Tidy data

“Tidy datasets are all alike but every messy dataset is messy in its own way” — Hadley Wickham

Tidy data

Three rules:

1. Each variable forms a column
2. Each observation forms a row
3. Each type of observational unit forms a table

Example: Contingency table

	survived	died
drug	15	3
placebo	4	12

not tidy

Example: Contingency table

	survived	died
drug	15	3
placebo	4	12

not tidy

tidy

treatment	outcome	count
drug	survived	15
drug	died	3
placebo	survived	4
placebo	died	12

Example: Contingency table

	survived	died
drug	15	3
placebo	4	12

not tidy

tidy

patient	treatment	outcome
1	drug	survived
2	drug	died
3	drug	survived
4	placebo	died
	⋮	

Working with tidy data in R: dplyr

Fundamental actions on data tables:

- select rows — `filter()`
- select columns — `select()`
- make new columns — `mutate()`
- arrange rows — `arrange()`
- calculate summary statistics — `summarize()`
- work on groups of data — `group_by()`

Pipe operator: %>%

Standard R:

```
> mean(iris$Sepal.Length)
[1] 5.843333
```

With pipe:

```
> iris$Sepal.Length %>% mean()
[1] 5.843333
```

Pipe operator: %>%

Standard R:

```
> head(iris)
  Sepal.Length Sepal.Width Petal.Length Petal.Width Species
1          5.1         3.5          1.4         0.2  setosa
2          4.9         3.0          1.4         0.2  setosa
3          4.7         3.2          1.3         0.2  setosa
4          4.6         3.1          1.5         0.2  setosa
5          5.0         3.6          1.4         0.2  setosa
6          5.4         3.9          1.7         0.4  setosa
```

Pipe operator: %>%

With pipe:

```
> iris %>% head()
  Sepal.Length Sepal.Width Petal.Length Petal.Width Species
1          5.1         3.5          1.4         0.2  setosa
2          4.9         3.0          1.4         0.2  setosa
3          4.7         3.2          1.3         0.2  setosa
4          4.6         3.1          1.5         0.2  setosa
5          5.0         3.6          1.4         0.2  setosa
6          5.4         3.9          1.7         0.4  setosa
```

Left and right assignment: <- and ->

Left assignment:

```
> x <- 5
> x
[1] 5
```

Right assignment:

```
> 6 -> x
> x
[1] 6
```

Combining pipe and right assignment

These three lines do all the same thing:

```
> mean.length <- mean(iris$Sepal.Length)
> mean.length <- iris$Sepal.Length %>% mean()
> iris$Sepal.Length %>% mean() -> mean.length
> mean.length
[1] 5.843333
```

dplyr example: count how many herbivores of different orders there are in `msleep`

dplyr example: count how many herbivores of different orders there are in `msleep`

```
> msleep %>% filter(vore=="herbi")
```

dplyr example: count how many herbivores of different orders there are in `msleep`

```
> msleep %>% filter(vore=="herbi") %>% group_by(order)
```

dplyr example: count how many herbivores of different orders there are in `msleep`

```
> msleep %>% filter(vore=="herbi") %>% group_by(order)
%>% summarize(count=n())
```

dplyr example: count how many herbivores of different orders there are in `msleep`

```
> msleep %>% filter(vore=="herbi") %>% group_by(order)
%>% summarize(count=n()) %>% arrange(desc(count))
```

dplyr example: count how many herbivores of different orders there are in `msleep`

```
> msleep %>% filter(vore=="herbi") %>% group_by(order)
%>% summarize(count=n()) %>% arrange(desc(count))
Source: local data frame [9 x 2]

           order count
1       Rodentia    16
2   Artiodactyla     5
3 Perissodactyla     3
4     Hyracoidea     2
5    Proboscidea     2
6  Diprotodontia     1
7     Lagomorpha     1
8         Pilosa     1
9       Primates     1
```