Avocado Data Wrangling

Anurag Karmarkar

08/08/2020

Setup

```
#install.packages("dplyr") #To use %>% and rename functions
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
```

The pre installed function read.csv was used to read the csv file

Data Description

This dataset consists of the details avocado sales in the United States of America. It was taken from the Hass Avacado Board. It contains the following columns.

1. Date - The date of the observation
2. Average Price - the average price of a single avocado
3. Type - conventional or organic
4. Year - the year
5. Region - the city or region of the observation
6. Total Volume - Total number of avocados sold
7. 4046 - Total number of avocados with PLU (Product Lookup code) 4046 sold
8. 4225 - Total number of avocados with PLU (Product Lookup code) 4225 sold
9. 4770 - Total number of avocados with PLU (Product Lookup code) 4770 sold
10. Small Bags - Totla Number of Small Bags
11. Medium Bags - Total Number of Medium Bags
12. Large Bags - Total Number of Large Bags
13. XLarge Bags - Total Number of Extra Large Bags

This dataset has the column of Type as a categorial column and Average price is numeric.

URL:- https://www.kaggle.com/neuromusic/avocado-prices (https://www.kaggle.com/neuromusic/avocado-prices)

Import Data

The data was imported from the location path mentioned in the code by using read.csv function.

```
avocado <- read.csv(file = 'C:/Users/smart/OneDrive/Documents/Master of Data Science (RMIT)/Seme
ster 2/Data Wrangling/Assignments/Assignment 1/avocado.csv')
head(avocado)
```

```
##   X       Date AveragePrice Total.Volume   X4046     X4225  X4770 Total.Bags
## 1 0 2015-12-27         1.33     64236.62 1036.74  54454.85  48.16    8696.87
## 2 1 2015-12-20         1.35     54876.98  674.28  44638.81  58.33    9505.56
## 3 2 2015-12-13         0.93    118220.22  794.70 109149.67 130.50    8145.35
## 4 3 2015-12-06         1.08     78992.15 1132.00  71976.41  72.58    5811.16
## 5 4 2015-11-29         1.28     51039.60  941.48  43838.39  75.78    6183.95
## 6 5 2015-11-22         1.26     55979.78 1184.27  48067.99  43.61    6683.91
##   Small.Bags Large.Bags XLarge.Bags         type year region
## 1    8603.62      93.25           0 conventional 2015 Albany
## 2    9408.07      97.49           0 conventional 2015 Albany
## 3    8042.21     103.14           0 conventional 2015 Albany
## 4    5677.40     133.76           0 conventional 2015 Albany
## 5    5986.26     197.69           0 conventional 2015 Albany
## 6    6556.47     127.44           0 conventional 2015 Albany
```

Inspect and Understand

Checking the dimentions of the dataframe using dim() function The dataset has 18249 rows and 14 columns

```
dim(avocado)
```

```
## [1] 18249    14
```

Checking the data types of all the variables.

```
class(avocado$X)
```

```
## [1] "integer"
```

```
class(avocado$Date)
```

```
## [1] "factor"
```

```
class(avocado$AveragePrice)
```

```
## [1] "numeric"
```

```
class(avocado$Total.Volume)
```

```
## [1] "numeric"
```

```
class(avocado$X4046)
```

```
## [1] "numeric"
```

```
class(avocado$X4225)
```

```
## [1] "numeric"
```

```
class(avocado$X4770)
```

```
## [1] "numeric"
```

```
class(avocado$Total.Bags)
```

```
## [1] "numeric"
```

```
class(avocado$Small.Bags)
```

```
## [1] "numeric"
```

```
class(avocado$Large.Bags)
```

```
## [1] "numeric"
```

```
class(avocado$XLarge.Bags)
```

```
## [1] "numeric"
```

```
class(avocado$type)
```

```
## [1] "factor"
```

```
class(avocado$year)
```

```
## [1] "integer"
```

```
class(avocado$region)
```

```
## [1] "factor"
```

```
typeof(avocado$X)
```

```
## [1] "integer"
```

```
typeof(avocado$Date)
```

```
## [1] "integer"
```

```
typeof(avocado$AveragePrice)
```

```
## [1] "double"
```

```
typeof(avocado$Total.Volume)
```

```
## [1] "double"
```

```
typeof(avocado$X4046)
```

```
## [1] "double"
```

```
typeof(avocado$X4225)
```

```
## [1] "double"
```

```
typeof(avocado$X4770)
```

```
## [1] "double"
```

```
typeof(avocado$Total.Bags)
```

```
## [1] "double"
```

```
typeof(avocado$Small.Bags)
```

```
## [1] "double"
```

```
typeof(avocado$Large.Bags)
```

```
## [1] "double"
```

```
typeof(avocado$XLarge.Bags)
```

```
## [1] "double"
```

```
typeof(avocado$type)
```

```
## [1] "integer"
```

```
typeof(avocado$year)
```

```
## [1] "integer"
```

```
typeof(avocado$region)
```

```
## [1] "integer"
```

It can be observed that the columns Date, type and region do not have the correct datatypes. All other columns are in their appropriate data types.

```
avocado %>% mutate(type = as.character(avocado$type),
                   region = as.character(avocado$region),
                   Date = as.Date(avocado$Date)) %>% head()
```

```
##   X       Date AveragePrice Total.Volume   X4046     X4225  X4770 Total.Bags
## 1 0 2015-12-27         1.33     64236.62 1036.74  54454.85  48.16    8696.87
## 2 1 2015-12-20         1.35     54876.98  674.28  44638.81  58.33    9505.56
## 3 2 2015-12-13         0.93    118220.22  794.70 109149.67 130.50    8145.35
## 4 3 2015-12-06         1.08     78992.15 1132.00  71976.41  72.58    5811.16
## 5 4 2015-11-29         1.28     51039.60  941.48  43838.39  75.78    6183.95
## 6 5 2015-11-22         1.26     55979.78 1184.27  48067.99  43.61    6683.91
##   Small.Bags Large.Bags XLarge.Bags         type year region
## 1    8603.62      93.25           0 conventional 2015 Albany
## 2    9408.07      97.49           0 conventional 2015 Albany
## 3    8042.21     103.14           0 conventional 2015 Albany
## 4    5677.40     133.76           0 conventional 2015 Albany
## 5    5986.26     197.69           0 conventional 2015 Albany
## 6    6556.47     127.44           0 conventional 2015 Albany
```

The datatypes have been corrected by using the mutate function from the dplyr library.

The rows can be arranged by ordering the “type” column by setting the level of “organic” less than “conventional”. This can be achieved by using factor() and arrange() functions.

```
avocado$type <-factor(avocado$type, levels = c("organic","conventional"),ordered = TRUE)
avocado <- arrange(avocado,type)
head(avocado)
```

```
##   X       Date AveragePrice Total.Volume X4046  X4225 X4770 Total.Bags
## 1 0 2015-12-27         1.83       989.55  8.16  88.59     0     892.80
## 2 1 2015-12-20         1.89      1163.03 30.24 172.14     0     960.65
## 3 2 2015-12-13         1.85       995.96 10.44 178.70     0     806.82
## 4 3 2015-12-06         1.84      1158.42 90.29 104.18     0     963.95
## 5 4 2015-11-29         1.94       831.69  0.00  94.73     0     736.96
## 6 5 2015-11-22         1.94       858.83 13.84  84.18     0     760.81
##   Small.Bags Large.Bags XLarge.Bags    type year region
## 1     892.80       0.00           0 organic 2015 Albany
## 2     960.65       0.00           0 organic 2015 Albany
## 3     806.82       0.00           0 organic 2015 Albany
## 4     948.52      15.43           0 organic 2015 Albany
## 5     736.96       0.00           0 organic 2015 Albany
## 6     755.69       5.12           0 organic 2015 Albany
```

The column names of the data frame are as follows

```
colnames(avocado)
```

```
##  [1] "X"            "Date"         "AveragePrice" "Total.Volume" "X4046"
##  [6] "X4225"        "X4770"        "Total.Bags"   "Small.Bags"   "Large.Bags"
## [11] "XLarge.Bags"  "type"         "year"         "region"
```

Some column names like “X4046” have to be changed to PLU4046 for clarity. This can be done using the rename() function from the dplyr library.

```
avocado <- avocado %>% rename(SrNo = X,
                              Total_Volume = Total.Volume,
                              PLU4046 = X4046,
                              PLU4225 = X4225,
                              PLU4770 = X4770,
                              Total_Bags = Total.Bags,
                              Small_Bags = Small.Bags,
                              Large_Bags = Large.Bags,
                              Extra_Large_Bags = XLarge.Bags)
head(avocado)
```

```
##   SrNo       Date AveragePrice Total_Volume PLU4046 PLU4225 PLU4770 Total_Bags
## 1    0 2015-12-27         1.83       989.55    8.16   88.59       0     892.80
## 2    1 2015-12-20         1.89      1163.03   30.24  172.14       0     960.65
## 3    2 2015-12-13         1.85       995.96   10.44  178.70       0     806.82
## 4    3 2015-12-06         1.84      1158.42   90.29  104.18       0     963.95
## 5    4 2015-11-29         1.94       831.69    0.00   94.73       0     736.96
## 6    5 2015-11-22         1.94       858.83   13.84   84.18       0     760.81
##   Small_Bags Large_Bags Extra_Large_Bags    type year region
## 1     892.80       0.00                0 organic 2015 Albany
## 2     960.65       0.00                0 organic 2015 Albany
## 3     806.82       0.00                0 organic 2015 Albany
## 4     948.52      15.43                0 organic 2015 Albany
## 5     736.96       0.00                0 organic 2015 Albany
## 6     755.69       5.12                0 organic 2015 Albany
```

Subsetting

The subset function was used to generate a sub dataset consisting of the first 10 rows.

```
avocado_subset <- subset(avocado, SrNo <= 10)
head(avocado_subset)
```

```
##   SrNo       Date AveragePrice Total_Volume PLU4046 PLU4225 PLU4770 Total_Bags
## 1    0 2015-12-27         1.83       989.55    8.16   88.59       0     892.80
## 2    1 2015-12-20         1.89      1163.03   30.24  172.14       0     960.65
## 3    2 2015-12-13         1.85       995.96   10.44  178.70       0     806.82
## 4    3 2015-12-06         1.84      1158.42   90.29  104.18       0     963.95
## 5    4 2015-11-29         1.94       831.69    0.00   94.73       0     736.96
## 6    5 2015-11-22         1.94       858.83   13.84   84.18       0     760.81
##   Small_Bags Large_Bags Extra_Large_Bags    type year region
## 1     892.80       0.00                0 organic 2015 Albany
## 2     960.65       0.00                0 organic 2015 Albany
## 3     806.82       0.00                0 organic 2015 Albany
## 4     948.52      15.43                0 organic 2015 Albany
## 5     736.96       0.00                0 organic 2015 Albany
## 6     755.69       5.12                0 organic 2015 Albany
```

The subseted data was converted into a matrix. The matrix has some numeric and character values. This is due to the prsesnce of character values and numeric values in the subsetted dataframe.

```
head(as.matrix(avocado_subset))
```

```
##   SrNo Date         AveragePrice Total_Volume  PLU4046       PLU4225
## 1 " 0" "2015-12-27" "1.83"       "     989.55" "       8.16" "      88.59"
## 2 " 1" "2015-12-20" "1.89"       "    1163.03" "      30.24" "     172.14"
## 3 " 2" "2015-12-13" "1.85"       "     995.96" "      10.44" "     178.70"
## 4 " 3" "2015-12-06" "1.84"       "    1158.42" "      90.29" "     104.18"
## 5 " 4" "2015-11-29" "1.94"       "     831.69" "       0.00" "      94.73"
## 6 " 5" "2015-11-22" "1.94"       "     858.83" "      13.84" "      84.18"
##   PLU4770      Total_Bags    Small_Bags    Large_Bags   Extra_Large_Bags
## 1 "      0.00" "     892.80" "     892.80" "      0.00" "     0.00"
## 2 "      0.00" "     960.65" "     960.65" "      0.00" "     0.00"
## 3 "      0.00" "     806.82" "     806.82" "      0.00" "     0.00"
## 4 "      0.00" "     963.95" "     948.52" "     15.43" "     0.00"
## 5 "      0.00" "     736.96" "     736.96" "      0.00" "     0.00"
## 6 "      0.00" "     760.81" "     755.69" "      5.12" "     0.00"
##   type      year   region
## 1 "organic" "2015" "Albany"
## 2 "organic" "2015" "Albany"
## 3 "organic" "2015" "Albany"
## 4 "organic" "2015" "Albany"
## 5 "organic" "2015" "Albany"
## 6 "organic" "2015" "Albany"
```

Create a New Dataframe

A new dataframe can be created as follows. It has an ordinal column and an integer column.

```
newdf <- data.frame (ordinal_column = c ("red", "blue", "yellow","blue", "yellow", "yellow","blu
e","red", "blue","red"), int_column = c("3e","t7","g2","a6","d1","f6","v3","d8","a4","c6"))
newdf
```

```
##    ordinal_column int_column
## 1             red         3e
## 2            blue         t7
## 3          yellow         g2
## 4            blue         a6
## 5          yellow         d1
## 6          yellow         f6
## 7            blue         v3
## 8             red         d8
## 9            blue         a4
## 10            red         c6
```

The structures of both the columns before arranging the ordinal column in order are as follows.

```
str(newdf)
```

```
## 'data.frame':    10 obs. of  2 variables:
##  $ ordinal_column: Factor w/ 3 levels "blue","red","yellow": 2 1 3 1 3 3 1 2 1 2
##  $ int_column    : Factor w/ 10 levels "3e","a4","a6",..: 1 9 8 3 5 7 10 6 2 4
```

Levels of the ordinal column are as follows.

```
newdf$ordinal_column <-factor(newdf$ordinal_column, levels = c("red","blue","yellow"),ordered =
TRUE)
newdf$ordinal_column
```

```
##  [1] red    blue   yellow blue   yellow yellow blue   red    blue   red
## Levels: red < blue < yellow
```

The ordinal column can be arranged and storred as follows.

```
newdf <- arrange(newdf,ordinal_column)
newdf
```

```
##    ordinal_column int_column
## 1             red         3e
## 2             red         d8
## 3             red         c6
## 4            blue         t7
## 5            blue         a6
## 6            blue         v3
## 7            blue         a4
## 8          yellow         g2
## 9          yellow         d1
## 10         yellow         f6
```

The structures of both the columns after arranging the ordinal column in order are as follows.

```
str(newdf)
```

```
## 'data.frame':    10 obs. of  2 variables:
##  $ ordinal_column: Ord.factor w/ 3 levels "red"<"blue"<"yellow": 1 1 1 2 2 2 2 3 3 3
##  $ int_column    : Factor w/ 10 levels "3e","a4","a6",..: 1 6 4 9 3 10 2 8 5 7
```

Creating a numeric vector.

```
numeric_vector <- c(4,6,2,6,1,8,0,4,6,2)
numeric_vector
```

```
##  [1] 4 6 2 6 1 8 0 4 6 2
```

Using cbind to add the vector as column to the existing dataframe

```
newdf <- cbind(newdf, numeric_vector)
newdf
```

```
##    ordinal_column int_column numeric_vector
## 1             red         3e              4
## 2             red         d8              6
## 3             red         c6              2
## 4            blue         t7              6
## 5            blue         a6              1
## 6            blue         v3              8
## 7            blue         a4              0
## 8          yellow         g2              4
## 9          yellow         d1              6
## 10         yellow         f6              2
```

References:-

1. Kaggle.com. 2020. Avocado Prices. [online] Available at: https://www.kaggle.com/neuromusic/avocado-prices (https://www.kaggle.com/neuromusic/avocado-prices) [Accessed 10 August 2020].
2. Medium. 2020. Renaming Columns With Dplyr In R. [online] Available at: https://medium.com/@HollyEmblem/renaming-columns-with-dplyr-in-r-55b42222cbdc (https://medium.com/@HollyEmblem/renaming-columns-with-dplyr-in-r-55b42222cbdc) [Accessed 10 August 2020].