Data visualizations and know-hows

<Kaitlyn Watson Group 9>

M3 ICA3

The data

For this section we will use the `nfl` data. We will use the `tidyverse` and `ggiraph` packages. You will need to first install `ggiraph` with `install.packages("ggiraph")`.

```
library(tidyverse)
library(ggplot2)
library(ggiraph)
```

Do any interesting relationships exists between NFL combine metrics and draft position?

Not all players have a draft position and not all NFL players have stats in the combine.

Below is data on every player who participated in the 2018 NFL draft combine or was drafted by an NFL team in 2018. Let's read in the data and set it as a tibble to make it easier to view. The argument `stringsAsFactors = FALSE` ensures variables such as `Player`, `Pos`, `School`, etc are read in as type character.

```
# A tibble: 336 x 14
   Player     Pos   School       Ht    Wt Dash40 Vertical Bench Broad.Jump Cone3
   <chr>      <chr> <chr>     <dbl> <int>  <dbl>    <dbl> <int>      <int> <dbl>
 1 Dante Pet~ WR    Washingt~    73   186  NA        NA      NA         NA NA
 2 Kemoko Tu~ EDGE  Rutgers      77   253   4.65     NA      NA         NA NA
 3 Josh Adams RB    Notre Da~    74   213  NA        NA      18         NA NA
 4 Ola Adeni~ EDGE  Toledo       74   248   4.83     31.5    26         NA  7.21
 5 Jordan Ak~ TE    Central ~    75   249  NA        NA      NA         NA NA
 6 Jaire Ale~ CB    Louisvil~    71   192   4.38     35      14        127  6.71
 7 Austin Al~ QB    Arkansas     73   210   4.81     29.5    NA        112  7.18
 8 Brian All~ C     Michigan~    73   298   5.34     26.5    27         99  7.81
 9 Josh Allen QB    Wyoming      77   237   4.75     33.5    NA        119  6.9
10 Marcus Al~ S     Penn St.     74   202  NA        37      15        127 NA
# ... with 326 more rows, and 4 more variables: Shuttle <dbl>, Team <chr>,
#   Round <int>, Pick <int>
```

- **Player**: player's name
- **Pos**: player's position
- **School**: college of player
- **Ht**: height in inches
- **Wt**: weight in pounds
- **Dash40**: forty yard dash time in seconds
- **Vertical**: vertical jump in inches
- **Bench**: number of bench press repititions at 225lbs

- **Broad.Jump**: broad jump distance in inches
- **Cone3**: 3 cone drill time in seconds
- **Shuttle**: twenty yard shuttle time in seconds
- **Team**: team that drafted the player
- **Round**: round the player was drafted (0 means no round)
- **Pick**: draft selection (0 means not drafted)

Investigation

As you can see above, `nfl` contains a lot of missing values - represented by `NA`. Not all players participate in the NFL combine, and some who do participate do not perform each skills test. Also, undrafted players have `NA` values for their `Team`. Before we create visualizations let's try to better understand some of the data.

Exercises

1. How many players went undrafted in the data set `nfl`?

```
undrafted<-nfl %>%
  filter(Pick==0)
count(undrafted)
```

```
# A tibble: 1 x 1
      n
  <int>
1   184
```

2. Which player had the fastest 40 yard dash time? What was his time?

```
nfl %>%
  select(Player, Dash40) %>%
  arrange(Dash40, na.rm=TRUE)
```

```
# A tibble: 336 x 2
   Player                  Dash40
   <chr>                    <dbl>
 1 Donte Jackson             4.32
 2 Parry Nickerson           4.32
 3 Denzel Ward               4.32
 4 Troy Apke                 4.34
 5 D.J. Chark                4.34
 6 Tony Brown                4.35
 7 Anthony Averett           4.36
 8 Marquez Valdes-Scantling  4.37
 9 Jaire Alexander           4.38
10 Shaquem Griffin           4.38
# ... with 326 more rows
```

Denzel Ward, Donte Jackson, Parry Nickerson have 4.32 as the fastest time.

3. How many players are in the data set for each position? *Hint*: `table()`

```
nfl %>%
  arrange((Pos)) %>%
  count(Pos)
```

```
# A tibble: 21 x 2
   Pos       n
   <chr> <int>
 1 C         9
 2 CB       41
 3 DB        2
 4 DE       16
 5 DT       24
 6 EDGE     23
 7 FB        1
 8 ILB      15
 9 K         4
10 LB        3
# ... with 11 more rows
```

4. What was the mean number of bench press repititions for all players listed at the position of `DT`?

```
nfl %>%
  filter(Pos=="DT") %>%
  summarise(mean(Bench, na.rm=TRUE))
```

```
# A tibble: 1 x 1
  `mean(Bench, na.rm = TRUE)`
                        <dbl>
1                        28.8
```

5. Which team drafted the most players in 2018?

```
nfl %>%
  count((Team)) %>%
  arrange(desc(n))
```

```
# A tibble: 33 x 2
   `(Team)`                    n
   <chr>                   <int>
 1  <NA>                    184
 2 "Denver Broncos "          9
 3 "Cincinnati Bengals "      8
 4 "Baltimore Ravens "        7
 5 "Indianapolis Colts "      7
 6 "Carolina Panthers "       6
 7 "Cleveland Browns "        6
 8 "Los Angeles Chargers "    6
 9 "Los Angeles Rams "        6
10 "New England Patriots "    6
# ... with 23 more rows
```

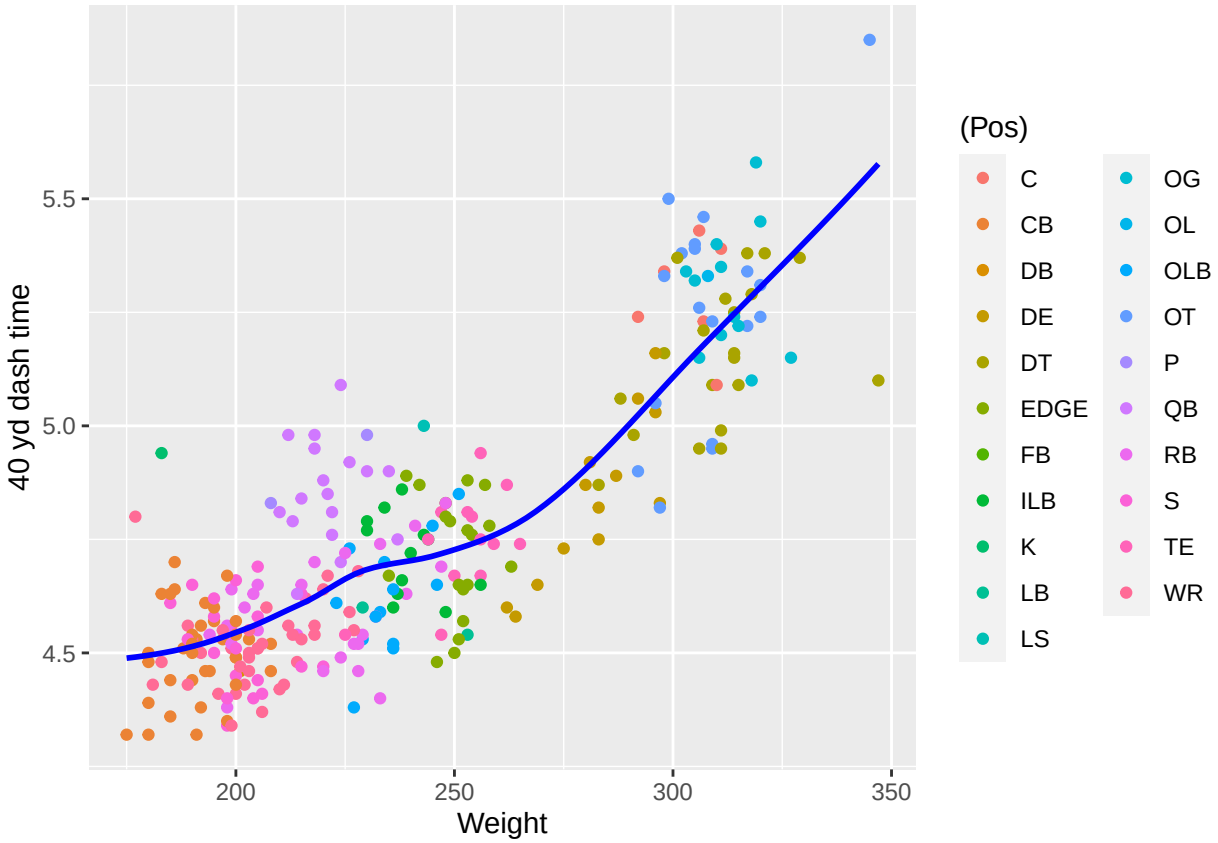
Denver Broncos drafted 9 players

Visualizations with ggplot

Exercises

Recreate plots 1 - 5. Add comments to the code that generated plots 6 and 7 to explain what is being done in those plots. Use the available hints before looking at the solution. Comment on any interesting trends/relationships you observe.

Plot 1

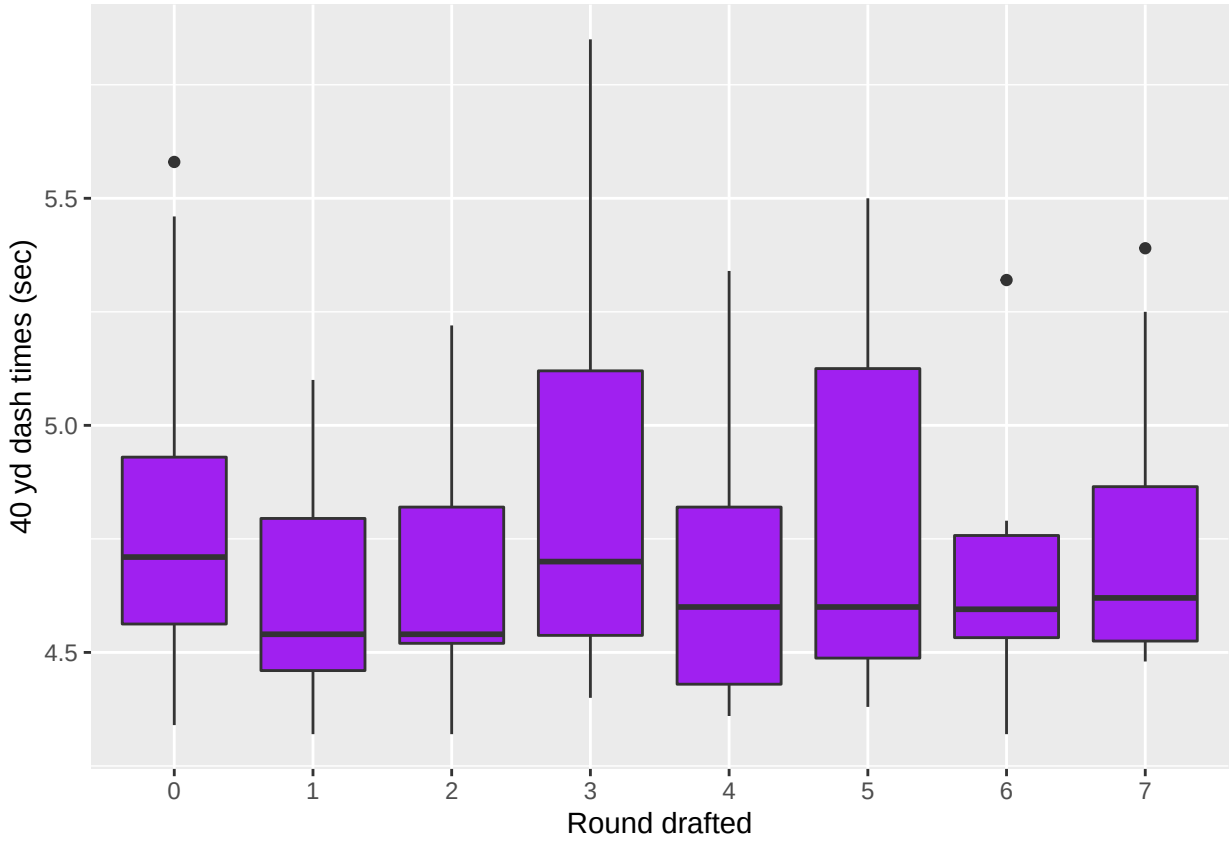


Plot

Hints

- `geom point()`
- `geom_smooth()`

Plot 2

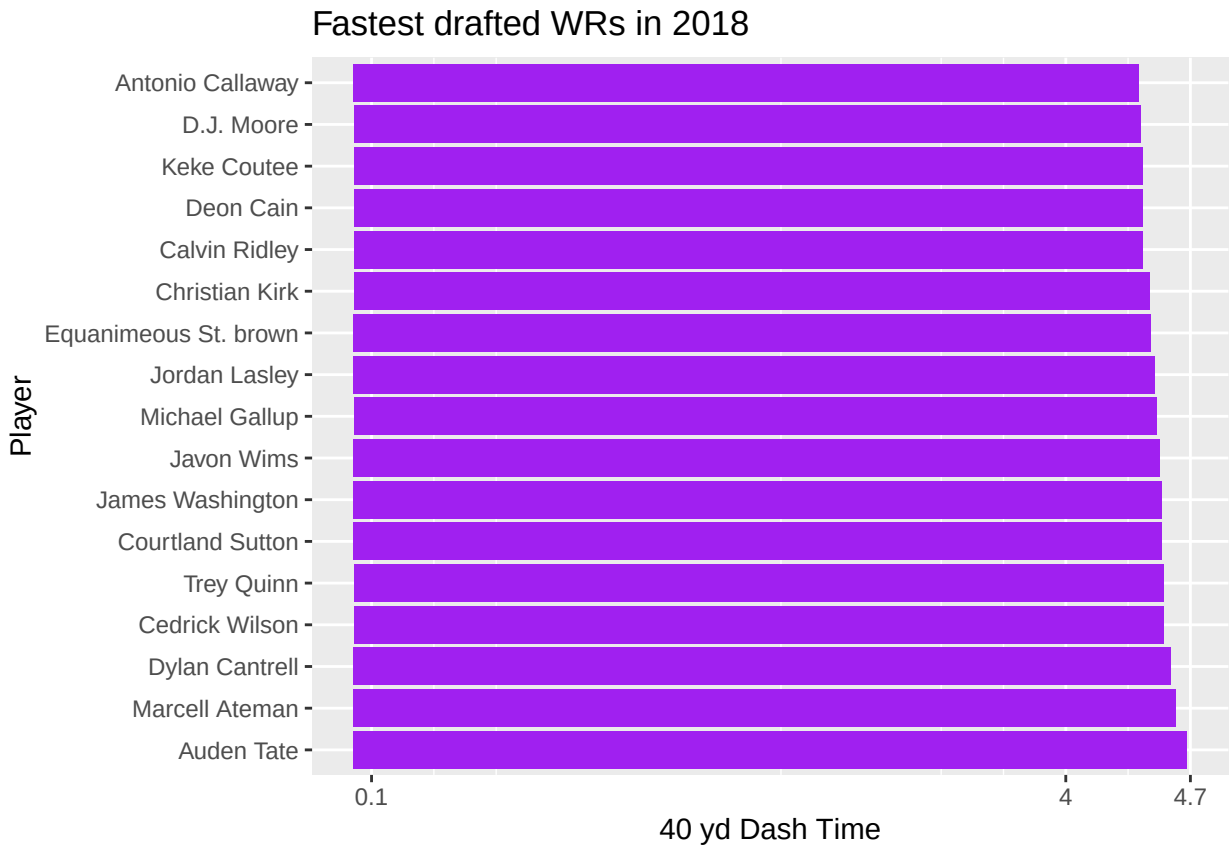


Plot

Hints

- `geom_boxplot`
- colors used: purple, black
- `caption` argument in `labs()`

Plot 3

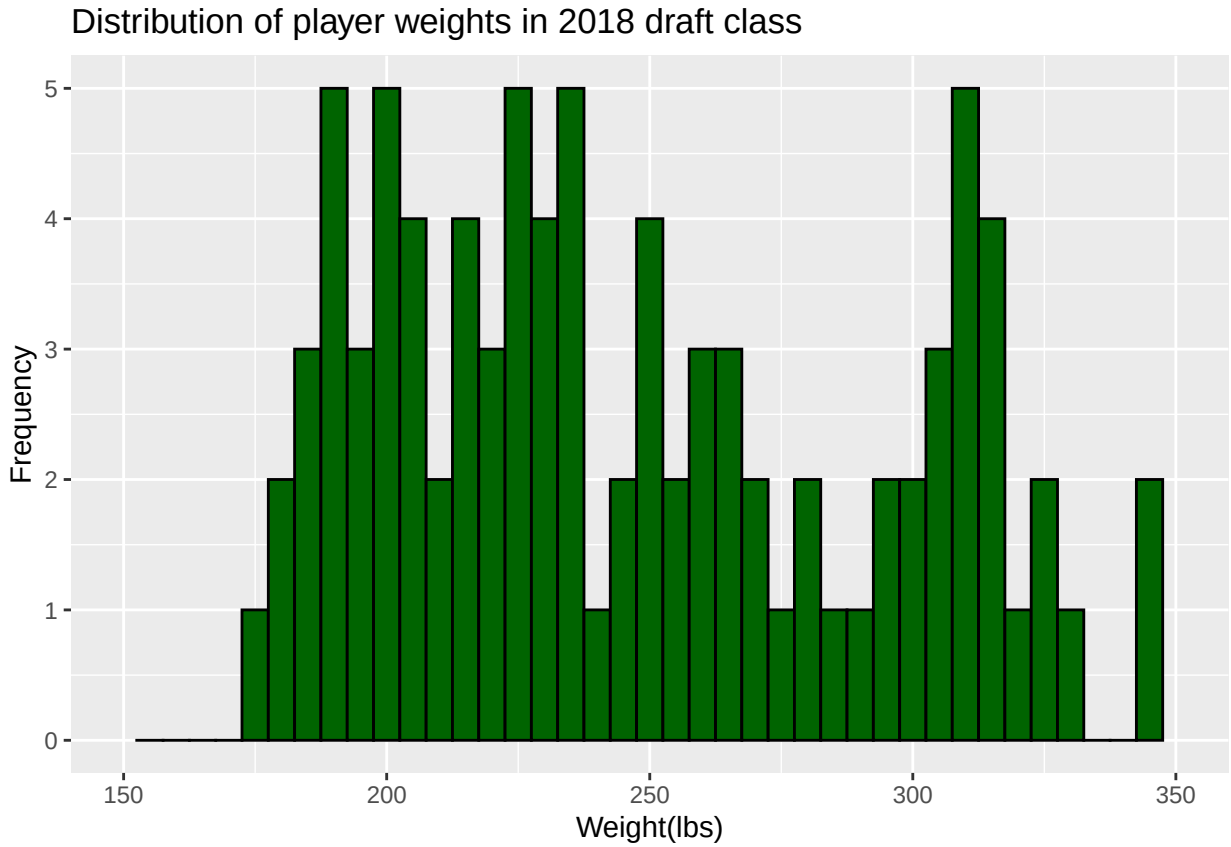


Plot

Hints

- subset `nfl` to create a data frame with only WR who were drafted
- colors used: purple, black
- `stat = "identity"` in `geom_bar()`
- flip coordinates
- `scale_y_continuous(breaks = seq(0, 1, .1), labels = seq((4, 5), .1))`
- to sort the bars use `reorder(Player, -Dash40)`

Plot 4



Plot

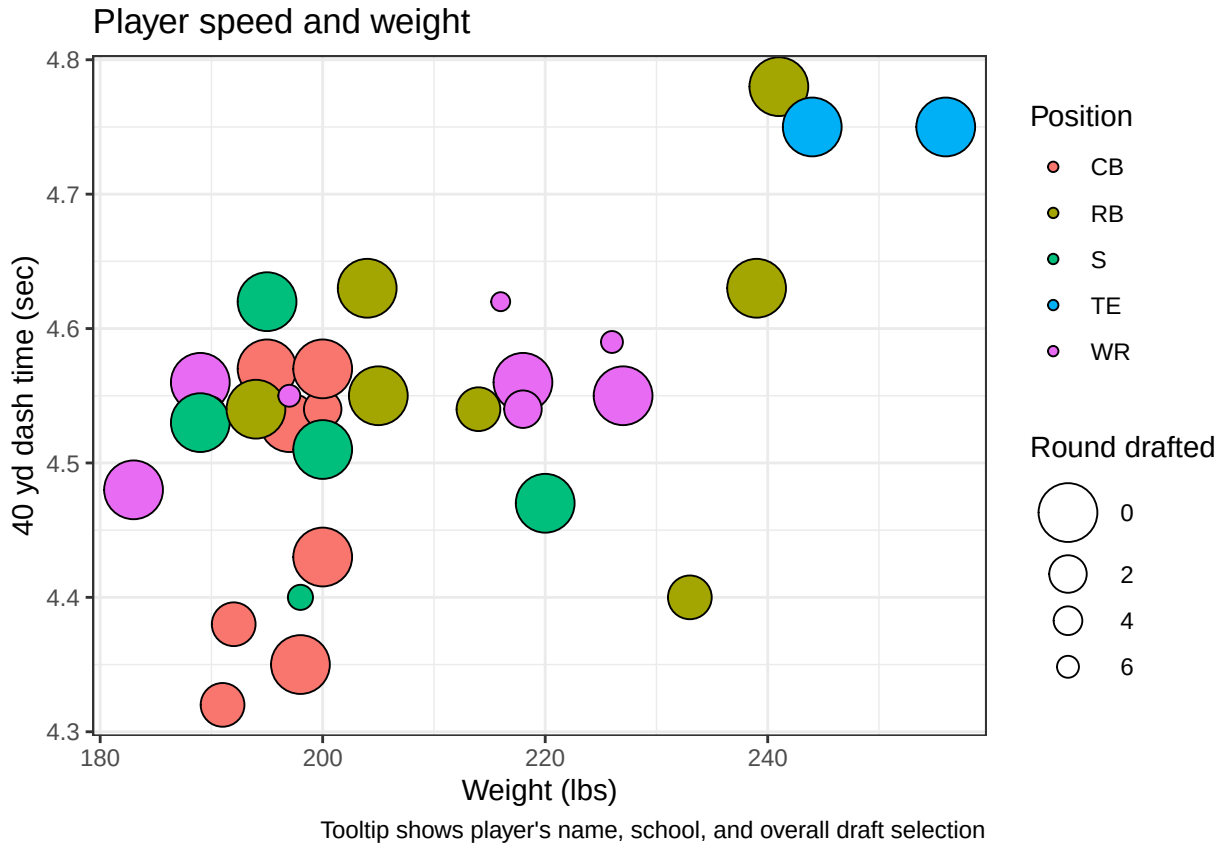
Hints

- `geom_histogram`
- colors used: darkgreen, black
- `binwidth = 5`

Plot 5

Plot

```
# A tibble: 35 x 14
   Player     Pos   School       Ht    Wt Dash40 Vertical Bench Broad.Jump Cone3
   <chr>      <chr> <chr>     <dbl> <int>  <dbl>    <dbl> <int>      <int> <dbl>
 1 Jaire Ale~ CB    Louisvil~    71   192   4.38     35      14        127  6.71
 2 Marcell A~ WR    Oklahoma~    76   216   4.62     34      13        121  7.07
 3 Saquon Ba~ RB    Penn St.     73   233   4.4      41      29         NA NA
 4 Tony Brown CB    Alabama      73   198   4.35     31.5    14        126  6.78
 5 Deontay B~ WR    USC          73   186  NA        NA      NA         NA NA
 6 Dylan Can~ WR    Texas Te~    75   226   4.59     38.5    18        130  6.56
 7 Chase Edm~ RB    Fordham      69   205   4.55     34      19        122  6.79
 8 Terrell E~ S     Virginia~    74   220   4.47     41.5    NA        134 NA
 9 Donnie Er~ RB    Western ~    75   241   4.78     31      15        114  7.09
10 Brandon F~ CB    Virginia~    74   197   4.53     NA      16         NA NA
# ... with 25 more rows, and 4 more variables: Shuttle <dbl>, Team <chr>,
#   Round <int>, Pick <int>
```



Hints

- subset `nfl` to only include the positions of CB, S, RB, WR, and TE for players that were drafted
- color used: black
- `shape = 21`
- `scale_size(range=c(10, 3))`
- `theme_bw()`

Plot 6

Plot The code allows for an interactive graph which means that by either hovering or clicking on it, additional information will be give. When looking at the solution, I saw that each time you hovered over a bubble on the map, there was information given in regard to the player name, what team they were from, and what pick they were. Looking at the trends of this graph, it is clear that many of the players who were lower in weight had much faster dash times. Additionally,the tight end (TE) players tended to be higher and weight and further up the draft pick. The corner backs seemed to have the earliest draft selection and lower in weight.

Plot 7

Plot Plot 7 is relatively the same information in regards to Plot 6. However, when hovering over one of the circles on the graph, all other circles of the same size highlight black as well. This is representing all the players who got picked in the same draft round. Looking up at the code above, it seems that by adding

girafe_options to the code allows the option to hover and fill black. Additionally, there are several points within the code where the draft is overridden.

Data know hows

In this section, we will get some practice reading in data sets and working with the apply function.

Read data

Data may be

1. available in base R or through an R package such as the `diamonds` data set that is available through `tidyverse`;
2. read in to R from a file on your computer;
3. read in to R directly from a website;
4. scraped from a website.

Today we will get practice with examples that involve 2 and 3. Some of the most popular R functions to accomplish this are

- `read.table()`
- `read.csv()`
- `load()`

Exercises

Read in the following data sets and save them as a well-named R object. A preview of each data set is given below for you to check your answer. Make sure all variable types are the same, headers are available when applicable, and `NA` values appear where appropriate.

1. `nevada_casino_sqft.csv` (available on D2L)

```
Nevada_Casino<-read.csv("nevada_casino_sqft.csv", na.strings = "")
head(Nevada_Casino)
```

```
  COUNTY AREA                                    NAME PITGAMES SLOTS KENO
1     14    0 ALAMO CASINO - MILL CITY                    0  3500    0
2      4    0 ALAMO CASINO AT WELLS PETRO                 0  2250    0
3     16    2 ALAMO TRAVEL CENTER                       900  6100    0
4      2    0 ALBERTSON'S #6046                           0   400    0
5      2    3 ALIANTE CASINO + HOTEL                   5060 98007    0
6      2    4 AQUARIUS CASINO RESORT                   8215 42075 1680
  BINGO SPORTS POKER  TOTAL
1     0      0     0   3500
2    NA      0     0     NA
3     0      0   150   7150
4     0      0     0    400
5  5624  14200  2109 125000
6     0   5100     0  57070
```

```
  COUNTY AREA                                    NAME PITGAMES SLOTS KENO
1     14    0 ALAMO CASINO - MILL CITY                0  3500    0
2      4    0 ALAMO CASINO AT WELLS PETRO             0  2250    0
3     16    2 ALAMO TRAVEL CENTER                   900  6100    0
4      2    0 ALBERTSON'S #6046                       0   400    0
5      2    3 ALIANTE CASINO + HOTEL               5060 98007    0
6      2    4 AQUARIUS CASINO RESORT               8215 42075 1680
  BINGO SPORTS POKER  TOTAL
1     0      0     0   3500
2    NA      0     0     NA
3     0      0   150   7150
4     0      0     0    400
5  5624  14200  2109 125000
6     0   5100     0  57070
```

2. `nevada_casino_sqft.csv` (available at: http://www.stat.ufl.edu/~winner/data/nevada_casino_sqft.csv)

```
Nevada_Casino2<-read.csv("http://www.stat.ufl.edu/~winner/data/nevada_casino_sqft.csv", stringsAsFactors
head(Nevada_Casino2)
```

```
  COUNTY AREA                                    NAME PITGAMES SLOTS KENO
1     14    0 ALAMO CASINO - MILL CITY                0  3500    0
2      4    0 ALAMO CASINO AT WELLS PETRO             0  2250    0
3     16    2 ALAMO TRAVEL CENTER                   900  6100    0
4      2    0 ALBERTSON'S #6046                       0   400    0
5      2    3 ALIANTE CASINO + HOTEL               5060 98007    0
6      2    4 AQUARIUS CASINO RESORT               8215 42075 1680
  BINGO SPORTS POKER  TOTAL
1     0      0     0   3500
2     0      0     0   2250
3     0      0   150   7150
4     0      0     0    400
5  5624  14200  2109 125000
6     0   5100     0  57070
```

```
  COUNTY AREA                                    NAME PITGAMES SLOTS KENO
1     14    0 ALAMO CASINO - MILL CITY                0  3500    0
2      4    0 ALAMO CASINO AT WELLS PETRO             0  2250    0
3     16    2 ALAMO TRAVEL CENTER                   900  6100    0
4      2    0 ALBERTSON'S #6046                       0   400    0
5      2    3 ALIANTE CASINO + HOTEL               5060 98007    0
6      2    4 AQUARIUS CASINO RESORT               8215 42075 1680
  BINGO SPORTS POKER  TOTAL
1     0      0     0   3500
2     0      0     0   2250
3     0      0   150   7150
4     0      0     0    400
5  5624  14200  2109 125000
6     0   5100     0  57070
```

3. `qqq.tsv` (available on D2L)

```
QQQ<-read.csv("qqq.tsv", header=TRUE, sep="\t")
head(QQQ)
```

```
  QQQ.Open QQQ.High QQQ.Low QQQ.Close QQQ.Volume QQQ.Adjusted
1    43.46    44.06   42.52     43.24  167689500     38.79258
2    43.30    44.21   43.15     44.06  136853500     39.52824
3    43.95    43.95   43.48     43.85  138958800     39.33984
4    43.89    44.12   43.64     43.88  106401600     39.36676
5    44.01    44.29   43.63     44.10  121577500     39.56412
6    43.96    44.66   43.82     44.62  121070100     40.03062
```

```
  QQQ.Open QQQ.High QQQ.Low QQQ.Close QQQ.Volume QQQ.Adjusted
1    43.46    44.06   42.52     43.24  167689500     38.79258
2    43.30    44.21   43.15     44.06  136853500     39.52824
3    43.95    43.95   43.48     43.85  138958800     39.33984
4    43.89    44.12   43.64     43.88  106401600     39.36676
5    44.01    44.29   43.63     44.10  121577500     39.56412
6    43.96    44.66   43.82     44.62  121070100     40.03062
```

4. `spy.txt` (available on D2L)

```
Spy<-read.table("spy.txt", header = FALSE, sep = "@")
head(Spy)
```

```
  V1     V2     V3     V4     V5       V6       V7
1  1 142.25 142.86 140.57 141.37 94807600 110.5206
2  2 141.23 142.05 140.61 141.67 69620600 110.7551
3  3 141.33 141.40 140.38 140.54 76645300 109.8717
4  4 140.82 141.41 140.25 141.19 71655000 110.3799
5  5 141.31 141.60 140.40 141.07 75680100 110.2861
6  6 140.58 141.57 140.30 141.54 72428000 110.6535
```

```
  V1     V2     V3     V4     V5       V6       V7
1  1 142.25 142.86 140.57 141.37 94807600 110.5206
2  2 141.23 142.05 140.61 141.67 69620600 110.7551
3  3 141.33 141.40 140.38 140.54 76645300 109.8717
4  4 140.82 141.41 140.25 141.19 71655000 110.3799
5  5 141.31 141.60 140.40 141.07 75680100 110.2861
6  6 140.58 141.57 140.30 141.54 72428000 110.6535
```

Apply

Matrix examples

The examples below can also be applied to data frames

```
# create a 3 x 3 matrix
my.matrix <- matrix(data = c(3, 5, 10, -1, 0, 2, 18, 5, -3),
                    nrow = 3, ncol = 3)
```

```
# matrix is filled column-wise, view matrix
my.matrix

# by columns
apply(X = my.matrix, MARGIN = 2, FUN = mean)
apply(X = my.matrix, MARGIN = 2, FUN = max)

# by rows
apply(X = my.matrix, MARGIN = 1, FUN = sd)
apply(X = my.matrix, MARGIN = 1, FUN = which.max)
apply(X = my.matrix, MARGIN = 1, FUN = sort)
```

1. Create a 4 x 3 matrix and the have the data filled in by rows.

```
# create a 4 x 3 matrix
Kaitlyn_matrix <- matrix(data = c(1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12),
                         nrow = 4, ncol = 3)
Kaitlyn_matrix
```

```
     [,1] [,2] [,3]
[1,]    1    5    9
[2,]    2    6   10
[3,]    3    7   11
[4,]    4    8   12
```

```
apply(X = Kaitlyn_matrix, MARGIN = 1, FUN = which.max)
```

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

2. Explain what the which.max function is doing.

It is telling us the location of the very first maximum number with is in the column 3

Array examples

```
# create a 2 x 2 x 3 array that contains the numbers 1 - 12
my.array <- array(data = c(1:12), dim = c(2,2,3))

# view the array
my.array

# apply sum over 1 dimension
apply(my.array, 1, sum)
apply(my.array, 2, sum)
apply(my.array, 3, sum)

# apply sum over multiple dimensions
apply(my.array, c(1,2), sum)
apply(my.array, c(1,3), sum)
```

```
apply(my.array, c(2,3), sum)
apply(my.array, c(3,1), sum)
apply(my.array, c(3,2), sum)
```

Further details

A summary of the `a,l,s,t apply` functions

Command	Description
`apply(X, MARGIN, FUN, ...)`	Obtain a vector/array/list by applying `FUN` along the specified `MARGIN` of an array or matrix `X`
`lapply(X, FUN, ...)`	Obtain a list by applying `FUN` to the elements of a list `X`
`sapply(X, FUN, ...)`	Simplified version of `lapply`. Returns a vector/array instead of list.
`tapply(X, INDEX, FUN, ...)`	Obtain a table by applying `FUN` to each combination of the factors given in `INDEX`

- The above functions are good alternatives to loops
- They are typically more efficient than loops (often run considerably faster on large data sets)
- They take practice to get used to, but make analysis easier to debug and less prone to error when used effectively
- Look at the Help's examples

References

1. https://www.pro-football-reference.com/draft/2018-combine.htm
2. http://r-statistics.co/Top50-Ggplot2-Visualizations-MasterList-R-Code.htm
3. http://www.ggplot2-exts.org/ggiraph.html
4. https://davidgohel.github.io/ggiraph/articles/offcran/using_ggiraph.html