Data Wrangling Exercise 01

Tom Thorpe

March 27, 2018

Springboard Data Science Foundation Class

Exercise 01 - Data Wrangling - Refine Data

Objective

Practice using *tidy* and *dplyr* packages to clean up data in a practice dataset.

Exercise Results

Load the tidy and dplyr libraries for exercise.

```
#library(devtools)
library(tidyr)
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
```

Identify input and output files

The practice data set was downloaded into a file called, "refine_original.csv" and the cleaned data will be stored in a file named, "refine_clean.csv".

```
infile = "C:/Users/Tom/git/datasciencefoundation/DataWrangleExer01/refine_original.csv"
outfile = "C:/Users/Tom/git/datasciencefoundation/DataWrangleExer01/refine_clean.csv"
```

Read CSV file into a dataframe

```
refineData <- read.csv(file=infile, header=TRUE, sep=",")
```

Lets see what the data looks like:

```
print(refineData)
```

```
##       company Product.code...number             address   city
## 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
## 11      ak zo                   q-5   Leeuwardenweg 182 arnhem
## 12       akzo                   q-9   Leeuwardenweg 183 arnhem
## 13       akzo                   x-8   Leeuwardenweg 184 arnhem
## 14   phillips                  p-56   Delfzijlstraat 54 arnhem
## 15    fillips                  v-67   Delfzijlstraat 55 arnhem
## 16     phlips                  v-21   Delfzijlstraat 56 arnhem
## 17 Van Houten                  x-45   Delfzijlstraat 57 arnhem
## 18 van Houten                  v-56   Delfzijlstraat 58 arnhem
## 19 van houten                  v-65   Delfzijlstraat 59 arnhem
## 20 van houten                  x-21   Delfzijlstraat 60 arnhem
## 21 Van Houten                  p-23   Delfzijlstraat 61 arnhem
## 22    unilver                   x-3      Jourestraat 23 arnhem
## 23   unilever                   q-4      Jourestraat 24 arnhem
## 24   Unilever                   q-6      Jourestraat 25 arnhem
## 25   unilever                   q-8      Jourestraat 26 arnhem
##            country             name
## 1  the netherlands    dhr p. jansen
## 2  the netherlands    dhr p. hansen
## 3  the netherlands    dhr j. Gansen
## 4  the netherlands    dhr p. mansen
## 5  the netherlands   dhr p. fransen
## 6  the netherlands  dhr p. franssen
## 7  the netherlands    dhr p. bansen
## 8  the netherlands    dhr p. vansen
## 9  the netherlands   dhr p. bransen
## 10 the netherlands   dhr p. janssen
## 11 the netherlands  mevr l.  rokken
## 12 the netherlands  mevr l.  lokken
## 13 the netherlands  mevr l.  mokken
## 14 the netherlands  mevr l.  mokken
## 15 the netherlands  mevr l.  mokken
## 16 the netherlands  mevr l.  mokken
## 17 the netherlands  mevr l.  sokken
## 18 the netherlands  mevr l.  wokken
## 19 the netherlands  mevr l.  kokken
## 20 the netherlands  mevr l.  Bokken
## 21 the netherlands  mevr l.  dokken
## 22 the netherlands  mevr l.  gokken
## 23 the netherlands mevr l.  stokken
## 24 the netherlands  mevr l.  rokken
## 25 the netherlands  mevr l.  rokken
```

Load into a local data frame and check it's contents.

```
products <- tbl_df(refineData)
print(products)
```

```
## # A tibble: 25 x 6
##    company  Product.code...number address             city   country name
##    <fct>    <fct>                 <fct>               <fct>  <fct>   <fct>
##  1 Phillips p-5                   Groningensingel 147 arnhem the ne~ dhr ~
##  2 phillips p-43                  Groningensingel 148 arnhem the ne~ dhr ~
##  3 philips  x-3                   Groningensingel 149 arnhem the ne~ dhr ~
##  4 phllips  x-34                  Groningensingel 150 arnhem the ne~ dhr ~
##  5 phillps  x-12                  Groningensingel 151 arnhem the ne~ dhr ~
##  6 phillipS p-23                  Groningensingel 152 arnhem the ne~ dhr ~
##  7 akzo     v-43                  Leeuwardenweg 178   arnhem the ne~ dhr ~
##  8 Akzo     v-12                  Leeuwardenweg 179   arnhem the ne~ dhr ~
##  9 AKZO     x-5                   Leeuwardenweg 180   arnhem the ne~ dhr ~
## 10 akz0     p-34                  Leeuwardenweg 181   arnhem the ne~ dhr ~
## # ... with 15 more rows
```

Clean Company Name

The first taks to is clean up company names by correcting spellings. The company names have various mis-spelling that most all start with the first letter of the company name. The exception is one entry that starts the spelling of 'philips' with an 'f'. A regular expression is used to select the appropriate records for each of the company names by checking the first character of the company name and change the name to 'adzo', philips', 'unilever', or 'van houten'

```
products$company[grepl("^a",products$company,ignore.case=TRUE)] <- "akzo"
products$company[grepl("^p|^f",products$company,ignore.case=TRUE)] <- "philips"
products$company[grepl("^u",products$company,ignore.case=TRUE)] <- "unilever"
products$company[grepl("^v",products$company,ignore.case=TRUE)] <- "van houten"
```

Check the results:

```
print(products,n=30)
```

```
## # A tibble: 25 x 6
##    company    Product.code...number address      city   country   name
##    <fct>      <fct>                 <fct>        <fct>  <fct>     <fct>
##  1 philips    p-5                   Groningensi~ arnhem the neth~ dhr p. ~
##  2 philips    p-43                  Groningensi~ arnhem the neth~ dhr p. ~
##  3 philips    x-3                   Groningensi~ arnhem the neth~ dhr j. ~
##  4 philips    x-34                  Groningensi~ arnhem the neth~ dhr p. ~
##  5 philips    x-12                  Groningensi~ arnhem the neth~ dhr p. ~
##  6 philips    p-23                  Groningensi~ arnhem the neth~ dhr p. ~
##  7 akzo       v-43                  Leeuwardenw~ arnhem the neth~ dhr p. ~
##  8 akzo       v-12                  Leeuwardenw~ arnhem the neth~ dhr p. ~
##  9 akzo       x-5                   Leeuwardenw~ arnhem the neth~ dhr p. ~
## 10 akzo       p-34                  Leeuwardenw~ arnhem the neth~ dhr p. ~
## 11 akzo       q-5                   Leeuwardenw~ arnhem the neth~ mevr l.~
## 12 akzo       q-9                   Leeuwardenw~ arnhem the neth~ mevr l.~
## 13 akzo       x-8                   Leeuwardenw~ arnhem the neth~ mevr l.~
## 14 philips    p-56                  Delfzijlstr~ arnhem the neth~ mevr l.~
## 15 philips    v-67                  Delfzijlstr~ arnhem the neth~ mevr l.~
## 16 philips    v-21                  Delfzijlstr~ arnhem the neth~ mevr l.~
```

```
## 17 van houten x-45                Delfzijlstr~ arnhem the neth~ mevr l.~
## 18 van houten v-56                Delfzijlstr~ arnhem the neth~ mevr l.~
## 19 van houten v-65                Delfzijlstr~ arnhem the neth~ mevr l.~
## 20 van houten x-21                Delfzijlstr~ arnhem the neth~ mevr l.~
## 21 van houten p-23                Delfzijlstr~ arnhem the neth~ mevr l.~
## 22 unilever   x-3                 Jourestraat~ arnhem the neth~ mevr l.~
## 23 unilever   q-4                 Jourestraat~ arnhem the neth~ mevr l.~
## 24 unilever   q-6                 Jourestraat~ arnhem the neth~ mevr l.~
## 25 unilever   q-8                 Jourestraat~ arnhem the neth~ mevr l.~
```

Separate Product Code Number

Next separate the product code number into *product_code* and *product_number* columns by using the *separate* function to split the data at the dash. The exercise instructions do not indicate if the *Product.code...number* column is to be kept or removed, so it is kept by setting the *remove* option to **FALSE**.

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

Check the results:

```
select(products,contains("product")) %>% print(n=25)
```

```
## # A tibble: 25 x 3
##    Product.code...number product_code product_number
##    <fct>                 <chr>        <chr>
##  1 p-5                   p            5
##  2 p-43                  p            43
##  3 x-3                   x            3
##  4 x-34                  x            34
##  5 x-12                  x            12
##  6 p-23                  p            23
##  7 v-43                  v            43
##  8 v-12                  v            12
##  9 x-5                   x            5
## 10 p-34                  p            34
## 11 q-5                   q            5
## 12 q-9                   q            9
## 13 x-8                   x            8
## 14 p-56                  p            56
## 15 v-67                  v            67
## 16 v-21                  v            21
## 17 x-45                  x            45
## 18 v-56                  v            56
## 19 v-65                  v            65
## 20 x-21                  x            21
## 21 p-23                  p            23
## 22 x-3                   x            3
## 23 q-4                   q            4
## 24 q-6                   q            6
## 25 q-8                   q            8
```

Create *product_category* column

A *product_category* column is created to give a descriptive name to the product code. First the column is created with the mutate function. The value is defaulted to **NA**.

```
products <- mutate(products,product_category=NA)
select(products,contains("product")) %>% print(n=25)
```

```
## # A tibble: 25 x 4
##    Product.code...number product_code product_number product_category
##    <fct>                 <chr>        <chr>          <lgl>
##  1 p-5                   p            5              NA
##  2 p-43                  p            43             NA
##  3 x-3                   x            3              NA
##  4 x-34                  x            34             NA
##  5 x-12                  x            12             NA
##  6 p-23                  p            23             NA
##  7 v-43                  v            43             NA
##  8 v-12                  v            12             NA
##  9 x-5                   x            5              NA
## 10 p-34                  p            34             NA
## 11 q-5                   q            5              NA
## 12 q-9                   q            9              NA
## 13 x-8                   x            8              NA
## 14 p-56                  p            56             NA
## 15 v-67                  v            67             NA
## 16 v-21                  v            21             NA
## 17 x-45                  x            45             NA
## 18 v-56                  v            56             NA
## 19 v-65                  v            65             NA
## 20 x-21                  x            21             NA
## 21 p-23                  p            23             NA
## 22 x-3                   x            3              NA
## 23 q-4                   q            4              NA
## 24 q-6                   q            6              NA
## 25 q-8                   q            8              NA
```

Then the column is populated based on the value of the *product_code* using two different logical tests:

```
products$product_category[grepl("p",products$product_code)] <- "Smartphone"
products$product_category[grepl("q",products$product_code)] <- "Tablet"
products$product_category[products$product_code=="v"] <- "TV"
products$product_category[products$product_code=="x"] <- "Laptop"
```

Check the results:

```
select(products,contains("product")) %>% print(n=25)
```

```
## # A tibble: 25 x 4
##    Product.code...number product_code product_number product_category
##    <fct>                 <chr>        <chr>          <chr>
##  1 p-5                   p            5              Smartphone
##  2 p-43                  p            43             Smartphone
##  3 x-3                   x            3              Laptop
##  4 x-34                  x            34             Laptop
##  5 x-12                  x            12             Laptop
##  6 p-23                  p            23             Smartphone
```

```
##  7 v-43                v           43            TV
##  8 v-12                v           12            TV
##  9 x-5                 x           5             Laptop
## 10 p-34                p           34            Smartphone
## 11 q-5                 q           5             Tablet
## 12 q-9                 q           9             Tablet
## 13 x-8                 x           8             Laptop
## 14 p-56                p           56            Smartphone
## 15 v-67                v           67            TV
## 16 v-21                v           21            TV
## 17 x-45                x           45            Laptop
## 18 v-56                v           56            TV
## 19 v-65                v           65            TV
## 20 x-21                x           21            Laptop
## 21 p-23                p           23            Smartphone
## 22 x-3                 x           3             Laptop
## 23 q-4                 q           4             Tablet
## 24 q-6                 q           6             Tablet
## 25 q-8                 q           8             Tablet
```

Create a full address column

To enable Geo-coding, a *full_address* column is added by combining the *address*, *city* and *state* columns, with each column separated by a comma.

```
products <- mutate(products,full_address=paste(address,",",city,",",country))
```

Check the results:

```
select(products,c("address","city","country","full_address")) %>% print(n=25)
```

```
## # A tibble: 25 x 4
##    address             city   country         full_address
##    <fct>               <fct>  <fct>           <chr>
##  1 Groningensingel 147 arnhem the netherlands Groningensingel 147 , arnhe~
##  2 Groningensingel 148 arnhem the netherlands Groningensingel 148 , arnhe~
##  3 Groningensingel 149 arnhem the netherlands Groningensingel 149 , arnhe~
##  4 Groningensingel 150 arnhem the netherlands Groningensingel 150 , arnhe~
##  5 Groningensingel 151 arnhem the netherlands Groningensingel 151 , arnhe~
##  6 Groningensingel 152 arnhem the netherlands Groningensingel 152 , arnhe~
##  7 Leeuwardenweg 178   arnhem the netherlands Leeuwardenweg 178 , arnhem ~
##  8 Leeuwardenweg 179   arnhem the netherlands Leeuwardenweg 179 , arnhem ~
##  9 Leeuwardenweg 180   arnhem the netherlands Leeuwardenweg 180 , arnhem ~
## 10 Leeuwardenweg 181   arnhem the netherlands Leeuwardenweg 181 , arnhem ~
## 11 Leeuwardenweg 182   arnhem the netherlands Leeuwardenweg 182 , arnhem ~
## 12 Leeuwardenweg 183   arnhem the netherlands Leeuwardenweg 183 , arnhem ~
## 13 Leeuwardenweg 184   arnhem the netherlands Leeuwardenweg 184 , arnhem ~
## 14 Delfzijlstraat 54   arnhem the netherlands Delfzijlstraat 54 , arnhem ~
## 15 Delfzijlstraat 55   arnhem the netherlands Delfzijlstraat 55 , arnhem ~
## 16 Delfzijlstraat 56   arnhem the netherlands Delfzijlstraat 56 , arnhem ~
## 17 Delfzijlstraat 57   arnhem the netherlands Delfzijlstraat 57 , arnhem ~
## 18 Delfzijlstraat 58   arnhem the netherlands Delfzijlstraat 58 , arnhem ~
## 19 Delfzijlstraat 59   arnhem the netherlands Delfzijlstraat 59 , arnhem ~
## 20 Delfzijlstraat 60   arnhem the netherlands Delfzijlstraat 60 , arnhem ~
## 21 Delfzijlstraat 61   arnhem the netherlands Delfzijlstraat 61 , arnhem ~
```

```
## 22 Jourestraat 23      arnhem the netherlands Jourestraat 23 , arnhem , t~
## 23 Jourestraat 24      arnhem the netherlands Jourestraat 24 , arnhem , t~
## 24 Jourestraat 25      arnhem the netherlands Jourestraat 25 , arnhem , t~
## 25 Jourestraat 26      arnhem the netherlands Jourestraat 26 , arnhem , t~
```

The *full_address* is being truncated. Since the *city* and *country* are the same for each row, lets just look at the *address* and *full_address*.

```
select(products,contains("address")) %>% print(n=25)
```

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

Create binary columns for data analysis

New binary category columns are created to allow for easier data analysis. The binary columns have a value of zero unless the data represented by the column is present for that row.

Four columns are created representing each possible company and the value is defaulted to 0.

```
products <- mutate(products,company_philips=0)
products <- mutate(products,company_akzo=0)
products <- mutate(products,company_van_houten=0)
products <- mutate(products,company_unilever=0)
```

Then the the company category column values are set to 1 if the corresponding company name is present.

```
products$company_akzo[products$company=="akzo"] <- 1
products$company_philips[products$company=="philips"] <- 1
```

```
products$company_van_houten[products$company=="van houten"] <- 1
products$company_unilever[products$company=="unilever"] <- 1
```

Check the results.

```
select(products,contains("company")) %>% print(n=25)
```

```
## # A tibble: 25 x 5
##    company  company_philips company_akzo company_van_hou~ company_unilever
##    <fct>              <dbl>        <dbl>            <dbl>             <dbl>
##  1 philips               1.           0.               0.                0.
##  2 philips               1.           0.               0.                0.
##  3 philips               1.           0.               0.                0.
##  4 philips               1.           0.               0.                0.
##  5 philips               1.           0.               0.                0.
##  6 philips               1.           0.               0.                0.
##  7 akzo                  0.           1.               0.                0.
##  8 akzo                  0.           1.               0.                0.
##  9 akzo                  0.           1.               0.                0.
## 10 akzo                  0.           1.               0.                0.
## 11 akzo                  0.           1.               0.                0.
## 12 akzo                  0.           1.               0.                0.
## 13 akzo                  0.           1.               0.                0.
## 14 philips               1.           0.               0.                0.
## 15 philips               1.           0.               0.                0.
## 16 philips               1.           0.               0.                0.
## 17 van hou~              0.           0.               1.                0.
## 18 van hou~              0.           0.               1.                0.
## 19 van hou~              0.           0.               1.                0.
## 20 van hou~              0.           0.               1.                0.
## 21 van hou~              0.           0.               1.                0.
## 22 unilever              0.           0.               0.                1.
## 23 unilever              0.           0.               0.                1.
## 24 unilever              0.           0.               0.                1.
## 25 unilever              0.           0.               0.                1.
```

Next create binary data columns for the product code. I was wondering if there was a way to create the column and set the value all on the same line and tried a few ways. I first tried the following and it produced an error:

```
products <- mutate(products,product_tablet2=(if(products$product_category=="Tablet"){1}else{0}))
```

```
## Warning in if (products$product_category == "Tablet") {: the condition has
## length > 1 and only the first element will be used
```

Remove the product_tablet2 column.

```
products <- select(products,-product_tablet2)
```

Next I tried the *ifelse* construct and found it worked and was able to create the column and set the value in one statement.

```
products <- mutate(products,product_smartphone=ifelse(product_category=="Smartphone",1,0))
products <- mutate(products,product_tv=ifelse(product_category=="TV",1,0))
products <- mutate(products,product_laptop=ifelse(product_category=="Laptop",1,0))
products <- mutate(products,product_tablet=ifelse(product_category=="Tablet",1,0))
```

Check the results using different selects.

```r
select(products,contains("product")) %>% print(n=25)
```

```
## # A tibble: 25 x 8
##    Product.code...number product_code product_number product_category
##    <fct>                 <chr>        <chr>          <chr>
##  1 p-5                   p            5              Smartphone
##  2 p-43                  p            43             Smartphone
##  3 x-3                   x            3              Laptop
##  4 x-34                  x            34             Laptop
##  5 x-12                  x            12             Laptop
##  6 p-23                  p            23             Smartphone
##  7 v-43                  v            43             TV
##  8 v-12                  v            12             TV
##  9 x-5                   x            5              Laptop
## 10 p-34                  p            34             Smartphone
## 11 q-5                   q            5              Tablet
## 12 q-9                   q            9              Tablet
## 13 x-8                   x            8              Laptop
## 14 p-56                  p            56             Smartphone
## 15 v-67                  v            67             TV
## 16 v-21                  v            21             TV
## 17 x-45                  x            45             Laptop
## 18 v-56                  v            56             TV
## 19 v-65                  v            65             TV
## 20 x-21                  x            21             Laptop
## 21 p-23                  p            23             Smartphone
## 22 x-3                   x            3              Laptop
## 23 q-4                   q            4              Tablet
## 24 q-6                   q            6              Tablet
## 25 q-8                   q            8              Tablet
## # ... with 4 more variables: product_smartphone <dbl>, product_tv <dbl>,
## #   product_laptop <dbl>, product_tablet <dbl>
```

```r
# The below was working
select(products,9,15:18) %>% print(n=25)
```

```
## # A tibble: 25 x 5
##    product_category product_smartphone product_tv product_laptop
##    <chr>                         <dbl>      <dbl>          <dbl>
##  1 Smartphone                       1.         0.             0.
##  2 Smartphone                       1.         0.             0.
##  3 Laptop                           0.         0.             1.
##  4 Laptop                           0.         0.             1.
##  5 Laptop                           0.         0.             1.
##  6 Smartphone                       1.         0.             0.
##  7 TV                               0.         1.             0.
##  8 TV                               0.         1.             0.
##  9 Laptop                           0.         0.             1.
## 10 Smartphone                       1.         0.             0.
## 11 Tablet                           0.         0.             0.
## 12 Tablet                           0.         0.             0.
## 13 Laptop                           0.         0.             1.
## 14 Smartphone                       1.         0.             0.
## 15 TV                               0.         1.             0.
```

```
## 16 TV                                0.         1.             0.
## 17 Laptop                            0.         0.             1.
## 18 TV                                0.         1.             0.
## 19 TV                                0.         1.             0.
## 20 Laptop                            0.         0.             1.
## 21 Smartphone                        1.         0.             0.
## 22 Laptop                            0.         0.             1.
## 23 Tablet                            0.         0.             0.
## 24 Tablet                            0.         0.             0.
## 25 Tablet                            0.         0.             0.
## # ... with 1 more variable: product_tablet <dbl>
```

```
# but now getting error:
#   Error in combine_vars(vars, ind_list) : Position must be between 0 and n
# It's working again. I had an error in the mutate functions
# and was not actually creating the new columns, so the 15:18 range was invalid.
select(products,contains("product"),
       -Product.code...number,
       -product_code,
       -product_number) %>%
  print(n=25)
```

```
## # A tibble: 25 x 5
##    product_category product_smartphone product_tv product_laptop
##    <chr>                         <dbl>      <dbl>          <dbl>
##  1 Smartphone                        1.         0.             0.
##  2 Smartphone                        1.         0.             0.
##  3 Laptop                            0.         0.             1.
##  4 Laptop                            0.         0.             1.
##  5 Laptop                            0.         0.             1.
##  6 Smartphone                        1.         0.             0.
##  7 TV                                0.         1.             0.
##  8 TV                                0.         1.             0.
##  9 Laptop                            0.         0.             1.
## 10 Smartphone                        1.         0.             0.
## 11 Tablet                            0.         0.             0.
## 12 Tablet                            0.         0.             0.
## 13 Laptop                            0.         0.             1.
## 14 Smartphone                        1.         0.             0.
## 15 TV                                0.         1.             0.
## 16 TV                                0.         1.             0.
## 17 Laptop                            0.         0.             1.
## 18 TV                                0.         1.             0.
## 19 TV                                0.         1.             0.
## 20 Laptop                            0.         0.             1.
## 21 Smartphone                        1.         0.             0.
## 22 Laptop                            0.         0.             1.
## 23 Tablet                            0.         0.             0.
## 24 Tablet                            0.         0.             0.
## 25 Tablet                            0.         0.             0.
## # ... with 1 more variable: product_tablet <dbl>
```

```
select(products,"product_category","product_tablet") %>% print(n=25)
```

```
## # A tibble: 25 x 2
```

```
##    product_category product_tablet
##    <chr>                     <dbl>
##  1 Smartphone                   0.
##  2 Smartphone                   0.
##  3 Laptop                       0.
##  4 Laptop                       0.
##  5 Laptop                       0.
##  6 Smartphone                   0.
##  7 TV                           0.
##  8 TV                           0.
##  9 Laptop                       0.
## 10 Smartphone                   0.
## 11 Tablet                       1.
## 12 Tablet                       1.
## 13 Laptop                       0.
## 14 Smartphone                   0.
## 15 TV                           0.
## 16 TV                           0.
## 17 Laptop                       0.
## 18 TV                           0.
## 19 TV                           0.
## 20 Laptop                       0.
## 21 Smartphone                   0.
## 22 Laptop                       0.
## 23 Tablet                       1.
## 24 Tablet                       1.
## 25 Tablet                       1.
```

```
#View(products)
```

Lets look at the final structure for fun.

```
str(products)
```

```
## Classes 'tbl_df', 'tbl' and 'data.frame':    25 obs. of  18 variables:
##  $ company              : Factor w/ 19 levels "ak zo","akz0",..: 7 7 7 7 7 7 3 3 3 3 ...
##  $ Product.code...number: Factor w/ 23 levels "p-23","p-34",..: 4 3 19 20 17 1 13 11 22 2 ...
##  $ product_code         : chr  "p" "p" "x" "x" ...
##  $ product_number       : chr  "5" "43" "3" "34" ...
##  $ address              : Factor w/ 25 levels "Delfzijlstraat 54",..: 9 10 11 12 13 14 19 20 21 22 .
##  $ city                 : Factor w/ 1 level "arnhem": 1 1 1 1 1 1 1 1 1 1 ...
##  $ country              : Factor w/ 1 level "the netherlands": 1 1 1 1 1 1 1 1 1 1 ...
##  $ name                 : Factor w/ 20 levels "dhr j. Gansen",..: 7 6 1 9 4 5 2 10 3 8 ...
##  $ product_category     : chr  "Smartphone" "Smartphone" "Laptop" "Laptop" ...
##  $ full_address         : chr  "Groningensingel 147 , arnhem , the netherlands" "Groningensingel 148
##  $ company_philips      : num  1 1 1 1 1 1 0 0 0 0 ...
##  $ company_akzo         : num  0 0 0 0 0 0 1 1 1 1 ...
##  $ company_van_houten   : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ company_unilever     : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ product_smartphone   : num  1 1 0 0 0 1 0 0 0 1 ...
##  $ product_tv           : num  0 0 0 0 0 0 1 1 0 0 ...
##  $ product_laptop       : num  0 0 1 1 1 0 0 0 1 0 ...
##  $ product_tablet       : num  0 0 0 0 0 0 0 0 0 0 ...
```

Save the cleaned data in a CSV file.

```
write.csv(products, file=outfile,row.names=FALSE)
```

That concludes the exercise.