
north_companies %>% filter(gvkey == "009652")
```

	gvkey	fyear	conm	loc	ch	p_ch	nrth_loc
34180	009652	1994	SHELL CANADA LTD -CL A	CAN	256	256	North America
34181	009652	1995	SHELL CANADA LTD -CL A	CAN	444	444	North America
34182	009652	1996	SHELL CANADA LTD -CL A	CAN	1190	1190	North America
34183	009652	1997	SHELL CANADA LTD -CL A	CAN	619	619	North America
34184	009652	1998	SHELL CANADA LTD -CL A	CAN	325	325	North America
34185	009652	1999	SHELL CANADA LTD -CL A	CAN	299	299	North America
34186	009652	2000	SHELL CANADA LTD -CL A	CAN	752	752	North America
34187	009652	2001	SHELL CANADA LTD -CL A	CAN	-25	0	North America
34188	009652	2002	SHELL CANADA LTD -CL A	CAN	0	0	North America
34189	009652	2003	SHELL CANADA LTD -CL A	CAN	0	0	North America
34190	009652	2004	SHELL CANADA LTD -CL A	CAN	127	127	North America
34191	009652	2005	SHELL CANADA LTD -CL A	CAN	1083	1083	North America
34192	009652	2006	SHELL CANADA LTD -CL A	CAN	0	0	North America

Using `if_else()` in `summarise()`

Question

Suppose you are asked to create a data frame that will summarize observations in every 'year' and 'loc' combination (i.e. group). For each group of firms, you record

- (i) total number of employees in all employers in the group ,
- (ii) total number of employees in big employers in the group and
- (iii) proportion of employees working in big employers in the group (that is, the ratio of (ii) over (i)).

Note: A firm employing at least 5,000 employees is considered a big employer.

$$\frac{\text{total number of employees working in firms employing at least 5,000 in a given fyear and loc}}{\text{total number of employees working in all the firms in a given fyear and loc}}$$

First, as a good practice (i.e., not mandatory), let's drop observations with missing `fyear`, or `loc` using the `filter()`. And keep only columns of interest by selecting `gvkey`, `fyear`, `loc` and `emp` *only*.

```
companies %>%
  filter(!is.na(fyear), !is.na(loc)) %>%
  select(gvkey, fyear, loc, emp)
```

Strategy to solve this question

We need to first `group_by` screened data frame according to `fyear` and `loc`, create the columns within `summarise()`

1) **Denominator**: we need to find out the total number of employees in firms for each country in `loc` for each year in `fyear`.

```
emp_yr_loc = sum(emp, na.rm = TRUE)
```

2) **Numerator**: we need to find out the total number of employees in firms with `emp>=5` for each country in `loc` for each year in `fyear`. Note that 5,000 employees is recorded as 5 in the dataset since the variable `emp` is in thousands.

```
emp_year_loc_big = sum(if_else(emp>=5, emp, 0), na.rm = TRUE)
```

3) Finally, we simply divide the result we get from the second calculation above by the result we get from the first calculation.

```
ratio_emp_big_over_emp_yr_loc = emp_year_loc_big/emp_year_loc
```

Code

- Let's put everything together.

```
emp_ratio <- companies %>%
  filter( !is.na(fyear) , !is.na(loc)) %>%
  select(gvkey, fyear, loc, emp) %>%
  group_by(fyear, loc) %>%
  summarise(emp_year_loc = sum(emp, na.rm = TRUE),
            emp_year_loc_big = sum(if_else(emp>=5, emp, 0), na.rm = TRUE),
            ratio_emp_big_over_emp_yr_loc = emp_year_loc_big/emp_year_loc)
```

Result

	fyear	loc	emp_year_loc	emp_year_loc_big	ratio_emp_big_over_emp_yr_loc
1	1994	ANT	14.976	14.200	0.9481838
2	1994	ARG	43.928	37.952	0.8639592
3	1994	AUS	421.347	415.792	0.9868161
4	1994	AUT	0.006	0.000	0.0000000
5	1994	BEL	0.000	0.000	NaN
6	1994	BHS	0.000	0.000	NaN
7	1994	BLZ	0.000	0.000	NaN
8	1994	BMU	53.894	36.814	0.6830816
9	1994	BOL	0.940	0.000	0.0000000
10	1994	BRA	98.779	95.596	0.9677766
11	1994	CAN	1944.792	1599.695	0.8225533
12	1994	CHE	300.551	296.085	0.9851406
13	1994	CHL	42.417	20.700	0.4880119
14	1994	CHN	88.832	76.870	0.8653413
15	1994	COL	0.000	0.000	NaN
16	1994	CRI	0.000	0.000	NaN

Showing 1 to 17 of 1,191 entries

Blue Box: You can see that out of all employees working for firms headquartered in Canada, the proportion of those were working for firms with at least 5,000 employees in 1994 is approximately 82 %.

Red Box: `NaN` is created in the ratio column because the denominator is zero.

`arrange()` Function

- `arrange()` orders the rows of the data frame by variable(s) specified.
  - The default is ascending order of the variable.
  - When multiple columns are specified, rows are ordered based on the first column specified. When there is a tie between some rows, the second column specified will be used to order the rows that were tied. The move to the next column specified continues until the tie is broken.

Syntax

arrange(column name(s)) where
desc(column name) orders the rows in decending order of the column.

Question

Question

Suppose you are asked to arrange observations in our North American Stock Market 1994-2013 Dataset in ascending order of ‘fyear‘ and descending order of ‘ch‘ within each ‘fyear‘ value.

```
ordered <- companies %>%
  arrange(fyear, desc(ch)) %>%
  select(fyear, ch, gvkey, conm)
```

Result

Filter

	fyear	ch	gvkey	conm
1	1994	18212.764	005650	HITACHI LTD
2	1994	17823.301	015627	BANK TOKYO-MITSUBISHI
3	1994	17359.899	007114	PANASONIC CORP
4	1994	16298.743	019661	TOYOTA MOTOR CORP
5	1994	15493.899	019349	SIEMENS AG
6	1994	13578.000	002024	BANKAMERICA CORP-OLD
7	1994	13377.898	013294	NATL WESTMINSTER BANK
8	1994	11442.000	015208	FEDERAL HOME LOAN MORTG CORP
9	1994	10619.000	012826	HANSON PLC
10	1994	10083.699	015889	ANZ-AUSTRALIA & NEW ZEALD BK
11	1994	9669.000	007471	MITSUI & CO LTD
12	1994	9582.000	007647	BANK OF AMERICA CORP
13	1994	8832.000	002968	JPMORGAN CHASE & CO
14	1994	8237.680	014802	NATIONAL AUSTRALIA BK
15	1994	7922.000	006066	INTL BUSINESS MACHINES CORP
16	1994	7337.450	010622	TOSHIBA CORP
17	1994	7197.000	012384	ROYAL DUTCH SHELL PLC
18	1994	6544.000	007908	NIPPON TELEGRAPH & TELEPHONE
19	1994	6470.000	003066	CITICORP
20	1994	6402.315	019043	MITSUBISHI ELECTRIC CORP
21	1994	6220.000	004839	FORD MOTOR CO
22	1994	6070.000	004710	FIRST INTERSTATE BNCP
23	1994	5984.765	004925	FUJIFILM HLDGS CORP
24	1994	5729.480	002721	CANON INC
25	1994	5406.000	012673	BARCLAYS PLC
26	1994	5343.315	009818	SONY CORP
27	1994	5208.000	004764	FLEETBOSTON FINANCIAL CORP
28	1994	5126.977	007652	NEC CORP

Showing 1 to 29 of 232,362 entries

As you can see the data is arranged in the *ascending* order of **fyear** and within each **fyear** value, the data is arranged in a *descending* order of **ch**.

The End

Thanks for watching

See you in next time!