Data in R.

Katlyn H. Degamo

2023-05-09

```
#To store something in R session, you will assign it a name using <- operator.
#You can assign your previous calculation to an object called my_sum.
#Simple calculation like this.

3 + 3
```

```
## [1] 6
```

```
my_sum <- 4 + 4
my_sum
```

```
## [1] 8
```

```
my_sum + 5
```

```
## [1] 13
```

```
#To display the value, the name of the object must be typed, the *print()* command or
#used the command should be wrapped in parenthesis.

my_sum
```

```
## [1] 8
```

```
print(my_sum)
```

```
## [1] 8
```

```
(another_sum <- 7 + 8)
```

```
## [1] 15
```

```
(new_sum <- my_sum + 5)
```

```
## [1] 13
```

Data Types

```
#The typeof function can be used to see the type of a single scalar value.
#integer can be signified by adding an 'L' to the end.
#Numeric data can be integer form or double form.

my_integer <- 2L
typeof(my_integer)
```

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

```r
my_double <- 5.94
typeof(my_double)
```

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

```r
 #Character data is text data sorrounded by single or doule qoutes.
my_character <- "HI I'M THE PROBLEM"
typeof(my_character)
```

```
## [1] "character"
```

```r
 #Logical data takes the form TRUE or FALSE.

my_logical <- FALSE
typeof(my_logical)
```

```
## [1] "logical"
```

Homogeneous Data Structures

```r
 #Vectors are one-dimensional structures containing data of the same type,
 #notated by using c().
 #Type of vector can also be viewed using the typeof function,
 #but str function can be used to display both contents and its type.

double_vector <- c(1.2, 2.3, 3.4, 45, 56, 7)
typeof(double_vector)
```

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

```r
str(double_vector)
```

```
##  num [1:6] 1.2 2.3 3.4 45 56 7
```

```r
 #Categorical data which takes only a finite number of possible values.

categories <- factor(c("A", "B", "C", "A", "A", "C"))
str(categories)
```

```
##  Factor w/ 3 levels "A","B","C": 1 2 3 1 1 3
```

```r
categories_char <- c("A", "B", "C", "A", "A", "C")
str(categories)
```

```
##  Factor w/ 3 levels "A","B","C": 1 2 3 1 1 3
```

```r
 #If needed, the factors can be given order.
 #Character vector.

ranking <- c("Medium", "High", "Low")
str(ranking)
```

```
##  chr [1:3] "Medium" "High" "Low"
```

```r
 #turn it into ordered factor.

ranking_factors <- ordered(ranking, levels = c("Low", "Medium", "High"))
str(ranking_factors)
```

```
##  Ord.factor w/ 3 levels "Low"<"Medium"<..: 2 3 1
```

```
 #number of elements in a vector can be seen using the length() function.

length(categories)
```

```
## [1] 6
```

```
categories
```

```
## [1] A B C A A C
## Levels: A B C
```

```
length(ranking_factors)
```

```
## [1] 3
```

```
 #simple numeric sequence vectors can be created using shorthand notation.

(my_sequence <- 1:10)
```

```
##  [1]  1  2  3  4  5  6  7  8  9 10
```

```
seq(1, 10, 2)
```

```
## [1] 1 3 5 7 9
```

```
seq(2, 20, 2)
```

```
##  [1]  2  4  6  8 10 12 14 16 18 20
```

```
seq(from = 1, to = 20, by = 5)
```

```
## [1]  1  6 11 16
```

```
 #numeric sequence vector.
 #create a new vector containing vec and the character "HELLO".
 #numeric values have been coerced into their character equivalents.

vec <- 1:10
str(vec)
```

```
##  int [1:10] 1 2 3 4 5 6 7 8 9 10
```

```
new_vec <- c(vec, "HELLO")
str(new_vec)
```

```
##  chr [1:11] "1" "2" "3" "4" "5" "6" "7" "8" "9" "10" "HELLO"
```

```
 #attempt a mixed logical and numeric.
 #logical has been converted to binary numeric (TRUE = 1)
 #try to add a numeric to our previous categories factor vector.
 #categories have been coerced to background integer representations.

vec[1] + vec [2]
```

```
## [1] 3
```

```
mix <- c(TRUE, 6)
str(mix)
```

```
##  num [1:2] 1 6
```

```r
new_categories <- c(categories,1)
str(new_categories)
```

```
##  num [1:7] 1 2 3 1 1 3 1
```

```r
str(categories)
```

```
##  Factor w/ 3 levels "A","B","C": 1 2 3 1 1 3
```

```r
 #Matrices are two-dimensional data structures of the same type and
 #are built from a vector by defining the number of rows and columns.

(m <- matrix(c(1, 2, 3, 4), nrow = 2, ncol = 2))
```

```
##      [,1] [,2]
## [1,]    1    3
## [2,]    2    4
```

```r
(m <- matrix(vec, nrow = 5, ncol = 2))
```

```
##      [,1] [,2]
## [1,]    1    6
## [2,]    2    7
## [3,]    3    8
## [4,]    4    9
## [5,]    5   10
```

Heterogeneous Data Structure

```r
 #List are one-dimensional data structures that can take data of any type.

my_list <- list(6, TRUE, "HELLO")
str(my_list)
```

```
## List of 3
##  $ : num 6
##  $ : logi TRUE
##  $ : chr "HELLO"
```

```r
new_list <- list(scalar = 6, some_vec = c("HELLO"), some_mat = matrix(1:4, nrow = 2, ncol = 4))
str(new_list)
```

```
## List of 3
##  $ scalar  : num 6
##  $ some_vec: chr "HELLO"
##  $ some_mat: int [1:2, 1:4] 1 2 3 4 1 2 3 4
```

```r
new_list[[3]]
```

```
##      [,1] [,2] [,3] [,4]
## [1,]    1    3    1    3
## [2,]    2    4    2    4
```

```r
new_list[[2]]
```

```
## [1] "HELLO"
```

```r
new_list[[1]]
```

```
## [1] 6
```

```
 #named list elements can be accessed by using $.

new_list$scalar
```

```
## [1] 6
```

```
new_list$some_vec
```

```
## [1] "HELLO"
```

```
new_list$some_mat
```

```
##      [,1] [,2] [,3] [,4]
## [1,]    1    3    1    3
## [2,]    2    4    2    4
```

```
 #Data frames - are the most used data structure in R;
          #very similar in nature to a typical database table or spreadsheet.
          #two vectors of different types but same length.

names <- c("John", "Ayesha")
ages <- c(31, 24)

 #Create a data frame.

(df <- data.frame(names, ages))
```

```
##    names ages
## 1   John   31
## 2 Ayesha   24
```

```
 #Get types of columns

str(df)
```

```
## 'data.frame':    2 obs. of  2 variables:
##  $ names: chr  "John" "Ayesha"
##  $ ages : num  31 24
```

```
 #Get dimensions of df

dim(df)
```

```
## [1] 2 2
```

Working with Dataframes

```
#Dataframes is the most common data structure used by analysts in R.
#R facilitates numerous ways of importing data from simple .csv files, from Excel files
 #from online sources or from databases.
```

Loading and tidying data in dataframes

```
#Let's load the salespeople data set, which contains some information on the sales,
 #average customer ratings and performance ratings of salespeople.
```

```
 #read.csv() function can accept a URL address of the file if it is online.

 #url of data set
 #dataframe called salespeople

data()
url <- "https://raw.githubusercontent.com/msuiitdmsgabriel/datasets-regression/main/salespeople.csv"
salespeople <- read.csv(url)

 #View the dimensions.

dim(salespeople)
```

```
## [1] 351   4
```

```
 #If it is too big to display, we can use head() function.

head(salespeople)
```

```
##   promoted sales customer_rate performance
## 1        0   594          3.94           2
## 2        0   446          4.06           3
## 3        1   674          3.83           4
## 4        0   525          3.62           2
## 5        1   657          4.40           3
## 6        1   918          4.54           2
```

```
 #If we wanted to see the 6th entry of sales column.
 #We can view a specific column by using $, and we can use brackets to view specific entry.

salespeople$sales[6]
```

```
## [1] 918
```

```
salespeople[34, 4]
```

```
## [1] 3
```

```
salespeople
```

```
##     promoted sales customer_rate performance
## 1          0   594          3.94           2
## 2          0   446          4.06           3
## 3          1   674          3.83           4
## 4          0   525          3.62           2
## 5          1   657          4.40           3
## 6          1   918          4.54           2
## 7          0   318          3.09           3
## 8          0   364          4.89           1
## 9          0   342          3.74           3
## 10         0   387          3.00           3
## 11         0   527          2.43           3
## 12         1   716          3.16           3
## 13         0   557          3.51           2
## 14         0   450          3.21           3
## 15         0   344          3.02           2
## 16         0   372          3.87           3
```

```
## 17          0   258          2.49          1
## 18          0   338          2.66          4
## 19          0   410          3.14          2
## 20          1   937          5.00          2
## 21          1   702          3.53          4
## 22          0   469          4.24          2
## 23          0   535          4.47          2
## 24          0   342          3.60          1
## 25          1   819          4.45          2
## 26          1   736          3.94          4
## 27          0   330          2.54          2
## 28          0   274          4.06          1
## 29          0   341          4.47          2
## 30          1   717          2.98          2
## 31          0   478          3.48          2
## 32          0   487          3.74          1
## 33          0   239          2.47          4
## 34          1   825          3.32          3
## 35          0   400          3.53          2
## 36          1   728          2.66          3
## 37          1   773          4.89          3
## 38          0   425          3.62          1
## 39          1   943          4.40          4
## 40          0   510          2.56          3
## 41          0   389          3.34          4
## 42          0   270          2.56          2
## 43          1   945          4.31          4
## 44          0   497          3.02          3
## 45          0   329          2.86          3
## 46          0   389          2.98          4
## 47          0   475          3.39          3
## 48          0   383          2.36          2
## 49          1   432          2.33          3
## 50          1   619          1.94          3
## 51          1   578          4.17          4
## 52          0   411          3.07          4
## 53          0   445          3.00          3
## 54          0   440          3.62          2
## 55          0   359          3.92          1
## 56          0   419          3.85          3
## 57          1   840          5.00          4
## 58          0   393          4.49          1
## 59          1   754          3.74          3
## 60          0   441          4.75          2
## 61          1   803          4.89          3
## 62          0   444          4.15          2
## 63          1   753          5.00          4
## 64          1   688          4.29          2
## 65          0   431          4.29          4
## 66          0   511          3.74          2
## 67          0   464          2.22          3
## 68          0   473          3.57          2
## 69          0   532          3.74          1
## 70          0   280          3.41          2
```

```
## 71          0   342          3.71          2
## 72          0   320          2.15          3
## 73          0   531          3.41          4
## 74          0   373          2.01          2
## 75          0   547          4.40          1
## 76          1   611          4.03          4
## 77          1   825          4.66          2
## 78          0   431          3.62          3
## 79          0   401          3.69          2
## 80          0   517          4.20          3
## 81          1   803          4.15          3
## 82          0   586          5.00          1
## 83          0   444          3.21          4
## 84          1   693          3.80          3
## 85          1   659          4.20          1
## 86          0   416          3.87          3
## 87          0   423          2.75          3
## 88          1   756          3.55          4
## 89          0   245          2.52          2
## 90          0   419          3.76          2
## 91          1   757          3.11          3
## 92          1   617          4.33          1
## 93          1   909          3.21          3
## 94          0   516          2.47          1
## 95          0   317          1.51          1
## 96          0   425          3.53          3
## 97          0   528          4.63          2
## 98          0   416          3.37          1
## 99          1   645          4.08          2
## 100         0   390          3.16          4
## 101         0   393          3.76          1
## 102         0   394          3.07          2
## 103         0   387          3.87          3
## 104         0   450          3.62          3
## 105         0   487          3.46          3
## 106         1   607          2.49          4
## 107         0   369          2.22          1
## 108         0   489          4.98          2
## 109         0   324          3.05          3
## 110         0   417          4.47          1
## 111         1   694          1.90          2
## 112         1   651          5.00          4
## 113         0   395          3.46          2
## 114         0   442          2.29          1
## 115         0   422          4.54          3
## 116         0   404          4.06          3
## 117         0   381          3.37          4
## 118         0   501          4.77          4
## 119         1   944          5.00          2
## 120         1   753          4.43          3
## 121         0   591          4.93          4
## 122         1   735          4.03          4
## 123         1   538          3.05          3
## 124         0   451          4.49          2
```

```
## 125        0   477          3.87          3
## 126        0   436          4.13          2
## 127        1   738          3.05          3
## 128        1   902          5.00          4
## 129        0   464          3.90          1
## 130        1   944          3.92          4
## 131        0   285          3.53          3
## 132        0   453          4.68          2
## 133        0   382          3.51          2
## 134        0   414          2.03          2
## 135        0   335          3.71          3
## 136        1   935          5.00          3
## 137        0   203          2.72          2
## 138        0   348          5.00          3
## 139        1   800          4.24          2
## 140        0   436          3.51          3
## 141        0   360          3.23          1
## 142        1   674          4.47          3
## 143        0   425          2.43          3
## 144        1   901          2.70          3
## 145        0   453          4.98          2
## 146        0   350          3.00          3
## 147        0   362          2.89          2
## 148        0   486          3.41          1
## 149        0   471          4.38          2
## 150        0   459          5.00          3
## 151        0   506          5.00          3
## 152        0   262          2.70          2
## 153        1   825          4.95          3
## 154        0   291          2.54          2
## 155        1   464          2.70          3
## 156        1   802          3.78          2
## 157        1   818          4.24          3
## 158        1   736          3.78          3
## 159        0   364          4.01          3
## 160        0   308          4.82          1
## 161        1   862          4.17          4
## 162        0   349          1.67          4
## 163        0   375          3.05          2
## 164        0   423          2.54          3
## 165        1   938          3.69          3
## 166        0   456          2.91          1
## 167        0   517          5.00          2
## 168        0   373          2.93          1
## 169        1   898          2.26          4
## 170        1   777          4.86          3
## 171        0   470          4.84          3
## 172        0   545          3.94          4
## 173        1   699          2.66          4
## 174        1   697          4.06          3
## 175        0   300          1.94          2
## 176        1   677          4.63          3
## 177        0   497          3.14          1
## 178        1   669          4.56          4
```

```
## 179        1   596          4.98          2
## 180        0   492          4.24          3
## 181        0   346          2.20          2
## 182        1   590          4.17          2
## 183        0   592          2.20          3
## 184        1   780          4.15          4
## 185        0   432          4.15          2
## 186        0   418          4.01          2
## 187        1   662          4.56          4
## 188        1   678          4.49          3
## 189        1   716          3.44          3
## 190        0   330          3.05          1
## 191        0   414          3.83          1
## 192        0   416          2.79          2
## 193        0   403          2.75          1
## 194        0   362          2.03          3
## 195        0   284          4.20          3
## 196        0   363          4.72          1
## 197        1   655          3.39          3
## 198        0   597          4.08          3
## 199        1   794          3.83          3
## 200        1   818          2.70          1
## 201        0   409          3.44          1
## 202        1   681          3.97          1
## 203        1   606          1.83          3
## 204        0   489          4.47          2
## 205        0   475          4.56          3
## 206        0   590          4.43          3
## 207        0   396          4.86          2
## 208        0   420          5.00          2
## 209        1   857          3.85          2
## 210        0   371          2.77          2
## 211        0   421          3.39          3
## 212        1   828          1.37          4
## 213        0   594          3.05          1
## 214        0   533          4.86          2
## 215        0   462          2.98          2
## 216        0   392          3.85          3
## 217        0   475          3.83          3
## 218        1   752          4.89          2
## 219        1   659          1.97          2
## 220        1   650          3.14          2
## 221        0   496          4.31          3
## 222        0   211          2.52          1
## 223        1   898          3.51          3
## 224        0   388          2.54          1
## 225        0   383          2.47          2
## 226        0   455          2.36          3
## 227        0   319          3.21          4
## 228        1   756          3.09          3
## 229        0   377          2.08          3
## 230        1   940          2.82          3
## 231        1   757          3.55          3
## 232        0   469          3.85          3
```

```
## 233        0   394         3.57          1
## 234        0   484         2.86          2
## 235        0   491         3.44          4
## 236        0   547         5.00          2
## 237        0   519         3.34          4
## 238        1   739         3.99          3
## 239        0   479         4.06          2
## 240        1   943         3.21          4
## 241        1   742         4.17          2
## 242        0   357         2.72          1
## 243        0   432         3.80          3
## 244        0   584         3.78          2
## 245        1   595         3.74          2
## 246        0   401         2.86          3
## 247        0   460         4.45          2
## 248        1   753         4.89          2
## 249        0   466         5.00          2
## 250        0   362         2.26          2
## 251        0   361         2.66          2
## 252        0   338         4.03          3
## 253        1   882         2.63          3
## 254        0   293         3.51          2
## 255        1   922         4.15          1
## 256        1   793         4.08          2
## 257        1   787         2.56          3
## 258        0   400         3.34          2
## 259        0   516         5.00          4
## 260        0   295         3.87          2
## 261        0   307         1.00          1
## 262        0   151         2.31          2
## 263        0   441         3.34          2
## 264        0   406         3.25          1
## 265        0   270         4.10          2
## 266        1   680         3.09          4
## 267        1   662         4.77          2
## 268        0   347         3.62          3
## 269        0   453         4.86          1
## 270        0   309         3.00          1
## 271        0   592         4.79          2
## 272        0   540         3.41          4
## 273        1   886         4.68          3
## 274        0   420         5.00          4
## 275        1   718         4.03          4
## 276        0   284         3.69          2
## 277        0   323         1.85          3
## 278        0   513         4.20          3
## 279        1   841         5.00          4
## 280        0   362         2.38          1
## 281        1   842         3.99          3
## 282        0   321         3.25          1
## 283        0   516         2.89          3
## 284        0   428         3.28          4
## 285        0   383         2.98          3
## 286        1   521         3.23          1
```

```
## 287        0   358         3.09          2
## 288        0   489         3.41          3
## 289        0   252         1.69          2
## 290        1   720         3.76          3
## 291        1   610         2.75          4
## 292        1   871         5.00          2
## 293        0   594         4.75          3
## 294        0   522         4.59          2
## 295        0   379         1.83          3
## 296        0   454         4.29          2
## 297        0   450         3.69          2
## 298        0   317         2.66          2
## 299        1   835         3.90          1
## 300        0   297         2.61          4
## 301        0   516         3.90          3
## 302        0   355         3.41          2
## 303        1   858         3.67          3
## 304        0   305         1.99          3
## 305        0   410         1.37          3
## 306        1   707         2.38          1
## 307        1   798         4.72          3
## 308        0   265         3.48          2
## 309        1   576         3.60          3
## 310        0   448         3.18          1
## 311        0   590         4.77          3
## 312        0   456         4.03          3
## 313        1   930         4.22          4
## 314        0   412         4.10          2
## 315        0   286         3.64          1
## 316        0   440         2.29          1
## 317        0   546         3.55          1
## 318        0   385         2.66          3
## 319        0   544         3.48          1
## 320        0   505         2.89          1
## 321        1   732         3.57          2
## 322        0   506         4.36          3
## 323        0   394         2.79          4
## 324        1   674         3.60          2
## 325        0   458         3.39          4
## 326        0   251         3.32          2
## 327        0   429         3.41          1
## 328        0   348         3.69          3
## 329        1   789         3.71          3
## 330        1   795         4.31          1
## 331        0   509         4.61          3
## 332        1   754         4.33          4
## 333        0   580         4.70          1
## 334        0   289         3.57          3
## 335        0   390         2.01          3
## 336        1   787         3.14          1
## 337        0   241         3.05          2
## 338        0   522         4.72          2
## 339        0   412         5.00          2
## 340        0   359         5.00          2
```

```
## 341        0   489          4.86           3
## 342        1   940          5.00           4
## 343        0   592          4.38           4
## 344        1   796          5.00           3
## 345        1   653          5.00           3
## 346        0   459          2.82           3
## 347        0   586          3.41           2
## 348        0   401          1.60           3
## 349        0   500          4.17           2
## 350        0   373          2.54           1
## 351        0    NA            NA          NA
```

```
 #We can take a look at the data types using str().

str(salespeople)
```

```
## 'data.frame':    351 obs. of  4 variables:
##  $ promoted     : int  0 0 1 0 1 1 0 0 0 0 ...
##  $ sales        : int  594 446 674 525 657 918 318 364 342 387 ...
##  $ customer_rate: num  3.94 4.06 3.83 3.62 4.4 4.54 3.09 4.89 3.74 3 ...
##  $ performance  : int  2 3 4 2 3 2 3 1 3 3 ...
```

```
 #We can also see a statistical summary of each column using summary().
 #Missing data is identified by a special NA value in R.

summary(salespeople)
```

```
##     promoted          sales       customer_rate    performance
##  Min.   :0.0000   Min.   :151.0   Min.   :1.000   Min.   :1.0
##  1st Qu.:0.0000   1st Qu.:389.2   1st Qu.:3.000   1st Qu.:2.0
##  Median :0.0000   Median :475.0   Median :3.620   Median :3.0
##  Mean   :0.3219   Mean   :527.0   Mean   :3.608   Mean   :2.5
##  3rd Qu.:1.0000   3rd Qu.:667.2   3rd Qu.:4.290   3rd Qu.:3.0
##  Max.   :1.0000   Max.   :945.0   Max.   :5.000   Max.   :4.0
##                   NA's   :1       NA's   :1       NA's   :1
```

```
 #The function is.na() will look at all values in a vector or dataframe
 #and return TRUE or FALSE
  #based on whether they are NA or not.

is.na(salespeople)
```

```
##        promoted sales customer_rate performance
##   [1,]    FALSE FALSE         FALSE       FALSE
##   [2,]    FALSE FALSE         FALSE       FALSE
##   [3,]    FALSE FALSE         FALSE       FALSE
##   [4,]    FALSE FALSE         FALSE       FALSE
##   [5,]    FALSE FALSE         FALSE       FALSE
##   [6,]    FALSE FALSE         FALSE       FALSE
##   [7,]    FALSE FALSE         FALSE       FALSE
##   [8,]    FALSE FALSE         FALSE       FALSE
##   [9,]    FALSE FALSE         FALSE       FALSE
##  [10,]    FALSE FALSE         FALSE       FALSE
##  [11,]    FALSE FALSE         FALSE       FALSE
##  [12,]    FALSE FALSE         FALSE       FALSE
##  [13,]    FALSE FALSE         FALSE       FALSE
```

```
##  [14,]    FALSE FALSE         FALSE       FALSE
##  [15,]    FALSE FALSE         FALSE       FALSE
##  [16,]    FALSE FALSE         FALSE       FALSE
##  [17,]    FALSE FALSE         FALSE       FALSE
##  [18,]    FALSE FALSE         FALSE       FALSE
##  [19,]    FALSE FALSE         FALSE       FALSE
##  [20,]    FALSE FALSE         FALSE       FALSE
##  [21,]    FALSE FALSE         FALSE       FALSE
##  [22,]    FALSE FALSE         FALSE       FALSE
##  [23,]    FALSE FALSE         FALSE       FALSE
##  [24,]    FALSE FALSE         FALSE       FALSE
##  [25,]    FALSE FALSE         FALSE       FALSE
##  [26,]    FALSE FALSE         FALSE       FALSE
##  [27,]    FALSE FALSE         FALSE       FALSE
##  [28,]    FALSE FALSE         FALSE       FALSE
##  [29,]    FALSE FALSE         FALSE       FALSE
##  [30,]    FALSE FALSE         FALSE       FALSE
##  [31,]    FALSE FALSE         FALSE       FALSE
##  [32,]    FALSE FALSE         FALSE       FALSE
##  [33,]    FALSE FALSE         FALSE       FALSE
##  [34,]    FALSE FALSE         FALSE       FALSE
##  [35,]    FALSE FALSE         FALSE       FALSE
##  [36,]    FALSE FALSE         FALSE       FALSE
##  [37,]    FALSE FALSE         FALSE       FALSE
##  [38,]    FALSE FALSE         FALSE       FALSE
##  [39,]    FALSE FALSE         FALSE       FALSE
##  [40,]    FALSE FALSE         FALSE       FALSE
##  [41,]    FALSE FALSE         FALSE       FALSE
##  [42,]    FALSE FALSE         FALSE       FALSE
##  [43,]    FALSE FALSE         FALSE       FALSE
##  [44,]    FALSE FALSE         FALSE       FALSE
##  [45,]    FALSE FALSE         FALSE       FALSE
##  [46,]    FALSE FALSE         FALSE       FALSE
##  [47,]    FALSE FALSE         FALSE       FALSE
##  [48,]    FALSE FALSE         FALSE       FALSE
##  [49,]    FALSE FALSE         FALSE       FALSE
##  [50,]    FALSE FALSE         FALSE       FALSE
##  [51,]    FALSE FALSE         FALSE       FALSE
##  [52,]    FALSE FALSE         FALSE       FALSE
##  [53,]    FALSE FALSE         FALSE       FALSE
##  [54,]    FALSE FALSE         FALSE       FALSE
##  [55,]    FALSE FALSE         FALSE       FALSE
##  [56,]    FALSE FALSE         FALSE       FALSE
##  [57,]    FALSE FALSE         FALSE       FALSE
##  [58,]    FALSE FALSE         FALSE       FALSE
##  [59,]    FALSE FALSE         FALSE       FALSE
##  [60,]    FALSE FALSE         FALSE       FALSE
##  [61,]    FALSE FALSE         FALSE       FALSE
##  [62,]    FALSE FALSE         FALSE       FALSE
##  [63,]    FALSE FALSE         FALSE       FALSE
##  [64,]    FALSE FALSE         FALSE       FALSE
##  [65,]    FALSE FALSE         FALSE       FALSE
##  [66,]    FALSE FALSE         FALSE       FALSE
##  [67,]    FALSE FALSE         FALSE       FALSE
```

```
##  [68,]    FALSE FALSE         FALSE       FALSE
##  [69,]    FALSE FALSE         FALSE       FALSE
##  [70,]    FALSE FALSE         FALSE       FALSE
##  [71,]    FALSE FALSE         FALSE       FALSE
##  [72,]    FALSE FALSE         FALSE       FALSE
##  [73,]    FALSE FALSE         FALSE       FALSE
##  [74,]    FALSE FALSE         FALSE       FALSE
##  [75,]    FALSE FALSE         FALSE       FALSE
##  [76,]    FALSE FALSE         FALSE       FALSE
##  [77,]    FALSE FALSE         FALSE       FALSE
##  [78,]    FALSE FALSE         FALSE       FALSE
##  [79,]    FALSE FALSE         FALSE       FALSE
##  [80,]    FALSE FALSE         FALSE       FALSE
##  [81,]    FALSE FALSE         FALSE       FALSE
##  [82,]    FALSE FALSE         FALSE       FALSE
##  [83,]    FALSE FALSE         FALSE       FALSE
##  [84,]    FALSE FALSE         FALSE       FALSE
##  [85,]    FALSE FALSE         FALSE       FALSE
##  [86,]    FALSE FALSE         FALSE       FALSE
##  [87,]    FALSE FALSE         FALSE       FALSE
##  [88,]    FALSE FALSE         FALSE       FALSE
##  [89,]    FALSE FALSE         FALSE       FALSE
##  [90,]    FALSE FALSE         FALSE       FALSE
##  [91,]    FALSE FALSE         FALSE       FALSE
##  [92,]    FALSE FALSE         FALSE       FALSE
##  [93,]    FALSE FALSE         FALSE       FALSE
##  [94,]    FALSE FALSE         FALSE       FALSE
##  [95,]    FALSE FALSE         FALSE       FALSE
##  [96,]    FALSE FALSE         FALSE       FALSE
##  [97,]    FALSE FALSE         FALSE       FALSE
##  [98,]    FALSE FALSE         FALSE       FALSE
##  [99,]    FALSE FALSE         FALSE       FALSE
## [100,]    FALSE FALSE         FALSE       FALSE
## [101,]    FALSE FALSE         FALSE       FALSE
## [102,]    FALSE FALSE         FALSE       FALSE
## [103,]    FALSE FALSE         FALSE       FALSE
## [104,]    FALSE FALSE         FALSE       FALSE
## [105,]    FALSE FALSE         FALSE       FALSE
## [106,]    FALSE FALSE         FALSE       FALSE
## [107,]    FALSE FALSE         FALSE       FALSE
## [108,]    FALSE FALSE         FALSE       FALSE
## [109,]    FALSE FALSE         FALSE       FALSE
## [110,]    FALSE FALSE         FALSE       FALSE
## [111,]    FALSE FALSE         FALSE       FALSE
## [112,]    FALSE FALSE         FALSE       FALSE
## [113,]    FALSE FALSE         FALSE       FALSE
## [114,]    FALSE FALSE         FALSE       FALSE
## [115,]    FALSE FALSE         FALSE       FALSE
## [116,]    FALSE FALSE         FALSE       FALSE
## [117,]    FALSE FALSE         FALSE       FALSE
## [118,]    FALSE FALSE         FALSE       FALSE
## [119,]    FALSE FALSE         FALSE       FALSE
## [120,]    FALSE FALSE         FALSE       FALSE
## [121,]    FALSE FALSE         FALSE       FALSE
```

```
## [122,]    FALSE FALSE         FALSE       FALSE
## [123,]    FALSE FALSE         FALSE       FALSE
## [124,]    FALSE FALSE         FALSE       FALSE
## [125,]    FALSE FALSE         FALSE       FALSE
## [126,]    FALSE FALSE         FALSE       FALSE
## [127,]    FALSE FALSE         FALSE       FALSE
## [128,]    FALSE FALSE         FALSE       FALSE
## [129,]    FALSE FALSE         FALSE       FALSE
## [130,]    FALSE FALSE         FALSE       FALSE
## [131,]    FALSE FALSE         FALSE       FALSE
## [132,]    FALSE FALSE         FALSE       FALSE
## [133,]    FALSE FALSE         FALSE       FALSE
## [134,]    FALSE FALSE         FALSE       FALSE
## [135,]    FALSE FALSE         FALSE       FALSE
## [136,]    FALSE FALSE         FALSE       FALSE
## [137,]    FALSE FALSE         FALSE       FALSE
## [138,]    FALSE FALSE         FALSE       FALSE
## [139,]    FALSE FALSE         FALSE       FALSE
## [140,]    FALSE FALSE         FALSE       FALSE
## [141,]    FALSE FALSE         FALSE       FALSE
## [142,]    FALSE FALSE         FALSE       FALSE
## [143,]    FALSE FALSE         FALSE       FALSE
## [144,]    FALSE FALSE         FALSE       FALSE
## [145,]    FALSE FALSE         FALSE       FALSE
## [146,]    FALSE FALSE         FALSE       FALSE
## [147,]    FALSE FALSE         FALSE       FALSE
## [148,]    FALSE FALSE         FALSE       FALSE
## [149,]    FALSE FALSE         FALSE       FALSE
## [150,]    FALSE FALSE         FALSE       FALSE
## [151,]    FALSE FALSE         FALSE       FALSE
## [152,]    FALSE FALSE         FALSE       FALSE
## [153,]    FALSE FALSE         FALSE       FALSE
## [154,]    FALSE FALSE         FALSE       FALSE
## [155,]    FALSE FALSE         FALSE       FALSE
## [156,]    FALSE FALSE         FALSE       FALSE
## [157,]    FALSE FALSE         FALSE       FALSE
## [158,]    FALSE FALSE         FALSE       FALSE
## [159,]    FALSE FALSE         FALSE       FALSE
## [160,]    FALSE FALSE         FALSE       FALSE
## [161,]    FALSE FALSE         FALSE       FALSE
## [162,]    FALSE FALSE         FALSE       FALSE
## [163,]    FALSE FALSE         FALSE       FALSE
## [164,]    FALSE FALSE         FALSE       FALSE
## [165,]    FALSE FALSE         FALSE       FALSE
## [166,]    FALSE FALSE         FALSE       FALSE
## [167,]    FALSE FALSE         FALSE       FALSE
## [168,]    FALSE FALSE         FALSE       FALSE
## [169,]    FALSE FALSE         FALSE       FALSE
## [170,]    FALSE FALSE         FALSE       FALSE
## [171,]    FALSE FALSE         FALSE       FALSE
## [172,]    FALSE FALSE         FALSE       FALSE
## [173,]    FALSE FALSE         FALSE       FALSE
## [174,]    FALSE FALSE         FALSE       FALSE
## [175,]    FALSE FALSE         FALSE       FALSE
```

```
## [176,]    FALSE FALSE         FALSE       FALSE
## [177,]    FALSE FALSE         FALSE       FALSE
## [178,]    FALSE FALSE         FALSE       FALSE
## [179,]    FALSE FALSE         FALSE       FALSE
## [180,]    FALSE FALSE         FALSE       FALSE
## [181,]    FALSE FALSE         FALSE       FALSE
## [182,]    FALSE FALSE         FALSE       FALSE
## [183,]    FALSE FALSE         FALSE       FALSE
## [184,]    FALSE FALSE         FALSE       FALSE
## [185,]    FALSE FALSE         FALSE       FALSE
## [186,]    FALSE FALSE         FALSE       FALSE
## [187,]    FALSE FALSE         FALSE       FALSE
## [188,]    FALSE FALSE         FALSE       FALSE
## [189,]    FALSE FALSE         FALSE       FALSE
## [190,]    FALSE FALSE         FALSE       FALSE
## [191,]    FALSE FALSE         FALSE       FALSE
## [192,]    FALSE FALSE         FALSE       FALSE
## [193,]    FALSE FALSE         FALSE       FALSE
## [194,]    FALSE FALSE         FALSE       FALSE
## [195,]    FALSE FALSE         FALSE       FALSE
## [196,]    FALSE FALSE         FALSE       FALSE
## [197,]    FALSE FALSE         FALSE       FALSE
## [198,]    FALSE FALSE         FALSE       FALSE
## [199,]    FALSE FALSE         FALSE       FALSE
## [200,]    FALSE FALSE         FALSE       FALSE
## [201,]    FALSE FALSE         FALSE       FALSE
## [202,]    FALSE FALSE         FALSE       FALSE
## [203,]    FALSE FALSE         FALSE       FALSE
## [204,]    FALSE FALSE         FALSE       FALSE
## [205,]    FALSE FALSE         FALSE       FALSE
## [206,]    FALSE FALSE         FALSE       FALSE
## [207,]    FALSE FALSE         FALSE       FALSE
## [208,]    FALSE FALSE         FALSE       FALSE
## [209,]    FALSE FALSE         FALSE       FALSE
## [210,]    FALSE FALSE         FALSE       FALSE
## [211,]    FALSE FALSE         FALSE       FALSE
## [212,]    FALSE FALSE         FALSE       FALSE
## [213,]    FALSE FALSE         FALSE       FALSE
## [214,]    FALSE FALSE         FALSE       FALSE
## [215,]    FALSE FALSE         FALSE       FALSE
## [216,]    FALSE FALSE         FALSE       FALSE
## [217,]    FALSE FALSE         FALSE       FALSE
## [218,]    FALSE FALSE         FALSE       FALSE
## [219,]    FALSE FALSE         FALSE       FALSE
## [220,]    FALSE FALSE         FALSE       FALSE
## [221,]    FALSE FALSE         FALSE       FALSE
## [222,]    FALSE FALSE         FALSE       FALSE
## [223,]    FALSE FALSE         FALSE       FALSE
## [224,]    FALSE FALSE         FALSE       FALSE
## [225,]    FALSE FALSE         FALSE       FALSE
## [226,]    FALSE FALSE         FALSE       FALSE
## [227,]    FALSE FALSE         FALSE       FALSE
## [228,]    FALSE FALSE         FALSE       FALSE
## [229,]    FALSE FALSE         FALSE       FALSE
```

```
## [230,]    FALSE FALSE         FALSE       FALSE
## [231,]    FALSE FALSE         FALSE       FALSE
## [232,]    FALSE FALSE         FALSE       FALSE
## [233,]    FALSE FALSE         FALSE       FALSE
## [234,]    FALSE FALSE         FALSE       FALSE
## [235,]    FALSE FALSE         FALSE       FALSE
## [236,]    FALSE FALSE         FALSE       FALSE
## [237,]    FALSE FALSE         FALSE       FALSE
## [238,]    FALSE FALSE         FALSE       FALSE
## [239,]    FALSE FALSE         FALSE       FALSE
## [240,]    FALSE FALSE         FALSE       FALSE
## [241,]    FALSE FALSE         FALSE       FALSE
## [242,]    FALSE FALSE         FALSE       FALSE
## [243,]    FALSE FALSE         FALSE       FALSE
## [244,]    FALSE FALSE         FALSE       FALSE
## [245,]    FALSE FALSE         FALSE       FALSE
## [246,]    FALSE FALSE         FALSE       FALSE
## [247,]    FALSE FALSE         FALSE       FALSE
## [248,]    FALSE FALSE         FALSE       FALSE
## [249,]    FALSE FALSE         FALSE       FALSE
## [250,]    FALSE FALSE         FALSE       FALSE
## [251,]    FALSE FALSE         FALSE       FALSE
## [252,]    FALSE FALSE         FALSE       FALSE
## [253,]    FALSE FALSE         FALSE       FALSE
## [254,]    FALSE FALSE         FALSE       FALSE
## [255,]    FALSE FALSE         FALSE       FALSE
## [256,]    FALSE FALSE         FALSE       FALSE
## [257,]    FALSE FALSE         FALSE       FALSE
## [258,]    FALSE FALSE         FALSE       FALSE
## [259,]    FALSE FALSE         FALSE       FALSE
## [260,]    FALSE FALSE         FALSE       FALSE
## [261,]    FALSE FALSE         FALSE       FALSE
## [262,]    FALSE FALSE         FALSE       FALSE
## [263,]    FALSE FALSE         FALSE       FALSE
## [264,]    FALSE FALSE         FALSE       FALSE
## [265,]    FALSE FALSE         FALSE       FALSE
## [266,]    FALSE FALSE         FALSE       FALSE
## [267,]    FALSE FALSE         FALSE       FALSE
## [268,]    FALSE FALSE         FALSE       FALSE
## [269,]    FALSE FALSE         FALSE       FALSE
## [270,]    FALSE FALSE         FALSE       FALSE
## [271,]    FALSE FALSE         FALSE       FALSE
## [272,]    FALSE FALSE         FALSE       FALSE
## [273,]    FALSE FALSE         FALSE       FALSE
## [274,]    FALSE FALSE         FALSE       FALSE
## [275,]    FALSE FALSE         FALSE       FALSE
## [276,]    FALSE FALSE         FALSE       FALSE
## [277,]    FALSE FALSE         FALSE       FALSE
## [278,]    FALSE FALSE         FALSE       FALSE
## [279,]    FALSE FALSE         FALSE       FALSE
## [280,]    FALSE FALSE         FALSE       FALSE
## [281,]    FALSE FALSE         FALSE       FALSE
## [282,]    FALSE FALSE         FALSE       FALSE
## [283,]    FALSE FALSE         FALSE       FALSE
```

```
## [284,]    FALSE FALSE         FALSE       FALSE
## [285,]    FALSE FALSE         FALSE       FALSE
## [286,]    FALSE FALSE         FALSE       FALSE
## [287,]    FALSE FALSE         FALSE       FALSE
## [288,]    FALSE FALSE         FALSE       FALSE
## [289,]    FALSE FALSE         FALSE       FALSE
## [290,]    FALSE FALSE         FALSE       FALSE
## [291,]    FALSE FALSE         FALSE       FALSE
## [292,]    FALSE FALSE         FALSE       FALSE
## [293,]    FALSE FALSE         FALSE       FALSE
## [294,]    FALSE FALSE         FALSE       FALSE
## [295,]    FALSE FALSE         FALSE       FALSE
## [296,]    FALSE FALSE         FALSE       FALSE
## [297,]    FALSE FALSE         FALSE       FALSE
## [298,]    FALSE FALSE         FALSE       FALSE
## [299,]    FALSE FALSE         FALSE       FALSE
## [300,]    FALSE FALSE         FALSE       FALSE
## [301,]    FALSE FALSE         FALSE       FALSE
## [302,]    FALSE FALSE         FALSE       FALSE
## [303,]    FALSE FALSE         FALSE       FALSE
## [304,]    FALSE FALSE         FALSE       FALSE
## [305,]    FALSE FALSE         FALSE       FALSE
## [306,]    FALSE FALSE         FALSE       FALSE
## [307,]    FALSE FALSE         FALSE       FALSE
## [308,]    FALSE FALSE         FALSE       FALSE
## [309,]    FALSE FALSE         FALSE       FALSE
## [310,]    FALSE FALSE         FALSE       FALSE
## [311,]    FALSE FALSE         FALSE       FALSE
## [312,]    FALSE FALSE         FALSE       FALSE
## [313,]    FALSE FALSE         FALSE       FALSE
## [314,]    FALSE FALSE         FALSE       FALSE
## [315,]    FALSE FALSE         FALSE       FALSE
## [316,]    FALSE FALSE         FALSE       FALSE
## [317,]    FALSE FALSE         FALSE       FALSE
## [318,]    FALSE FALSE         FALSE       FALSE
## [319,]    FALSE FALSE         FALSE       FALSE
## [320,]    FALSE FALSE         FALSE       FALSE
## [321,]    FALSE FALSE         FALSE       FALSE
## [322,]    FALSE FALSE         FALSE       FALSE
## [323,]    FALSE FALSE         FALSE       FALSE
## [324,]    FALSE FALSE         FALSE       FALSE
## [325,]    FALSE FALSE         FALSE       FALSE
## [326,]    FALSE FALSE         FALSE       FALSE
## [327,]    FALSE FALSE         FALSE       FALSE
## [328,]    FALSE FALSE         FALSE       FALSE
## [329,]    FALSE FALSE         FALSE       FALSE
## [330,]    FALSE FALSE         FALSE       FALSE
## [331,]    FALSE FALSE         FALSE       FALSE
## [332,]    FALSE FALSE         FALSE       FALSE
## [333,]    FALSE FALSE         FALSE       FALSE
## [334,]    FALSE FALSE         FALSE       FALSE
## [335,]    FALSE FALSE         FALSE       FALSE
## [336,]    FALSE FALSE         FALSE       FALSE
## [337,]    FALSE FALSE         FALSE       FALSE
```

```
## [338,]    FALSE FALSE         FALSE       FALSE
## [339,]    FALSE FALSE         FALSE       FALSE
## [340,]    FALSE FALSE         FALSE       FALSE
## [341,]    FALSE FALSE         FALSE       FALSE
## [342,]    FALSE FALSE         FALSE       FALSE
## [343,]    FALSE FALSE         FALSE       FALSE
## [344,]    FALSE FALSE         FALSE       FALSE
## [345,]    FALSE FALSE         FALSE       FALSE
## [346,]    FALSE FALSE         FALSE       FALSE
## [347,]    FALSE FALSE         FALSE       FALSE
## [348,]    FALSE FALSE         FALSE       FALSE
## [349,]    FALSE FALSE         FALSE       FALSE
## [350,]    FALSE FALSE         FALSE       FALSE
## [351,]    FALSE  TRUE          TRUE        TRUE
```

```
 #By adding these up using the sum() function, it will take TRUE as 1 and FALSE as 0,
  #which effectively provides a count of missing data.

sum(is.na(salespeople))
```

```
## [1] 3
```

```
 #The easiest way is to use the complete.cases() function,
 #which identifies the rows that have no NAs,
  #and then we can select those rows from the dataframe based on that conditon.

complete.cases(salespeople)
```

```
##   [1]  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE
##  [13]  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE
##  [25]  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE
##  [37]  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE
##  [49]  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE
##  [61]  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE
##  [73]  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE
##  [85]  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE
##  [97]  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE
## [109]  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE
## [121]  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE
## [133]  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE
## [145]  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE
## [157]  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE
## [169]  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE
## [181]  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE
## [193]  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE
## [205]  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE
## [217]  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE
## [229]  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE
## [241]  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE
## [253]  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE
## [265]  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE
## [277]  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE
## [289]  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE
## [301]  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE
## [313]  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE
```

```
## [325]  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE
## [337]  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE
## [349]  TRUE  TRUE FALSE
```

```
salespeople[complete.cases(salespeople),]
```

```
##     promoted sales customer_rate performance
## 1          0   594          3.94           2
## 2          0   446          4.06           3
## 3          1   674          3.83           4
## 4          0   525          3.62           2
## 5          1   657          4.40           3
## 6          1   918          4.54           2
## 7          0   318          3.09           3
## 8          0   364          4.89           1
## 9          0   342          3.74           3
## 10         0   387          3.00           3
## 11         0   527          2.43           3
## 12         1   716          3.16           3
## 13         0   557          3.51           2
## 14         0   450          3.21           3
## 15         0   344          3.02           2
## 16         0   372          3.87           3
## 17         0   258          2.49           1
## 18         0   338          2.66           4
## 19         0   410          3.14           2
## 20         1   937          5.00           2
## 21         1   702          3.53           4
## 22         0   469          4.24           2
## 23         0   535          4.47           2
## 24         0   342          3.60           1
## 25         1   819          4.45           2
## 26         1   736          3.94           4
## 27         0   330          2.54           2
## 28         0   274          4.06           1
## 29         0   341          4.47           2
## 30         1   717          2.98           2
## 31         0   478          3.48           2
## 32         0   487          3.74           1
## 33         0   239          2.47           4
## 34         1   825          3.32           3
## 35         0   400          3.53           2
## 36         1   728          2.66           3
## 37         1   773          4.89           3
## 38         0   425          3.62           1
## 39         1   943          4.40           4
## 40         0   510          2.56           3
## 41         0   389          3.34           4
## 42         0   270          2.56           2
## 43         1   945          4.31           4
## 44         0   497          3.02           3
## 45         0   329          2.86           3
## 46         0   389          2.98           4
## 47         0   475          3.39           3
## 48         0   383          2.36           2
```

```
## 49          1   432          2.33          3
## 50          1   619          1.94          3
## 51          1   578          4.17          4
## 52          0   411          3.07          4
## 53          0   445          3.00          3
## 54          0   440          3.62          2
## 55          0   359          3.92          1
## 56          0   419          3.85          3
## 57          1   840          5.00          4
## 58          0   393          4.49          1
## 59          1   754          3.74          3
## 60          0   441          4.75          2
## 61          1   803          4.89          3
## 62          0   444          4.15          2
## 63          1   753          5.00          4
## 64          1   688          4.29          2
## 65          0   431          4.29          4
## 66          0   511          3.74          2
## 67          0   464          2.22          3
## 68          0   473          3.57          2
## 69          0   532          3.74          1
## 70          0   280          3.41          2
## 71          0   342          3.71          2
## 72          0   320          2.15          3
## 73          0   531          3.41          4
## 74          0   373          2.01          2
## 75          0   547          4.40          1
## 76          1   611          4.03          4
## 77          1   825          4.66          2
## 78          0   431          3.62          3
## 79          0   401          3.69          2
## 80          0   517          4.20          3
## 81          1   803          4.15          3
## 82          0   586          5.00          1
## 83          0   444          3.21          4
## 84          1   693          3.80          3
## 85          1   659          4.20          1
## 86          0   416          3.87          3
## 87          0   423          2.75          3
## 88          1   756          3.55          4
## 89          0   245          2.52          2
## 90          0   419          3.76          2
## 91          1   757          3.11          3
## 92          1   617          4.33          1
## 93          1   909          3.21          3
## 94          0   516          2.47          1
## 95          0   317          1.51          1
## 96          0   425          3.53          3
## 97          0   528          4.63          2
## 98          0   416          3.37          1
## 99          1   645          4.08          2
## 100         0   390          3.16          4
## 101         0   393          3.76          1
## 102         0   394          3.07          2
```

```
## 103        0   387          3.87          3
## 104        0   450          3.62          3
## 105        0   487          3.46          3
## 106        1   607          2.49          4
## 107        0   369          2.22          1
## 108        0   489          4.98          2
## 109        0   324          3.05          3
## 110        0   417          4.47          1
## 111        1   694          1.90          2
## 112        1   651          5.00          4
## 113        0   395          3.46          2
## 114        0   442          2.29          1
## 115        0   422          4.54          3
## 116        0   404          4.06          3
## 117        0   381          3.37          4
## 118        0   501          4.77          4
## 119        1   944          5.00          2
## 120        1   753          4.43          3
## 121        0   591          4.93          4
## 122        1   735          4.03          4
## 123        1   538          3.05          3
## 124        0   451          4.49          2
## 125        0   477          3.87          3
## 126        0   436          4.13          2
## 127        1   738          3.05          3
## 128        1   902          5.00          4
## 129        0   464          3.90          1
## 130        1   944          3.92          4
## 131        0   285          3.53          3
## 132        0   453          4.68          2
## 133        0   382          3.51          2
## 134        0   414          2.03          2
## 135        0   335          3.71          3
## 136        1   935          5.00          3
## 137        0   203          2.72          2
## 138        0   348          5.00          3
## 139        1   800          4.24          2
## 140        0   436          3.51          3
## 141        0   360          3.23          1
## 142        1   674          4.47          3
## 143        0   425          2.43          3
## 144        1   901          2.70          3
## 145        0   453          4.98          2
## 146        0   350          3.00          3
## 147        0   362          2.89          2
## 148        0   486          3.41          1
## 149        0   471          4.38          2
## 150        0   459          5.00          3
## 151        0   506          5.00          3
## 152        0   262          2.70          2
## 153        1   825          4.95          3
## 154        0   291          2.54          2
## 155        1   464          2.70          3
## 156        1   802          3.78          2
```

```
## 157        1   818          4.24          3
## 158        1   736          3.78          3
## 159        0   364          4.01          3
## 160        0   308          4.82          1
## 161        1   862          4.17          4
## 162        0   349          1.67          4
## 163        0   375          3.05          2
## 164        0   423          2.54          3
## 165        1   938          3.69          3
## 166        0   456          2.91          1
## 167        0   517          5.00          2
## 168        0   373          2.93          1
## 169        1   898          2.26          4
## 170        1   777          4.86          3
## 171        0   470          4.84          3
## 172        0   545          3.94          4
## 173        1   699          2.66          4
## 174        1   697          4.06          3
## 175        0   300          1.94          2
## 176        1   677          4.63          3
## 177        0   497          3.14          1
## 178        1   669          4.56          4
## 179        1   596          4.98          2
## 180        0   492          4.24          3
## 181        0   346          2.20          2
## 182        1   590          4.17          2
## 183        0   592          2.20          3
## 184        1   780          4.15          4
## 185        0   432          4.15          2
## 186        0   418          4.01          2
## 187        1   662          4.56          4
## 188        1   678          4.49          3
## 189        1   716          3.44          3
## 190        0   330          3.05          1
## 191        0   414          3.83          1
## 192        0   416          2.79          2
## 193        0   403          2.75          1
## 194        0   362          2.03          3
## 195        0   284          4.20          3
## 196        0   363          4.72          1
## 197        1   655          3.39          3
## 198        0   597          4.08          3
## 199        1   794          3.83          3
## 200        1   818          2.70          1
## 201        0   409          3.44          1
## 202        1   681          3.97          1
## 203        1   606          1.83          3
## 204        0   489          4.47          2
## 205        0   475          4.56          3
## 206        0   590          4.43          3
## 207        0   396          4.86          2
## 208        0   420          5.00          2
## 209        1   857          3.85          2
## 210        0   371          2.77          2
```

```
## 211        0   421          3.39          3
## 212        1   828          1.37          4
## 213        0   594          3.05          1
## 214        0   533          4.86          2
## 215        0   462          2.98          2
## 216        0   392          3.85          3
## 217        0   475          3.83          3
## 218        1   752          4.89          2
## 219        1   659          1.97          2
## 220        1   650          3.14          2
## 221        0   496          4.31          3
## 222        0   211          2.52          1
## 223        1   898          3.51          3
## 224        0   388          2.54          1
## 225        0   383          2.47          2
## 226        0   455          2.36          3
## 227        0   319          3.21          4
## 228        1   756          3.09          3
## 229        0   377          2.08          3
## 230        1   940          2.82          3
## 231        1   757          3.55          3
## 232        0   469          3.85          3
## 233        0   394          3.57          1
## 234        0   484          2.86          2
## 235        0   491          3.44          4
## 236        0   547          5.00          2
## 237        0   519          3.34          4
## 238        1   739          3.99          3
## 239        0   479          4.06          2
## 240        1   943          3.21          4
## 241        1   742          4.17          2
## 242        0   357          2.72          1
## 243        0   432          3.80          3
## 244        0   584          3.78          2
## 245        1   595          3.74          2
## 246        0   401          2.86          3
## 247        0   460          4.45          2
## 248        1   753          4.89          2
## 249        0   466          5.00          2
## 250        0   362          2.26          2
## 251        0   361          2.66          2
## 252        0   338          4.03          3
## 253        1   882          2.63          3
## 254        0   293          3.51          2
## 255        1   922          4.15          1
## 256        1   793          4.08          2
## 257        1   787          2.56          3
## 258        0   400          3.34          2
## 259        0   516          5.00          4
## 260        0   295          3.87          2
## 261        0   307          1.00          1
## 262        0   151          2.31          2
## 263        0   441          3.34          2
## 264        0   406          3.25          1
```

```
## 265        0   270         4.10          2
## 266        1   680         3.09          4
## 267        1   662         4.77          2
## 268        0   347         3.62          3
## 269        0   453         4.86          1
## 270        0   309         3.00          1
## 271        0   592         4.79          2
## 272        0   540         3.41          4
## 273        1   886         4.68          3
## 274        0   420         5.00          4
## 275        1   718         4.03          4
## 276        0   284         3.69          2
## 277        0   323         1.85          3
## 278        0   513         4.20          3
## 279        1   841         5.00          4
## 280        0   362         2.38          1
## 281        1   842         3.99          3
## 282        0   321         3.25          1
## 283        0   516         2.89          3
## 284        0   428         3.28          4
## 285        0   383         2.98          3
## 286        1   521         3.23          1
## 287        0   358         3.09          2
## 288        0   489         3.41          3
## 289        0   252         1.69          2
## 290        1   720         3.76          3
## 291        1   610         2.75          4
## 292        1   871         5.00          2
## 293        0   594         4.75          3
## 294        0   522         4.59          2
## 295        0   379         1.83          3
## 296        0   454         4.29          2
## 297        0   450         3.69          2
## 298        0   317         2.66          2
## 299        1   835         3.90          1
## 300        0   297         2.61          4
## 301        0   516         3.90          3
## 302        0   355         3.41          2
## 303        1   858         3.67          3
## 304        0   305         1.99          3
## 305        0   410         1.37          3
## 306        1   707         2.38          1
## 307        1   798         4.72          3
## 308        0   265         3.48          2
## 309        1   576         3.60          3
## 310        0   448         3.18          1
## 311        0   590         4.77          3
## 312        0   456         4.03          3
## 313        1   930         4.22          4
## 314        0   412         4.10          2
## 315        0   286         3.64          1
## 316        0   440         2.29          1
## 317        0   546         3.55          1
## 318        0   385         2.66          3
```

```
## 319        0   544          3.48           1
## 320        0   505          2.89           1
## 321        1   732          3.57           2
## 322        0   506          4.36           3
## 323        0   394          2.79           4
## 324        1   674          3.60           2
## 325        0   458          3.39           4
## 326        0   251          3.32           2
## 327        0   429          3.41           1
## 328        0   348          3.69           3
## 329        1   789          3.71           3
## 330        1   795          4.31           1
## 331        0   509          4.61           3
## 332        1   754          4.33           4
## 333        0   580          4.70           1
## 334        0   289          3.57           3
## 335        0   390          2.01           3
## 336        1   787          3.14           1
## 337        0   241          3.05           2
## 338        0   522          4.72           2
## 339        0   412          5.00           2
## 340        0   359          5.00           2
## 341        0   489          4.86           3
## 342        1   940          5.00           4
## 343        0   592          4.38           4
## 344        1   796          5.00           3
## 345        1   653          5.00           3
## 346        0   459          2.82           3
## 347        0   586          3.41           2
## 348        0   401          1.60           3
## 349        0   500          4.17           2
## 350        0   373          2.54           1
```

```
salespeople <- salespeople[complete.cases(salespeople),]

 #confirm no NAs.

sum(is.na(salespeople))
```

```
## [1] 0
```

```
 #We can see the unique values of a vector or column using the unique() function.

salespeople$performance
```

```
##   [1] 2 3 4 2 3 2 3 1 3 3 3 3 2 3 2 3 1 4 2 2 4 2 2 1 2 4 2 1 2 2 2 1 4 3 2 3 3
##  [38] 1 4 3 4 2 4 3 3 4 3 2 3 3 4 4 3 2 1 3 4 1 3 2 3 2 4 2 4 2 3 2 1 2 2 3 4 2
##  [75] 1 4 2 3 2 3 3 1 4 3 1 3 3 4 2 2 3 1 3 1 1 3 2 1 2 4 1 2 3 3 3 4 1 2 3 1 2
## [112] 4 2 1 3 3 4 4 2 3 4 4 3 2 3 2 3 4 1 4 3 2 2 2 3 3 2 3 2 3 1 3 3 3 2 3 2 1
## [149] 2 3 3 2 3 2 3 2 3 3 3 1 4 4 2 3 3 1 2 1 4 3 3 4 4 3 2 3 1 4 2 3 2 2 3 4 2
## [186] 2 4 3 3 1 1 2 1 3 3 1 3 3 3 1 1 1 3 2 3 3 2 2 2 2 3 4 1 2 2 3 3 2 2 2 3 1
## [223] 3 1 2 3 4 3 3 3 3 3 1 2 4 2 4 3 2 4 2 1 3 2 2 3 2 2 2 2 2 3 3 2 1 2 3 2 4
## [260] 2 1 2 2 1 2 4 2 3 1 1 2 4 3 4 4 2 3 3 4 1 3 1 3 4 3 1 2 3 2 3 4 2 3 2 3 2
## [297] 2 2 1 4 3 2 3 3 3 1 3 2 3 1 3 3 4 2 1 1 1 3 1 1 2 3 4 2 4 2 1 3 3 1 3 4 1
## [334] 3 3 1 2 2 2 2 3 4 4 3 3 3 2 3 2 1
```

```r
unique(salespeople$performance)
```

```
## [1] 2 3 4 1
```

```r
 #If we need to change the type of a column in a dataframe, we can use the
  # as.numeric(), as.character(), as.logical() or as.factor() functions.

as.factor(salespeople$performance)
```

```
##   [1] 2 3 4 2 3 2 3 1 3 3 3 3 2 3 2 3 1 4 2 2 4 2 2 1 2 4 2 1 2 2 2 1 4 3 2 3 3
##  [38] 1 4 3 4 2 4 3 3 4 3 2 3 3 4 4 3 2 1 3 4 1 3 2 3 2 4 2 4 2 3 2 1 2 2 3 4 2
##  [75] 1 4 2 3 2 3 3 1 4 3 1 3 3 4 2 2 3 1 3 1 1 3 2 1 2 4 1 2 3 3 3 4 1 2 3 1 2
## [112] 4 2 1 3 3 4 4 2 3 4 4 3 2 3 2 3 4 1 4 3 2 2 2 3 3 2 3 2 3 1 3 3 3 2 3 2 1
## [149] 2 3 3 2 3 2 3 2 3 3 3 1 4 4 2 3 3 1 2 1 4 3 3 4 4 3 2 3 1 4 2 3 2 2 3 4 2
## [186] 2 4 3 3 1 1 2 1 3 3 1 3 3 3 1 1 1 3 2 3 3 2 2 2 2 3 4 1 2 2 3 3 2 2 2 3 1
## [223] 3 1 2 3 4 3 3 3 3 3 1 2 4 2 4 3 2 4 2 1 3 2 2 3 2 2 2 2 2 3 3 2 1 2 3 2 4
## [260] 2 1 2 2 1 2 4 2 3 1 1 2 4 3 4 4 2 3 3 4 1 3 1 3 4 3 1 2 3 2 3 4 2 3 2 3 2
## [297] 2 2 1 4 3 2 3 3 3 1 3 2 3 1 3 3 4 2 1 1 1 3 1 1 2 3 4 2 4 2 1 3 3 1 3 4 1
## [334] 3 3 1 2 2 2 2 3 4 4 3 3 3 2 3 2 1
## Levels: 1 2 3 4
```

```r
str(salespeople)
```

```
## 'data.frame':    350 obs. of  4 variables:
##  $ promoted     : int  0 0 1 0 1 1 0 0 0 0 ...
##  $ sales        : int  594 446 674 525 657 918 318 364 342 387 ...
##  $ customer_rate: num  3.94 4.06 3.83 3.62 4.4 4.54 3.09 4.89 3.74 3 ...
##  $ performance  : int  2 3 4 2 3 2 3 1 3 3 ...
```

```r
salespeople$performance <- as.factor(salespeople$performance)
str(salespeople)
```

```
## 'data.frame':    350 obs. of  4 variables:
##  $ promoted     : int  0 0 1 0 1 1 0 0 0 0 ...
##  $ sales        : int  594 446 674 525 657 918 318 364 342 387 ...
##  $ customer_rate: num  3.94 4.06 3.83 3.62 4.4 4.54 3.09 4.89 3.74 3 ...
##  $ performance  : Factor w/ 4 levels "1","2","3","4": 2 3 4 2 3 2 3 1 3 3 ...
```

Manipulating dataframes

```r
 #Dataframes can be subsetted to contain only rows that satisfy specific conditions.
url <- "https://raw.githubusercontent.com/msuiitdmsgabriel/datasets-regression/main/salespeople.csv"
salespeople <- read.csv(url)

(sales_720 <- subset(salespeople, subset = sales == 720))
```

```
##     promoted sales customer_rate performance
## 290        1   720          3.76           3
```

```r
 #Note the use of ==, which is used in many programming languages,
  #to test for precise equality.

unique((salespeople$sales))
```

```
##   [1] 594 446 674 525 657 918 318 364 342 387 527 716 557 450 344 372 258 338
##  [19] 410 937 702 469 535 819 736 330 274 341 717 478 487 239 825 400 728 773
##  [37] 425 943 510 389 270 945 497 329 475 383 432 619 578 411 445 440 359 419
```

```
##  [55] 840 393 754 441 803 444 753 688 431 511 464 473 532 280 320 531 373 547
##  [73] 611 401 517 586 693 659 416 423 756 245 757 617 909 516 317 528 645 390
##  [91] 394 607 369 489 324 417 694 651 395 442 422 404 381 501 944 591 735 538
## [109] 451 477 436 738 902 285 453 382 414 335 935 203 348 800 360 901 350 362
## [127] 486 471 459 506 262 291 802 818 308 862 349 375 938 456 898 777 470 545
## [145] 699 697 300 677 669 596 492 346 590 592 780 418 662 678 403 284 363 655
## [163] 597 794 409 681 606 396 420 857 371 421 828 533 462 392 752 650 496 211
## [181] 388 455 319 377 940 484 491 519 739 479 742 357 584 595 460 466 361 882
## [199] 293 922 793 787 295 307 151 406 680 347 309 540 886 718 323 513 841 842
## [217] 321 428 521 358 252 720 610 871 522 379 454 835 297 355 858 305 707 798
## [235] 265 576 448 930 412 286 546 385 544 505 732 458 251 429 789 795 509 580
## [253] 289 241 796 653 500  NA
```

```
 #Similarly we can select columns based on inequalities

(subset(salespeople, subset = sales >=700))
```

```
##     promoted sales customer_rate performance
## 6          1   918          4.54           2
## 12         1   716          3.16           3
## 20         1   937          5.00           2
## 21         1   702          3.53           4
## 25         1   819          4.45           2
## 26         1   736          3.94           4
## 30         1   717          2.98           2
## 34         1   825          3.32           3
## 36         1   728          2.66           3
## 37         1   773          4.89           3
## 39         1   943          4.40           4
## 43         1   945          4.31           4
## 57         1   840          5.00           4
## 59         1   754          3.74           3
## 61         1   803          4.89           3
## 63         1   753          5.00           4
## 77         1   825          4.66           2
## 81         1   803          4.15           3
## 88         1   756          3.55           4
## 91         1   757          3.11           3
## 93         1   909          3.21           3
## 119        1   944          5.00           2
## 120        1   753          4.43           3
## 122        1   735          4.03           4
## 127        1   738          3.05           3
## 128        1   902          5.00           4
## 130        1   944          3.92           4
## 136        1   935          5.00           3
## 139        1   800          4.24           2
## 144        1   901          2.70           3
## 153        1   825          4.95           3
## 156        1   802          3.78           2
## 157        1   818          4.24           3
## 158        1   736          3.78           3
## 161        1   862          4.17           4
## 165        1   938          3.69           3
## 169        1   898          2.26           4
```

```
## 170        1   777          4.86           3
## 184        1   780          4.15           4
## 189        1   716          3.44           3
## 199        1   794          3.83           3
## 200        1   818          2.70           1
## 209        1   857          3.85           2
## 212        1   828          1.37           4
## 218        1   752          4.89           2
## 223        1   898          3.51           3
## 228        1   756          3.09           3
## 230        1   940          2.82           3
## 231        1   757          3.55           3
## 238        1   739          3.99           3
## 240        1   943          3.21           4
## 241        1   742          4.17           2
## 248        1   753          4.89           2
## 253        1   882          2.63           3
## 255        1   922          4.15           1
## 256        1   793          4.08           2
## 257        1   787          2.56           3
## 273        1   886          4.68           3
## 275        1   718          4.03           4
## 279        1   841          5.00           4
## 281        1   842          3.99           3
## 290        1   720          3.76           3
## 292        1   871          5.00           2
## 299        1   835          3.90           1
## 303        1   858          3.67           3
## 306        1   707          2.38           1
## 307        1   798          4.72           3
## 313        1   930          4.22           4
## 321        1   732          3.57           2
## 329        1   789          3.71           3
## 330        1   795          4.31           1
## 332        1   754          4.33           4
## 336        1   787          3.14           1
## 342        1   940          5.00           4
## 344        1   796          5.00           3
```

```
high_sales <- subset(salespeople, subset = sales >= 700)
head(high_sales)
```

```
##    promoted sales customer_rate performance
## 6         1   918          4.54           2
## 12        1   716          3.16           3
## 20        1   937          5.00           2
## 21        1   702          3.53           4
## 25        1   819          4.45           2
## 26        1   736          3.94           4
```

```
 #To select specific columns use the select argument.

salespeople_sales_perf <- subset(salespeople, select = c("sales", "performance"))
head(salespeople_sales_perf)
```

```
##   sales performance
## 1   594           2
## 2   446           3
## 3   674           4
## 4   525           2
## 5   657           3
## 6   918           2
```

```
 #bind the rows of low_sales and high_sales together.

low_sales <- subset(salespeople, subet = sales < 400)
```

```
## Warning: In subset.data.frame(salespeople, subet = sales < 400) :
##  extra argument 'subet' will be disregarded
```

```
low_and_high_sales = rbind(low_sales, high_sales)
head(low_and_high_sales)
```

```
##   promoted sales customer_rate performance
## 1        0   594          3.94           2
## 2        0   446          4.06           3
## 3        1   674          3.83           4
## 4        0   525          3.62           2
## 5        1   657          4.40           3
## 6        1   918          4.54           2
```

```
head(high_sales)
```

```
##    promoted sales customer_rate performance
## 6         1   918          4.54           2
## 12        1   716          3.16           3
## 20        1   937          5.00           2
## 21        1   702          3.53           4
## 25        1   819          4.45           2
## 26        1   736          3.94           4
```

```
 #Two dataframes with two columns each.
 #Bind the columns to create a dataframe with four columns.

sales_perf <- subset(salespeople, select = c("sales", "performance"))
prom_custrate <- subset(salespeople, select = c("promoted", "customer_rate"))
full_df <- cbind(sales_perf, prom_custrate)
head(full_df)
```

```
##   sales performance promoted customer_rate
## 1   594           2        0          3.94
## 2   446           3        0          4.06
## 3   674           4        1          3.83
## 4   525           2        0          3.62
## 5   657           3        1          4.40
## 6   918           2        1          4.54
```

```
 #In the code so far we have used a variety of functions.
 #Exampe head(), subset(), rbind().

 #Functions exist to perform common useful operations.

 #Often there are a large number of arguments that a function can take,
```

```
  #but many are optional and not required to be specified by the user.
 #Example, the function head(), which displays the first rows of a dataframe
  #has only on required argument x: the name of the dataframe.

head(prom_custrate)
```

```
##   promoted customer_rate
## 1        0          3.94
## 2        0          4.06
## 3        1          3.83
## 4        0          3.62
## 5        1          4.40
## 6        1          4.54
```

```
head(full_df, n = 5)
```

```
##   sales performance promoted customer_rate
## 1   594           2        0          3.94
## 2   446           3        0          4.06
## 3   674           4        1          3.83
## 4   525           2        0          3.62
## 5   657           3        1          4.40
```

```
head(x = salespeople)
```

```
##   promoted sales customer_rate performance
## 1        0   594          3.94           2
## 2        0   446          4.06           3
## 3        1   674          3.83           4
## 4        0   525          3.62           2
## 5        1   657          4.40           3
## 6        1   918          4.54           2
```

```
head(x = salespeople, n=4)
```

```
##   promoted sales customer_rate performance
## 1        0   594          3.94           2
## 2        0   446          4.06           3
## 3        1   674          3.83           4
## 4        0   525          3.62           2
```

```
 #When running a function, you can either specify the arguments by name or
  #you can enter them in order without their names.

 #see fewer rows - arguments to be in the right order if not named.

head(salespeople, 3)
```

```
##   promoted sales customer_rate performance
## 1        0   594          3.94           2
## 2        0   446          4.06           3
## 3        1   674          3.83           4
```

```
head(salespeople, 6)
```

```
##   promoted sales customer_rate performance
## 1        0   594          3.94           2
```

```
## 2        0   446          4.06           3
## 3        1   674          3.83           4
## 4        0   525          3.62           2
## 5        1   657          4.40           3
## 6        1   918          4.54           2
```

```
 #or if you don't know the right order name your arguments
 #and you can put them in any order.

head(n=3, x = salespeople)
```

```
##   promoted sales customer_rate performance
## 1        0   594          3.94           2
## 2        0   446          4.06           3
## 3        1   674          3.83           4
```

```
head(n=6, x = salespeople)
```

```
##   promoted sales customer_rate performance
## 1        0   594          3.94           2
## 2        0   446          4.06           3
## 3        1   674          3.83           4
## 4        0   525          3.62           2
## 5        1   657          4.40           3
## 6        1   918          4.54           2
```

Functions, packages and libraries

```
 #Most functions in R have excellent help documentations
 #To get help on the head() function, type help(head) or ?head.

?head
help(head)

 #Alternatively you can open the Help browser window directly in RStudio
 #and do a search there.
```

Writing your own functions

```
 #Functions are not limited to those that come packaged in R.
 #Users can write their own functions to perform tasks
 #that are helpful to their objectives.

 #In this example, a simple function is written
 #which generates a report on a dataframe

df_report <- function(df){paste("This dataframe contains", nrow(df), "rows and", ncol(df), "columns.
                                There are", sum(is.na(df)), "NA entries.")}

 #We can test our function by using the built-in mtcars data set in R.

df_report(mtcars)
```

```
## [1] "This dataframe contains 32 rows and 11 columns. \n                                There are 0 NA
```

```
paste("This is how the paste", "functioning works")
```

```
## [1] "This is how the paste functioning works"
```

```
df_report(iris)
```

```
## [1] "This dataframe contains 150 rows and 5 columns. \n                                                    There are 0 NA
```

Installing packages

Using packages

```
 #Once you have installed a package into your package library,
 #to use it in your R session you need to load it using the library() function

library(MASS)
help(package = "MASS")

 #Once a package is loaded from your library,
 #you can use any of the functions inside it.

#MASS::stepAIC()
```

The pipe operator

```
url <- "https://raw.githubusercontent.com/msuiitdmsgabriel/datasets-regression/main/salespeople.csv"
salespeople <- read.csv(url)

 #The pipe operator makes code more natural to read and write and reduces
 #the typical computing problem of many nested operations inside parentheses.

sample(14,14)
```

```
##  [1] 10 14 13  5 12  8 11  2  7  4  9  3  1  6
```

```
sales <- subset(salespeople , subset = sales < 500)

 #take the mean of those values

mean(sales$sales)
```

```
## [1] 388.6684
```

```
mean(subset(salespeople$sales, subset = salespeople$sales < 500))
```

```
## [1] 388.6684
```

```
mean(subset(salespeople, subset = sales < 500)$sales)
```

```
## [1] 388.6684
```

```
 #load magrittr library to get the pipe operator

library(magrittr)
subset(salespeople$sales, subset = salespeople$sales < 500) %>%
mean()
```

```
## [1] 388.6684
```

```
round(mean(subset(salespeople, subset = sales < 500)$sales))
```

```
## [1] 389
```

Errors, warnings and messages

```
url <- "https://raw.githubusercontent.com/msuiitdmsgabriel/datasets-regression/main/salespeople.csv"
salespeople <- read.csv(url)

 #Errors are serious problems which usually result in the halting of your code
 #and a failure to return your requested output.

 #subset(salespeople, subset = sales = 720) - the equal sign should be double

subset(salespeople, subset = sales == 720)
```

```
##     promoted sales customer_rate performance
## 290        1   720          3.76           3
```

```
 #head[salespeople] - parentheses, not brackets

head(salespeople)
```

```
##   promoted sales customer_rate performance
## 1        0   594          3.94           2
## 2        0   446          4.06           3
## 3        1   674          3.83           4
## 4        0   525          3.62           2
## 5        1   657          4.40           3
## 6        1   918          4.54           2
```

```
 #Warnings are less serious and usually alert you to something that
 #you might be overlooking and which could indicate a problem with the output.
 #Messages are pieces of information that may
 #or may not be useful to you ata particular point in time.
```

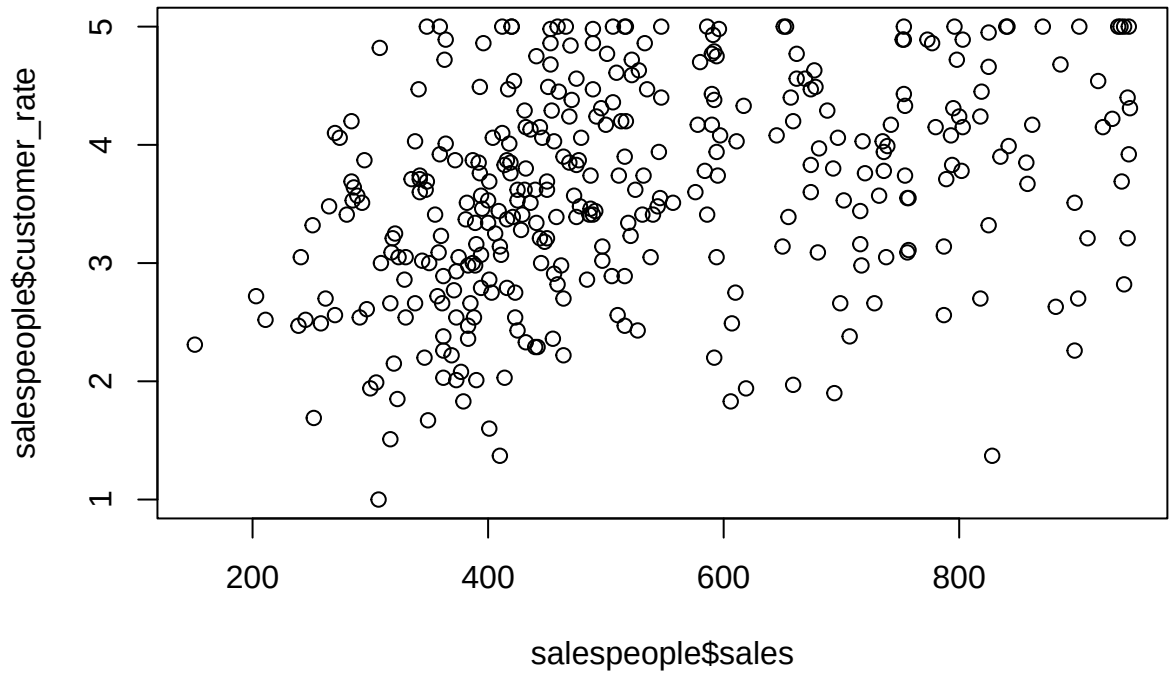
Plotting and graphing

```
 #there are numerous ways to plot and graph information in R.
 #graphical output will be rendered in the Plots pane,
 #where you can copy it  or save it as an image.
```

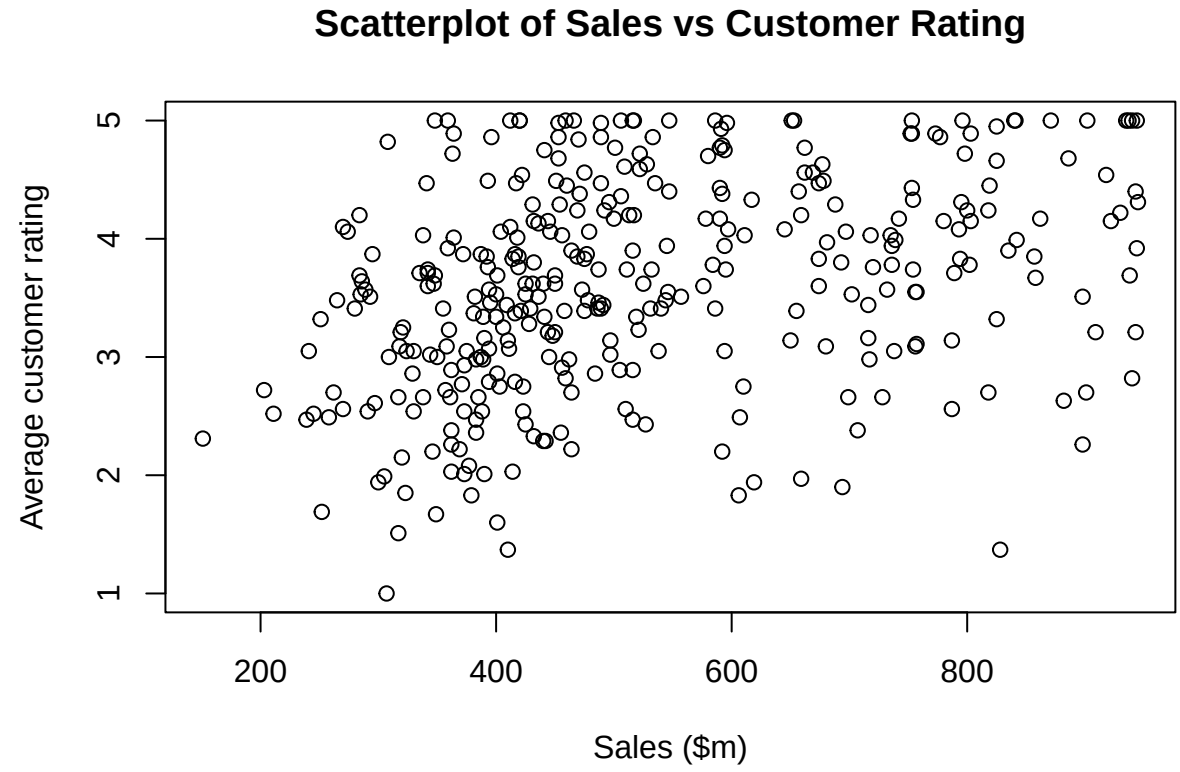
Plotting in base R

```
url <- "https://raw.githubusercontent.com/msuiitdmsgabriel/datasets-regression/main/salespeople.csv"
salespeople <- read.csv(url)

 #The simplest plot function in base R is plot(). This performs basic X-Y plotting.
 #Note the use of the arguments main, xlab and ylab
 #for customizing the axis labels and title for the plot.
```

```
plot(x = salespeople$sales, y = salespeople$customer_rate)
```

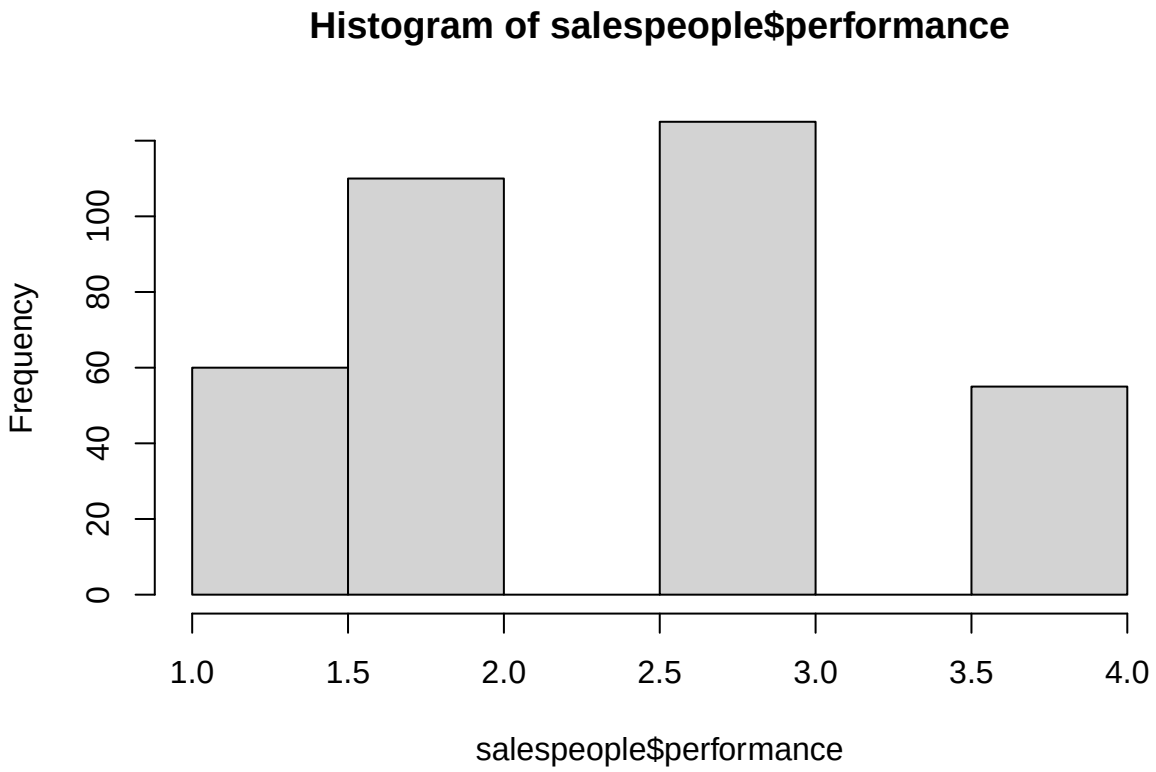



```
plot(x = salespeople$sales, y = salespeople$customer_rate,
     xlab = "Sales ($m)", ylab = "Average customer rating",
     main = "Scatterplot of Sales vs Customer Rating")
```

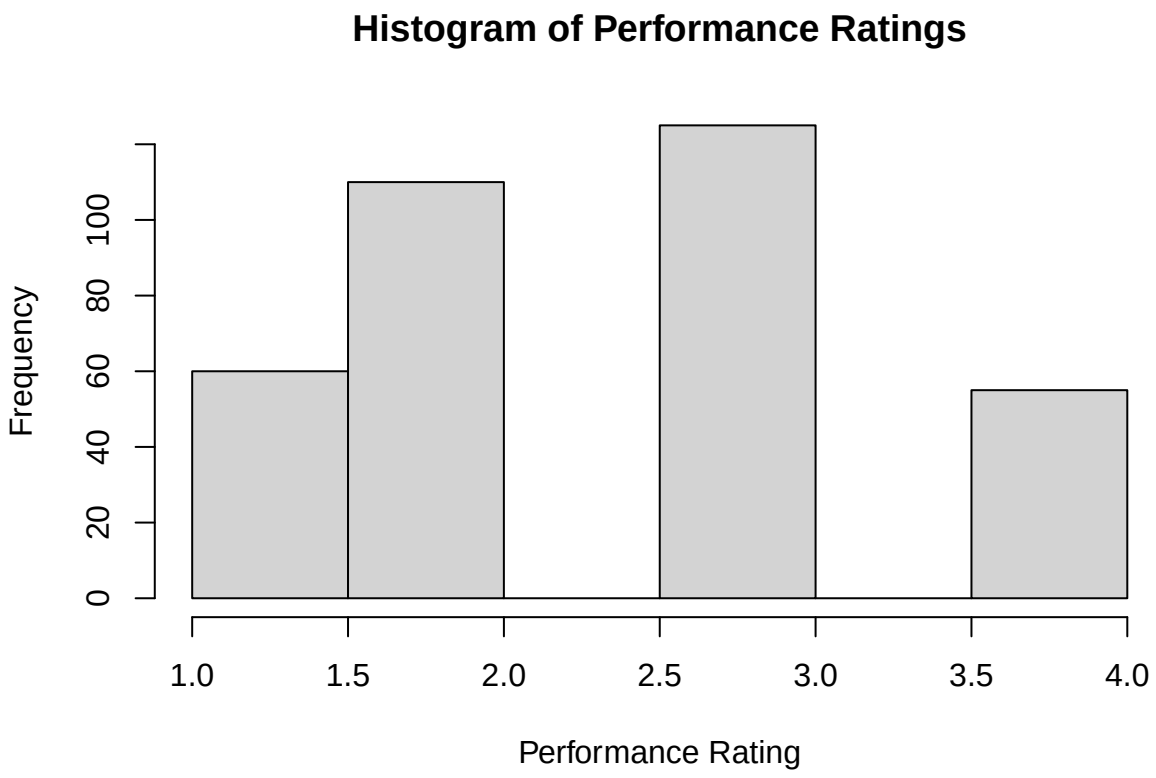



```
#Histograms of data can be generated using the hist() function.
```

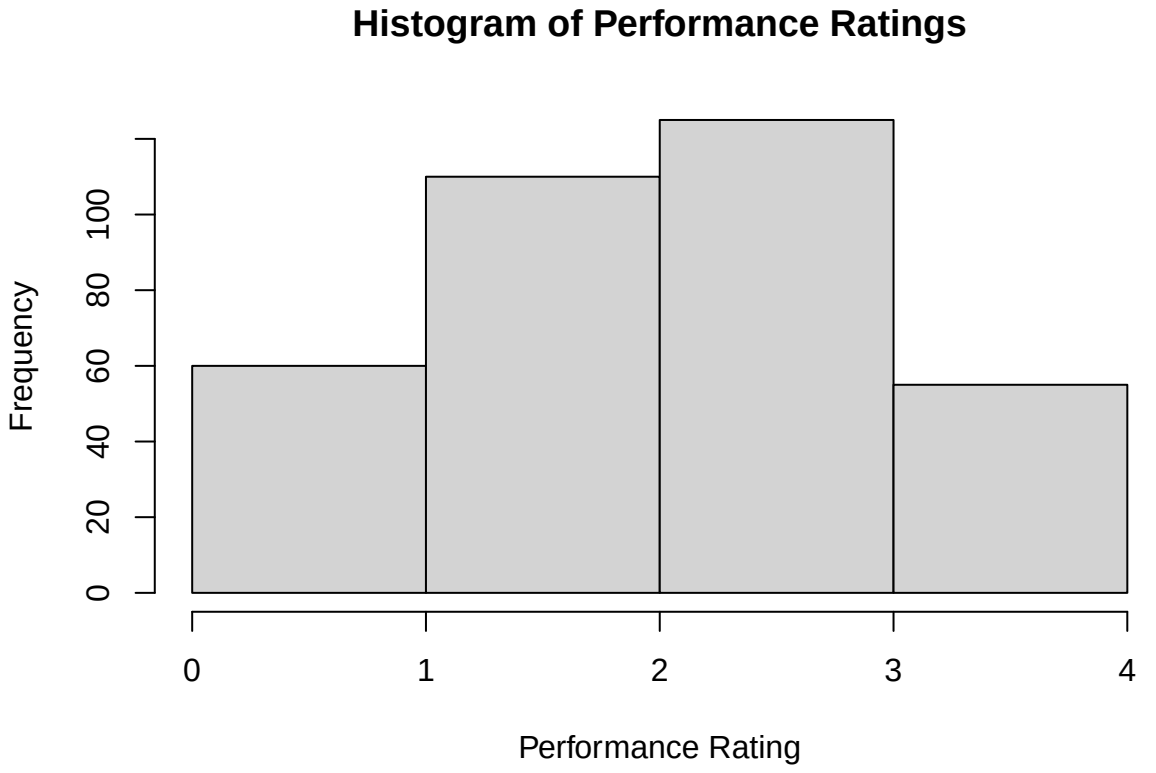
```
  #Note the use of breaks to customize how the bars appear.

hist(salespeople$performance)
```



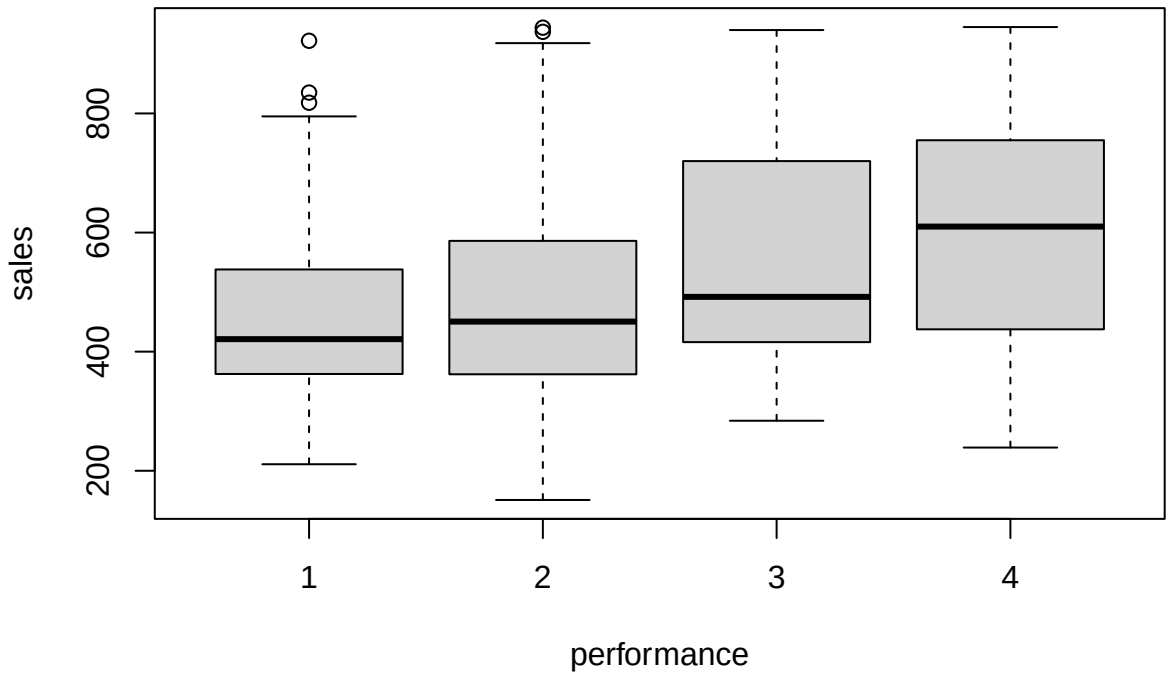

```
hist(salespeople$performance,
     xlab = "Performance Rating",
     main = "Histogram of Performance Ratings")
```



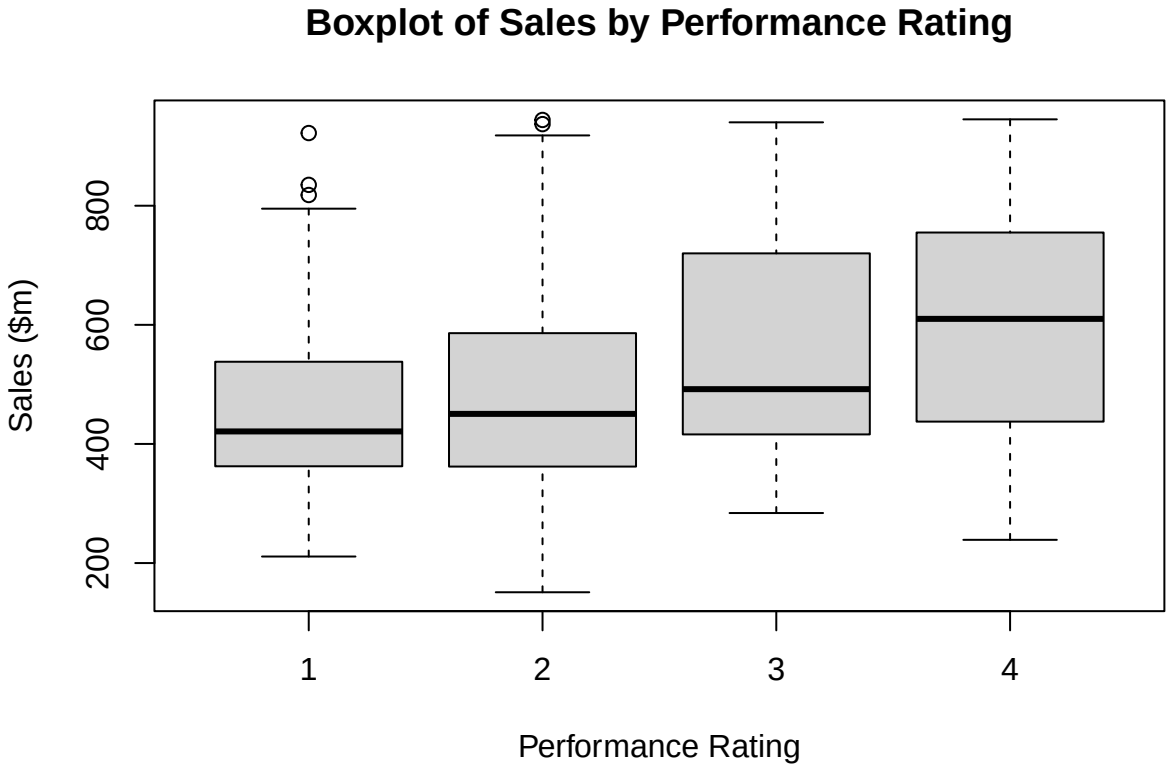

```
hist(salespeople$performance, breaks = 0:4,
     xlab = "Performance Rating",
     main = "Histogram of Performance Ratings")
```



```
#Box and whisker plots are excellent ways to see the distribution of a variable,
#and can be grouped against another variable to see bivariate       patterns.
#Note the use of the formula and data notation here to define the variable
#we are interested in and how we want it grouped.

boxplot(formula = sales ~ performance, data = salespeople)
```




```
boxplot(formula = sales ~ performance, data = salespeople,
        xlab = "Performance Rating", ylab = "Sales ($m)",
        main = "Boxplot of Sales by Performance Rating")
```



Specialist plotting and graphing packages

```
 #the most commonly used specialist plotting and graphing package in R is ggplot2.
 #ggplot2 allows the flexible construction of a very wide range of charts and  graphs,
 #but uses a very specific command grammar which can take some getting  used to.

#install.packages("ggplot2")
library(ggplot2)
 #The plotly package allows the use of the plotly graphing library in R.
 #This is an excellent package for interactive graphing
 #and is used for 3D illustrations in this book.

#install.packages("plotly")

 #GGally is a package that extends ggplot2 to allow easy combination of charts and graphs.
 #One of its most popular functions is ggpairs(), which produces a pairplot.

install.packages("GGally")
```

```
## Installing package into '/cloud/lib/x86_64-pc-linux-gnu-library/4.3'
## (as 'lib' is unspecified)
```

```
url <- "https://raw.githubusercontent.com/msuiitdmsgabriel/datasets-regression/main/salespeople.csv"
salespeople <- read.csv(url)
salespeople$promoted <- as.factor(salespeople$promoted)
salespeople$performance <- as.factor(salespeople$performance)
GGally::ggpairs(salespeople)
```

```
## Registered S3 method overwritten by 'GGally':
##   method from
```

```
##   +.gg   ggplot2
```

```
## Warning: Removed 1 rows containing non-finite values (`stat_boxplot()`).
## Removed 1 rows containing non-finite values (`stat_boxplot()`).
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```

```
## Warning: Removed 1 rows containing non-finite values (`stat_bin()`).
```

```
## Warning: Removed 1 rows containing non-finite values (`stat_density()`).
```

```
## Warning in ggally_statistic(data = data, mapping = mapping, na.rm = na.rm, :
## Removing 1 row that contained a missing value
```

```
## Warning: Removed 1 rows containing non-finite values (`stat_boxplot()`).
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```

```
## Warning: Removed 1 rows containing non-finite values (`stat_bin()`).
```

```
## Warning: Removed 1 rows containing missing values (`geom_point()`).
```

```
## Warning: Removed 1 rows containing non-finite values (`stat_density()`).
```

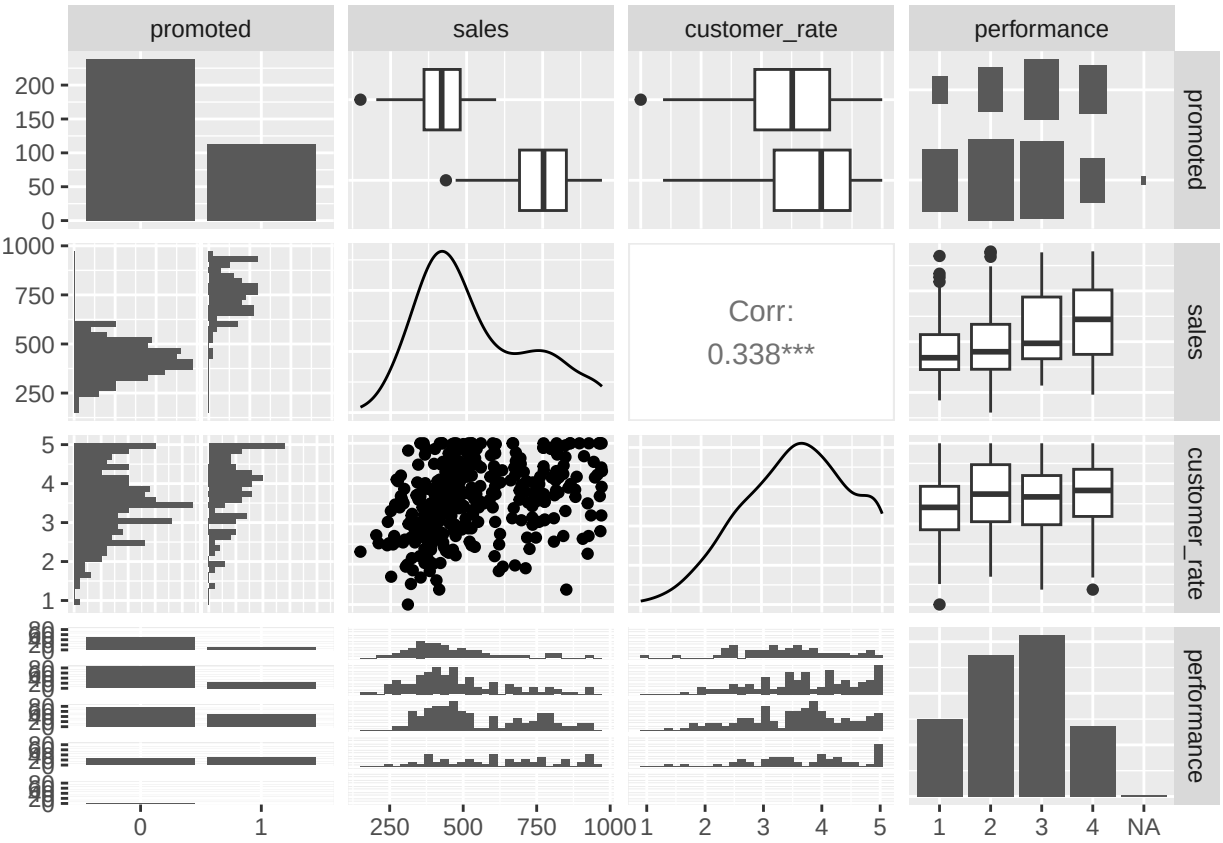
```
## Warning: Removed 1 rows containing non-finite values (`stat_boxplot()`).
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```

```
## Warning: Removed 1 rows containing non-finite values (`stat_bin()`).
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```

```
## Warning: Removed 1 rows containing non-finite values (`stat_bin()`).
```



References

- Venables, W. N., and B. D. Ripley. 2002. Modern Applied Statistics with s.Wickham, Hadley. 2016. ggplot2: Elegant Graphics for Data Analysis.
- Wickham, Hadley, and Garrett Grolemund. 2016. R for Data Science.
- Xie, Yihui, Christophe Dervieux, and Emily Riederer. 2020. R Markdown Cookbook.
- salespeople