Explore

In this note we will examine ways to explore a dataset, first using built-in functions, then extracting subsets of interest.

As an example, let us examine baseball team performance data. The dataset contains average salary paid in 2016 and win percentage in 2017. (Source: Lahman Baseball Database available at SeanLahman.com)

To get started, read the data contained in team_salary.csv. Since the dataset is .csv file, we can use `read.csv()` to import it into R. In addition to the filename, R has to be told where on the computer this file is located. This is illustrated below.

```
team_salary = read.csv(file = 'c:/users/vlala/downloads/team_salary.csv')
```

Another alternative is to hard code the location where files are kept into the current R session. `setwd()` will set the working directory, so when a file is read in or saved, it will by default reference the working directory. `setwd()` needs to be run only once in an R session. This approach is illustrated below.

```
setwd('c:/users/vlala/downloads')
team_salary = read.csv(file = 'team_salary.csv')
```

Most spreadsheet type formats of data resemble a data.frame or related formats like a tibble.

```
class(team_salary)
```

```
## [1] "data.frame"
```

Examine the first six rows of the dataset

```
head(team_salary)
```

```
##   Team_ID            Team_Name   win_pct Average_Salary
## 1     ARI Arizona Diamondbacks 0.5740741        3363041
## 2     ATL       Atlanta Braves 0.4444444        2362010
## 3     BAL    Baltimore Orioles 0.4629630        5581498
## 4     BOS       Boston Red Sox 0.5740741        6501578
## 5     CHA    Chicago White Sox 0.4135802        4519947
## 6     CHN         Chicago Cubs 0.5679012        5312678
```

. . . and the last five rows of the dataset

```
tail(team_salary,n = 5)
```

```
##    Team_ID            Team_Name   win_pct Average_Salary
## 26     SLN  St. Louis Cardinals 0.5123457        4614629
## 27     TBA       Tampa Bay Rays 0.4938272        2039190
## 28     TEX        Texas Rangers 0.4814815        6070301
## 29     TOR    Toronto Blue Jays 0.4691358        4782817
## 30     WAS Washington Nationals 0.5987654        5448179
```

Structure of the dataset provides basic information about number of rows and columns and class of variables.

```
str(team_salary)
```

```
## 'data.frame':    30 obs. of  4 variables:
##  $ Team_ID      : chr  "ARI" "ATL" "BAL" "BOS" ...
##  $ Team_Name    : chr  "Arizona Diamondbacks" "Atlanta Braves" "Baltimore Orioles" "Boston Red Sox"
```

```
##  $ win_pct       : num  0.574 0.444 0.463 0.574 0.414 ...
##  $ Average_Salary: num  3363041 2362010 5581498 6501578 4519947 ...
```

A few other handy functions are illustrated below.

```
nrow(team_salary)   # Number of rows
```

```
## [1] 30
```

```
ncol(team_salary)   # Number of columns
```

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

```
dim(team_salary)    # Number of rows and columns
```

```
## [1] 30  4
```

```
names(team_salary)  # Variable names
```

```
## [1] "Team_ID"        "Team_Name"      "win_pct"        "Average_Salary"
```

As noted earlier, packages can enhance the capabilities of Base R. In the example below, `datatable()` from library(DT) generates an interactive table.

```
library(DT)
datatable(team_salary)
```

Show 10 entries Search:

	Team_ID	Team_Name	win_pct	Average_Salary
1	ARI	Arizona Diamondbacks	0.574074074074074	3363040.88461538
2	ATL	Atlanta Braves	0.444444444444444	2362010.03448276
3	BAL	Baltimore Orioles	0.462962962962963	5581498.48275862
4	BOS	Boston Red Sox	0.574074074074074	6501577.96551724
5	CHA	Chicago White Sox	0.41358024691358	4519946.68
6	CHN	Chicago Cubs	0.567901234567901	5312678.20689655
7	CIN	Cincinnati Reds	0.419753086419753	3066898.5862069
8	CLE	Cleveland Indians	0.62962962962963	2752292.59259259
9	COL	Colorado Rockies	0.537037037037037	3413487
10	DET	Detroit Tigers	0.395061728395062	6286338.09677419

Showing 1 to 10 of 30 entries Previous 1 2 3 Next

Subset

A common task in data analysis is to extract a subset of the data. Elements in a data frame can be identified by their matrix location, using a general notation: [row number, column number]. For instance, to extract the element in row 4, column 5 of a data frame, df, use: df[4,5]. Let us examine a few examples with `team_salary`. Subset row 18, column 2

```
team_salary[18,2]
```

```
## [1] "New York Yankees"
```

Subset row 18, column 3

```
team_salary[18,3]
```

```
## [1] 0.5617284
```

Subset row 18, column 4

```
team_salary[18,4]
```

```
## [1] 7689579
```

Subset row 19, column 2

```
team_salary[19,2]
```

```
## [1] "New York Mets"
```

To formalize this, there a couple of ways of subsetting Integers: Positive returns specified elements, Negative returns everything but specified elements.

```
team_salary[18,-1]
```

```
##           Team_Name   win_pct Average_Salary
## 18 New York Yankees 0.5617284        7689579
```

Blank Spaces: Returns everything. E.g. df[2,] or df[,2]

```
team_salary[18,]
```

```
##    Team_ID        Team_Name   win_pct Average_Salary
## 18     NYA New York Yankees 0.5617284        7689579
```

```
team_salary[,1]
```

```
##  [1] "ARI" "ATL" "BAL" "BOS" "CHA" "CHN" "CIN" "CLE" "COL" "DET" "HOU" "KCA"
## [13] "LAA" "LAN" "MIA" "MIL" "MIN" "NYA" "NYN" "OAK" "PHI" "PIT" "SDN" "SEA"
## [25] "SFN" "SLN" "TBA" "TEX" "TOR" "WAS"
```

Names: Returns all elements in columns.

```
team_salary[,'Team_ID']
```

```
##  [1] "ARI" "ATL" "BAL" "BOS" "CHA" "CHN" "CIN" "CLE" "COL" "DET" "HOU" "KCA"
## [13] "LAA" "LAN" "MIA" "MIL" "MIN" "NYA" "NYN" "OAK" "PHI" "PIT" "SDN" "SEA"
## [25] "SFN" "SLN" "TBA" "TEX" "TOR" "WAS"
```

Dollar: Returns a column

```
team_salary$Team_ID
```

```
##  [1] "ARI" "ATL" "BAL" "BOS" "CHA" "CHN" "CIN" "CLE" "COL" "DET" "HOU" "KCA"
## [13] "LAA" "LAN" "MIA" "MIL" "MIN" "NYA" "NYN" "OAK" "PHI" "PIT" "SDN" "SEA"
## [25] "SFN" "SLN" "TBA" "TEX" "TOR" "WAS"
```

Logicals: Returns elements that correspond to TRUE. Here is a simple illustration.

```
team_salary[,c(T,F,F,F)]
```

```
##  [1] "ARI" "ATL" "BAL" "BOS" "CHA" "CHN" "CIN" "CLE" "COL" "DET" "HOU" "KCA"
## [13] "LAA" "LAN" "MIA" "MIL" "MIN" "NYA" "NYN" "OAK" "PHI" "PIT" "SDN" "SEA"
## [25] "SFN" "SLN" "TBA" "TEX" "TOR" "WAS"
```

Logicals are widely used for subsetting but seldom in the form illustrated above. It is most commonly used to select data that meets a certain condition. For e.g., names of all teams that had a win_pct greater than 0.5. To do this we pass a logical vector to a subsetting operation.

```
team_salary$win_pct>0.5
```

```
##  [1]  TRUE FALSE FALSE  TRUE FALSE  TRUE FALSE  TRUE  TRUE FALSE  TRUE FALSE
## [13] FALSE  TRUE FALSE  TRUE  TRUE  TRUE FALSE FALSE FALSE FALSE FALSE FALSE
## [25] FALSE  TRUE FALSE FALSE FALSE  TRUE
```

```
team_salary[team_salary$win_pct>0.5,'Team_Name']
```

```
##  [1] "Arizona Diamondbacks" "Boston Red Sox"       "Chicago Cubs"
##  [4] "Cleveland Indians"    "Colorado Rockies"     "Houston Astros"
##  [7] "Los Angeles Dodgers"  "Milwaukee Brewers"    "Minnesota Twins"
## [10] "New York Yankees"     "St. Louis Cardinals"  "Washington Nationals"
```

Similarly, here is a way to extract team names and win_pct for teams that pay more than $6,000,000.

```
team_salary[team_salary$Average_Salary>6000000, c('Team_Name','win_pct')]
```

```
##               Team_Name   win_pct
## 4        Boston Red Sox 0.5740741
## 10       Detroit Tigers 0.3950617
## 14  Los Angeles Dodgers 0.6419753
## 18     New York Yankees 0.5617284
## 25 San Francisco Giants 0.3950617
## 28        Texas Rangers 0.4814815
```

It is a good practice to share data in a meaningful order. So, let us order the above subset in ascending order by win_pct.

```
team_salary_subset = team_salary[team_salary$Average_Salary>6000000, c('Team_Name','win_pct')]
team_salary_subset[order(team_salary_subset$win_pct,decreasing = F),]
```

```
##               Team_Name   win_pct
## 10       Detroit Tigers 0.3950617
## 25 San Francisco Giants 0.3950617
## 28        Texas Rangers 0.4814815
## 18     New York Yankees 0.5617284
## 4        Boston Red Sox 0.5740741
## 14  Los Angeles Dodgers 0.6419753
```

dplyr functions

Over the last few years, dplyr has become a popular package for subsetting and transforming data. Here, we will examine two dplyr functions that are useful for performing subsetting operations.

```
library(dplyr)
select(team_salary,'Team_ID')
```

```
##    Team_ID
## 1      ARI
## 2      ATL
## 3      BAL
## 4      BOS
## 5      CHA
## 6      CHN
## 7      CIN
## 8      CLE
```

```
## 9      COL
## 10     DET
## 11     HOU
## 12     KCA
## 13     LAA
## 14     LAN
## 15     MIA
## 16     MIL
## 17     MIN
## 18     NYA
## 19     NYN
## 20     OAK
## 21     PHI
## 22     PIT
## 23     SDN
## 24     SEA
## 25     SFN
## 26     SLN
## 27     TBA
## 28     TEX
## 29     TOR
## 30     WAS
```

When conducting a series of computations using dplyr functions, it is best to use piping operations. To illustrate, the above operation can be rewritten using a piping operation.

```
team_salary %>%
  select(Team_ID)
```

```
##    Team_ID
## 1      ARI
## 2      ATL
## 3      BAL
## 4      BOS
## 5      CHA
## 6      CHN
## 7      CIN
## 8      CLE
## 9      COL
## 10     DET
## 11     HOU
## 12     KCA
## 13     LAA
## 14     LAN
## 15     MIA
## 16     MIL
## 17     MIN
## 18     NYA
## 19     NYN
## 20     OAK
## 21     PHI
## 22     PIT
## 23     SDN
## 24     SEA
## 25     SFN
```

```
## 26     SLN
## 27     TBA
## 28     TEX
## 29     TOR
## 30     WAS
```

Next, let us use dplyr functions and piping operations to extract team names and win_pct for teams that pay more than $6,000,000.

```
team_salary%>%
  filter(Average_Salary>6000000)%>%
  select(Team_Name,win_pct)
```

```
##              Team_Name   win_pct
## 1       Boston Red Sox 0.5740741
## 2       Detroit Tigers 0.3950617
## 3  Los Angeles Dodgers 0.6419753
## 4     New York Yankees 0.5617284
## 5 San Francisco Giants 0.3950617
## 6        Texas Rangers 0.4814815
```

Finally, to sort the above data in ascending order, use `arrange()`

```
team_salary%>%
  filter(Average_Salary>6000000)%>%
  select(Team_Name,win_pct)%>%
  arrange(win_pct)
```

```
##              Team_Name   win_pct
## 1       Detroit Tigers 0.3950617
## 2 San Francisco Giants 0.3950617
## 3        Texas Rangers 0.4814815
## 4     New York Yankees 0.5617284
## 5       Boston Red Sox 0.5740741
## 6  Los Angeles Dodgers 0.6419753
```