dplyr: summarizing and grouping

Bio724D: Fall 2023

2023-09-16

Data set: Yeast colony morphology

Reference

Granek, J. A., D. Murray, Ö. Kayıkçı, and P. M. Magwene. 2013. The genetic architecture of biofilm formation in a clinical isolate of *Saccharomyces cerevisiae*. Genetics 193(2):587-600.https://doi.org/10.1534/genetics.112.142067

Data availability on Dryad

Dryad link: https://doi.org/10.5061/dryad.mn71g

Brief description

- 70 offspring from a genetic cross used to carry out QTL mapping
- Genotype information at two major QTLs [grouping variable]
- Organismal and molecular phenotypes such as colony complexity score, gene expression, concentration of cyclic AMP [discrete and continuous traits]

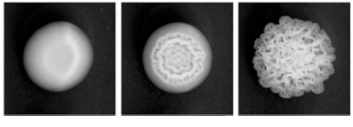

Figure 1: Yeast colonies

Data set: Yeast colony morphology **seg_strain_table.csv**

- See `README_for_seg_strain_table.csv` on Dryad for explanation of columns

Load the data

```
ccm <- read_csv("seg_strain_table.csv")
```

	Strain	Segregant	Pool	Cyr1.geno	Flo11.geno	CM.a	CM.b	CM.c	cAMP	Cyr1.expr	Flo11.expr	Adhes.a	Adhes.b	Adhes.c
1	PMY1278	s44	C	C	S	4.3	3.9	3.9	221.34	1816	226922	0.4055	0.4576	0.4564
2	PMY1330	s67	C	C	C	3.6	3.6	3.5	276.70	*NA*	*NA*	0.4989	0.9868	1.0785
3	PMY1331	s68	C	C	C	3.2	3.2	3.5	122.06	*NA*	*NA*	0.5176	1.2117	1.0973
4	PMY1254	s20	S	S	S	1.0	1.0	1.0	164.80	1418	22337	0.0576	0.3082	0.2970
5	PMY1289	s55	C	S	C	3.1	3.0	3.6	195.59	2159	184194	0.3301	0.4025	0.4269
6	PMY1255	s21	S	S	S	1.0	1.0	1.0	252.33	1555	156586	0.1703	0.1821	0.1847
7	PMY1294	s60	C	C	C	3.3	3.6	3.7	247.58	1864	136993	0.7280	1.5931	1.4825
8	PMY1288	s54	C	C	C	3.6	3.7	3.5	197.00	1140	136704	0.4053	0.6482	0.6894
9	PMY1260	s26	S	S	S	1.0	1.0	1.0	194.97	1444	2113	0.0887	0.1066	0.1055
10	PMY1264	s30	S	S	S	1.0	1.0	1.0	176.29	1265	82581	0.1414	0.2100	0.1957

Figure 2: A sample of rows from `seg_strain_table.csv`

Summarizing using `dplyr::summarize`

- When used on an un-grouped data frame (see below) `dplyr::summarize` applies a function to one or more columns in a data frame, returning a new data frame with a single row
- Data are "collapsed" across rows

```
ccm |>
  summarize(mean_cAMP = mean(cAMP, na.rm=TRUE))
```

mean_cAMP
216.2056

summarize can calculate multiple summaries simultaneously

Example

```
ccm |>
  summarize(mean_cAMP = mean(cAMP, na.rm=TRUE),
            mean_log2_Cyr1 = sd(log2(Cyr1.expr), na.rm=TRUE))
```

mean_cAMP	mean_log2_Cyr1
216.2056	0.4665786

Group assignment: Summarizing

Tukey's Five Number Summary

John Tukey, an important 20th-century statistician and mathematician who is often consider one of the founders of Data Science, recommended that exploratory analyses of data always start with calculation of "five number" summary of key variables:

1. Minimum
2. First quartile (25th percentile)
3. Median (50th percentile)
4. Third quartile (75th percentile)
5. Maximum

Assignment

- Use the `dplyr::summary` to produce a 5-number summary of the `CM.a` variable from the yeast colony morphology data set
- Hint: The function `quantile` may be useful here, among others.

Grouping using `dplyr::group_by`

The function `group_by` "decorates" a data frame with grouping information that you specify

- `group_by` by itself doesn't do any further calculations

Example

```
grouped_ccm <-
  ccm |>
  group_by(Cyr1.geno)
```

Grouping and Summarizing

Once a data frame is grouped, `summarize` then applies it's functions in a group-wise manner

Example

```
grouped_ccm |>
  summarize(nobs = n(),
            mean_cAMP = mean(cAMP, na.rm=TRUE),
            sd_cAMP = sd(cAMP, na.rm=TRUE))
```

Cyr1.geno	nobs	mean_cAMP	sd_cAMP
C	22	233.7377	49.01749
H	2	231.7150	25.20836
S	46	206.7345	46.33473

Grouping: Multiple grouping variables can be used simultaneously

```
ccm |>
  group_by(Pool, Cyr1.geno) |>
  summarize(nobs = n(),
            mean_cAMP = mean(cAMP, na.rm=TRUE),
            sd_cAMP = sd(cAMP, na.rm=TRUE))
```

Pool	Cyr1.geno	nobs	mean_cAMP	sd_cAMP
C	C	22	233.7377	49.01749
C	H	2	231.7150	25.20836
C	S	11	229.4689	54.24839
S	S	35	200.8886	43.02055

Grouping within rows using **dplyr::rowwise**

Sometimes you need to apply a grouping of variables row-wise, such as when you have replicate measures of the same variable and you want to average those replicates. `rowwise` applies grouping on a per-row basis.

Example

```
ccm |>
  rowwise() |>
  mutate(AvgCM = mean(c(CM.a, CM.b, CM.c), na.rm=TRUE)) |>
  ungroup() |>  # remove grouping info so slice_sample does what we want
  slice_sample(n=5) |>
  select(Strain, Segregant, Pool, AvgCM)
```

Strain	Segregant	Pool	AvgCM
PMY1287	s53	C	3.700000
PMY1250	s16	S	1.000000
PMY1272	s38	C	3.033333
PMY1269	s35	C	3.833333
PMY1236	s2	S	1.000000

Group Assignment: Using `rowwise`

Assignment

- What does the call to `slice_sample()` do in the code above?
- Repeat the calculation on the previous slide but leave out the call to `rowwise()` and compare the output. What happens?
- The variables `Adhes.a`, `Adhes.b`, and `Adhes.c` are replicate measures of how well each strain adheres to plastic.
  a. Use `rowwise` and `mutate` to create a new column `AvgAdhes` representing the average `Adhes` of each strain in the mapping population.
  b. Calculate the mean `AvgAdhes` for the data grouped by the `Pool` variable