Data Wrangling 1

Georgie Knight

8 August 2016

The task here is to load a data set which is saved as a .csv file and 'clean up' the data in order to make it easier to analyse.

Dplyr

We first load the *dplyr* and *tidyr* packages which will help us wrangle the data:

```
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
```

```
library("tidyr")
```

Loading up the data

Then we load the file 'refine_original.csv' into R and save it as a data frame called 'data_fr':

```
refine_original <- read.csv("C:/Users/Georgie/Dropbox/Springboard/DataWrangling1/refine_original.csv")
data_fr <- data.frame(refine_original)
```

We then convert it to a table called 'my_tbl' within the dplyr package...

```
my_tbl<-dplyr::tbl_df(data_fr)
```

... and take a quick look at it:

```
## # A tibble: 25 x 6
##     company Product.code...number             address   city
##      <fctr>                <fctr>              <fctr> <fctr>
## 1  Phillips                   p-5 Groningensingel 147 arnhem
## 2  phillips                  p-43 Groningensingel 148 arnhem
## 3   philips                   x-3 Groningensingel 149 arnhem
## 4   phllips                  x-34 Groningensingel 150 arnhem
```

```
## 5   phillps                 x-12 Groningensingel 151 arnhem
## 6  phillipS                 p-23 Groningensingel 152 arnhem
## 7      akzo                 v-43   Leeuwardenweg 178 arnhem
## 8      Akzo                 v-12   Leeuwardenweg 179 arnhem
## 9      AKZO                  x-5   Leeuwardenweg 180 arnhem
## 10     akz0                 p-34   Leeuwardenweg 181 arnhem
## # ... with 15 more rows, and 2 more variables: country <fctr>, name <fctr>
```

Task 1: Clean up brand names

Clean up the 'company' column, so all of the misspellings of the brand names are standardized.

We first standardise the column company by using the *mutate* function combined with the *tolower* function

```
my_tbl <- mutate(my_tbl, company = tolower(company))
my_tbl$company
```

```
##  [1] "phillips"   "phillips"   "philips"    "phllips"    "phillps"
##  [6] "phillips"   "akzo"       "akzo"       "akzo"       "akz0"
## [11] "ak zo"      "akzo"       "akzo"       "phillips"   "fillips"
## [16] "phlips"     "van houten" "van houten" "van houten" "van houten"
## [21] "van houten" "unilver"    "unilever"   "unilever"   "unilever"
```

Check the first letters of the company names that are unique as we will use these to rewrite the company column:

```
temp <- my_tbl %>% select(company) %>% mutate(first_letter = substr(company,1,1))
unique(temp$first_letter)
```

```
## [1] "p" "a" "f" "v" "u"
```

We then use a *pipe* to first create a column which contains the first letter of the company name and then use this column along with *replace* to standardise all the names

```
my_tbl <- my_tbl  %>%
  mutate(first_letter = substr(company,1,1))  %>%
  mutate(company = replace(company, first_letter == "p", "philips")) %>%
  mutate(company = replace(company, first_letter == "a", "akzo")) %>%
  mutate(company = replace(company, first_letter == 'f', 'philips')) %>%
  mutate(company = replace(company, first_letter == "v", "van houten")) %>%
  mutate(company = replace(company, first_letter == 'u', 'unilever')) %>%
  select(-first_letter)

my_tbl$company
```

```
##  [1] "philips"    "philips"    "philips"    "philips"    "philips"
##  [6] "philips"    "akzo"       "akzo"       "akzo"       "akzo"
## [11] "akzo"       "akzo"       "akzo"       "philips"    "philips"
## [16] "philips"    "van houten" "van houten" "van houten" "van houten"
## [21] "van houten" "unilever"   "unilever"   "unilever"   "unilever"
```

Task 2: Separate product code and number

Separate the product code and product number into separate columns

We'll use the *separate* function from *tidyr* to do this.

```
my_tbl <-
separate(my_tbl, Product.code...number , c("product_code", "product_number"),
         sep = '-', remove = FALSE)
my_tbl
```

```
## # A tibble: 25 x 8
##    company Product.code...number product_code product_number
## *    <chr>                <fctr>        <chr>          <chr>
## 1  philips                   p-5            p              5
## 2  philips                  p-43            p             43
## 3  philips                   x-3            x              3
## 4  philips                  x-34            x             34
## 5  philips                  x-12            x             12
## 6  philips                  p-23            p             23
## 7     akzo                  v-43            v             43
## 8     akzo                  v-12            v             12
## 9     akzo                   x-5            x              5
## 10    akzo                  p-34            p             34
## # ... with 15 more rows, and 4 more variables: address <fctr>,
## #   city <fctr>, country <fctr>, name <fctr>
```

Task 3: Add product categories

You learn that the product codes actually represent the following product categories: p = Smartphone, v = TV, x = Laptop, q = Tablet. In order to make the data more readable, add a column with the product category for each record.

We again just use the *mutate* function along with *replace* for this.

```
my_tbl <- my_tbl  %>%
mutate(product_category = product_code)  %>%
mutate(
product_category = replace(product_category, product_code == "p", "Smartphone"),
product_category = replace(product_category, product_code == "v", "TV"),
product_category = replace(product_category, product_code == "x", "Laptop"),
product_category = replace(product_category, product_code== "q", "Tablet")
      )

select(my_tbl,product_code, product_category)
```

```
## # A tibble: 25 x 2
##    product_code product_category
##           <chr>            <chr>
## 1             p       Smartphone
## 2             p       Smartphone
## 3             x           Laptop
## 4             x           Laptop
```

```
## 5             x           Laptop
## 6             p       Smartphone
## 7             v               TV
## 8             v               TV
## 9             x           Laptop
## 10            p       Smartphone
## # ... with 15 more rows
```

Task 4: Add address

Concatenate the address, city and country into a full_address column using the *paste* function:

```
my_tbl <-
  mutate(
         my_tbl, full_address = paste(address, city, country, sep= ',')
         )

my_tbl$full_address
```

```
##  [1] "Groningensingel 147,arnhem,the netherlands"
##  [2] "Groningensingel 148,arnhem,the netherlands"
##  [3] "Groningensingel 149,arnhem,the netherlands"
##  [4] "Groningensingel 150,arnhem,the netherlands"
##  [5] "Groningensingel 151,arnhem,the netherlands"
##  [6] "Groningensingel 152,arnhem,the netherlands"
##  [7] "Leeuwardenweg 178,arnhem,the netherlands"
##  [8] "Leeuwardenweg 179,arnhem,the netherlands"
##  [9] "Leeuwardenweg 180,arnhem,the netherlands"
## [10] "Leeuwardenweg 181,arnhem,the netherlands"
## [11] "Leeuwardenweg 182,arnhem,the netherlands"
## [12] "Leeuwardenweg 183,arnhem,the netherlands"
## [13] "Leeuwardenweg 184,arnhem,the netherlands"
## [14] "Delfzijlstraat 54,arnhem,the netherlands"
## [15] "Delfzijlstraat 55,arnhem,the netherlands"
## [16] "Delfzijlstraat 56,arnhem,the netherlands"
## [17] "Delfzijlstraat 57,arnhem,the netherlands"
## [18] "Delfzijlstraat 58,arnhem,the netherlands"
## [19] "Delfzijlstraat 59,arnhem,the netherlands"
## [20] "Delfzijlstraat 60,arnhem,the netherlands"
## [21] "Delfzijlstraat 61,arnhem,the netherlands"
## [22] "Jourestraat 23,arnhem,the netherlands"
## [23] "Jourestraat 24,arnhem,the netherlands"
## [24] "Jourestraat 25,arnhem,the netherlands"
## [25] "Jourestraat 26,arnhem,the netherlands"
```

Taks 5: Create dummy variables for company and product category

For each company and product, create a binary $0, 1$ column which contains 1 when the company/ product is in that row, 0 otherwise. We use a TRUTH assignment converted to integer for this.

```
my_tbl <-mutate( my_tbl,
          company_philips = as.integer(company == 'philips'),
          company_akzo = as.integer(company == 'akzo'),
          company_van_houten = as.integer(company == 'van houten'),
          company_unilever = as.integer(company == 'unilever'),
          product_smartphone = as.integer(product_code == 'p'),
          product_tv = as.integer(product_code == 'v'),
          product_laptop = as.integer(product_code == 'x'),
          product_tablet =  as.integer(product_code == 'q')
          )

select(my_tbl, contains('com'))
```

```
## # A tibble: 25 x 5
##    company company_philips company_akzo company_van_houten
##      <chr>           <int>        <int>               <int>
## 1  philips               1            0                   0
## 2  philips               1            0                   0
## 3  philips               1            0                   0
## 4  philips               1            0                   0
## 5  philips               1            0                   0
## 6  philips               1            0                   0
## 7     akzo               0            1                   0
## 8     akzo               0            1                   0
## 9     akzo               0            1                   0
## 10    akzo               0            1                   0
## # ... with 15 more rows, and 1 more variables: company_unilever <int>
```

```
select(my_tbl, contains('product_'), -product_number)
```

```
## # A tibble: 25 x 6
##    product_code product_category product_smartphone product_tv
##           <chr>            <chr>               <int>      <int>
## 1             p       Smartphone                   1          0
## 2             p       Smartphone                   1          0
## 3             x           Laptop                   0          0
## 4             x           Laptop                   0          0
## 5             x           Laptop                   0          0
## 6             p       Smartphone                   1          0
## 7             v               TV                   0          1
## 8             v               TV                   0          1
## 9             x           Laptop                   0          0
## 10            p       Smartphone                   1          0
## # ... with 15 more rows, and 2 more variables: product_laptop <int>,
## #   product_tablet <int>
```

Now save the cleaned table as a .csv.

```
write.csv(my_tbl, file="refine_clean.csv")
```