Getting / cleaning data 2

Selecting columns using regular expressions

Tidy select

There are `tidyverse` functions to make selecting variables more straightforwards. You can call these functions as arguments of the `select` function to streamline variable selection. Examples include: `starts_with()`, `ends_with()`, and `contains()`.

Tidy select (helpers)

Here we use `starts_with("t")` to select all variables that begin with `t`.

```r
titanic_train %>%
  select(starts_with("t")) %>%
  slice(1:3)
```

```
##              Ticket
## 1         A/5 21171
## 2          PC 17599
## 3 STON/O2. 3101282
```

Tidy select (helpers)

Here we use `contains("ss")` to select all variables that contain `ss`.

```
titanic_train %>%
  select(contains("ss")) %>%
  slice(1:3)
```

```
##   PassengerId Pclass
## 1           1      3
## 2           2      1
## 3           3      3
```

Tidy select

The are also tidyverse functions that allow us to easily operate on a selection of variables. These functions are called `scoped varients`. You can identity these functions by these `_all`, `_at`, and `_if` suffixes.

Tidy select (*_at)

Here we use `select_at` to select all the variables that contain `ss` in their name:

```
titanic_train %>%
  select_at(vars(contains("ss"))) %>%
  slice(1:3)
```

```
##   PassengerId Pclass
## 1           1      3
## 2           2      1
## 3           3      3
```

Tidy select (*_at)

Here we use `select_at` to select all the variables that contain `ss` in their name and then convert their names to lower case (a handy function to tidy the variable names).

```
titanic_train %>%
  select_at(vars(contains("ss")), .funs = str_to_lower) %>%
  slice(1:3)
```

```
##   passengerid pclass
## 1           1      3
## 2           2      1
## 3           3      3
```

Tidy select (*_if)

Here we use `select_if` to select all the numeric variables in a dataframe:

```
titanic_train %>%
  select_if(is.numeric) %>%
  slice(1:3)
```

```
##   PassengerId Survived Pclass Age SibSp Parch    Fare
## 1           1        0      3  22     1     0  7.2500
## 2           2        1      1  38     1     0 71.2833
## 3           3        1      3  26     0     0  7.9250
```

Tidy select (*_if)

We can also add on a function with `.funs =` to convert their names to lower case (a handy function to tidy the variable names).

```
titanic_train %>%
  select_if(is.numeric, .funs = str_to_lower) %>%
  slice(1:3)
```

```
##   passengerid survived pclass age sibsp parch    fare
## 1           1        0      3  22     1     0  7.2500
## 2           2        1      1  38     1     0 71.2833
## 3           3        1      3  26     0     0  7.9250
```

Tidy select (*_if)

The `select_if` function takes the following form.

```
## Generic code
new_df <- select_if(old_df,
                    .predicate [selects the variable to keep],
                    .funs = [the function to apply to
                             the selected column names])
```