Merging Data

Contents

1 Merging Data **1**
- 1.1 Peer review data . . . 1
- 1.2 Merging data - merge() . . . 1
- 1.3 Default - merge all common column names . . . 2
- 1.4 Using join in the plyr package . . . 2
- 1.5 If you have multiple data frames . . . 4
- 1.6 More on merging data . . . 4

1 Merging Data

1.1 Peer review data

```
if(!file.exists("data")){dir.create("data")}
```

```
reviews = read.csv("./data/reviews.csv");
solutions <- read.csv("./data/solutions.csv")
head(reviews,2)
```

id	solution_id	reviewer_id	start	stop	time_left	accept
1	3	27	1304095698	1304095758	1754	1
2	4	22	1304095188	1304095206	2306	1

```
head(solutions,2)
```

id	problem_id	subject_id	start	stop	time_left	answer
1	156	29	1304095119	1304095169	2343	B
2	269	25	1304095119	1304095183	2329	C

1.2 Merging data - merge()

- Merges data frames
- Important parameters: *x,y,by,by.x,by.y,all*

```
names(reviews)
```

```
## [1] "id"          "solution_id" "reviewer_id" "start"       "stop"
## [6] "time_left"   "accept"
```

```
names(solutions)
```

```
## [1] "id"         "problem_id" "subject_id" "start"      "stop"
## [6] "time_left"  "answer"
```

```
mergedData <- merge(reviews, solutions, by.x = "solution_id", by.y = "id", all = TRUE)
```

```
head(mergedData)
```

solution_id	id	reviewer_id	start.x	stop.x	time_left	accept	problem_id	subject_id	start.y	stop.y	time_left	answer
1	4	26	1304095267	1304095423	2089	1	156	29	1304095119	1304095169	2343	B
2	6	29	1304095473	1304095513	1999	1	269	25	1304095119	1304095183	2329	C
3	1	27	1304095698	1304095758	1754	1	34	22	1304095127	1304095146	2366	C
4	2	22	1304095188	1304095206	2306	1	19	23	1304095127	1304095150	2362	D
5	3	28	1304095276	1304095320	2192	1	605	26	1304095127	1304095167	2345	A
6	16	22	1304095303	1304095471	2041	1	384	27	1304095133	1304095270	2242	C

1.3 Default - merge all common column names

```
intersect(names(reviews),names(solutions))
```

```
## [1] "id"        "start"     "stop"      "time_left"
```

```
mergedData2 = merge(reviews,solutions,all=TRUE)
head(mergedData2)
```

id	start	stop	time_left	solution_id	reviewer_id	accept	problem_id	subject_id	answer
1	1304095119	1304095169	2343	NA	NA	NA	156	29	B
1	1304095698	1304095758	1754	3	27	1	NA	NA	NA
2	1304095119	1304095183	2329	NA	NA	NA	269	25	C
2	1304095188	1304095206	2306	4	22	1	NA	NA	NA
3	1304095127	1304095146	2366	NA	NA	NA	34	22	C
3	1304095276	1304095320	2192	5	28	1	NA	NA	NA

1.4 Using join in the plyr package

```
library(dplyr)
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
df1 <- data.frame(id=sample(1:10), x=rnorm(10))
df2 <- data.frame(id=sample(1:10), y=rnorm(10))
```

```
head(df1)
```

id	x
5	-0.4090594
3	-0.0456409

id	x
2	-0.9438070
10	0.0780068
4	-0.8461630
1	-2.5301369

```
head(df2)
```

id	y
8	-0.2891753
3	0.0113062
5	0.9543795
10	1.0475310
7	0.3401024
1	0.1796638

Faster, but less full featured - defaults to left join, see help file for more

```
library(plyr)
```

```
## ------------------------------------------------------------------------------
## You have loaded plyr after dplyr - this is likely to cause problems.
## If you need functions from both plyr and dplyr, please load plyr first, then dplyr:
## library(plyr); library(dplyr)
## ------------------------------------------------------------------------------
##
## Attaching package: 'plyr'
## The following objects are masked from 'package:dplyr':
##
##     arrange, count, desc, failwith, id, mutate, rename, summarise,
##     summarize
```

```
arrange(join(df1,df2),id)
```

```
## Joining by: id
```

id	x	y
1	-2.5301369	0.1796638
2	-0.9438070	-0.6001905
3	-0.0456409	0.0113062
4	-0.8461630	0.7753235
5	-0.4090594	0.9543795
6	0.4127289	-0.5890957
7	-0.2243543	0.3401024
8	0.5547680	-0.2891753
9	0.1698095	2.1941504
10	0.0780068	1.0475310

1.5 If you have multiple data frames

```
df1 = data.frame(id=sample(1:10),x=rnorm(10))
df2 = data.frame(id=sample(1:10),y=rnorm(10))
df3 = data.frame(id=sample(1:10),z=rnorm(10))
dfList = list(df1,df2,df3)
join_all(dfList)
```

```
## Joining by: id
## Joining by: id
```

id	x	y	z
2	-1.6190223	-0.0615511	-0.6460507
3	0.5593703	-1.3381703	-0.0802357
1	-0.2141153	-0.8207579	1.7008888
9	1.4634574	-0.6399772	-0.4526983
6	-2.1395973	-1.7274193	-1.3362159
7	-1.0104334	0.4601981	-2.3322159
10	-0.4342609	0.8817071	1.3608391
8	-0.1541254	-1.2398836	-0.6704555
4	-1.6669141	-0.3762667	-0.7230555
5	-1.1115823	1.9885448	1.2973730

1.6 More on merging data

- The quick R data merging page - http://www.statmethods.net/management/merging.html
- plyr information - http://plyr.had.co.nz/
- Types of joins - http://en.wikipedia.org/wiki/Join_(SQL)