Athletics Fantasy Team

Bryan Sam

July 15th, 2021

1. Athletics needs a new breed of scouts and managers

Athletics goes back to the original Olympics. Since then, little has changed. Athletes compete as individuals, seeking to throw the farthest, jump the farthest (or highest) and run the fastest. But people like cheering for teams, waving banners and yelling like mad during matches, wearing their favorite player's jerseys and staying loyal to their side through thick and thin.

What if athletics was a team sport? It could potentially be more interesting and would give us a new set of sports analytics to discuss. We might even reduce the incentives to do unsavory things in the pursuit of *altius*, *fortius*, and *citius*.

This dataset contains results from American athletes in the horizontal jumps (triple jump and long jump) and throws (shot put, discus, javelin, hammer and weight). Let's read, analyze, and scout **women's javelin**.

```
# Load the tidyverse package
library(tidyverse)

# Import the full dataset
data <- read_csv('athletics.csv')

# Select the results of interest: women's javelin
```

```
javelin <- data %>%
            filter(Event == "Javelin",
                  Male_Female == "Female") %>%
            select(-c("Male_Female", "Event"))

# Review data
head(javelin)
```

```
## # A tibble: 6 x 8
##   EventID Athlete            Flight1 Flight2 Flight3 Flight4 Flight5 Flight6
##     <dbl> <chr>                <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>
## 1       8 Brittany Borman       54.0    51.2    57.3    52.6    57.0    60.9
## 2       8 Ariana Ince           49.0    54.8    53.6    55.1    55.3    56.7
## 3       8 Kara Patterson        50.1    52.1     0      50.8    55.9    54.6
## 4       8 Kimberley Hamilton    48.0     0      50.9    54.1    55.2    53.3
## 5       8 Laura Loht            44.4    53.8    50.6    54.2     0      49.0
## 6       8 Brianna Bain          49.3     0      51.3     0      48.6    53.0
```

```
summary(javelin)
```

```
##     EventID         Athlete             Flight1         Flight2
##  Min.   :   8.0   Length:178         Min.   : 0.00   Min.   : 0.00
##  1st Qu.: 178.0   Class :character   1st Qu.:41.53   1st Qu.:40.23
##  Median : 511.0   Mode  :character   Median :48.85   Median :48.85
##  Mean   : 796.8                      Mean   :40.80   Mean   :39.87
##  3rd Qu.:1703.0                      3rd Qu.:53.20   3rd Qu.:53.07
##  Max.   :1859.0                      Max.   :64.94   Max.   :61.38
##     Flight3         Flight4         Flight5         Flight6
##  Min.   : 0.00   Min.   : 0.00   Min.   : 0.00   Min.   : 0.00
##  1st Qu.: 0.00   1st Qu.:40.57   1st Qu.: 0.00   1st Qu.: 0.00
##  Median :47.34   Median :49.30   Median :48.01   Median :46.80
##  Mean   :34.22   Mean   :39.37   Mean   :32.97   Mean   :34.82
##  3rd Qu.:52.08   3rd Qu.:52.10   3rd Qu.:51.44   3rd Qu.:52.44
##  Max.   :62.42   Max.   :61.56   Max.   :60.84   Max.   :64.45
```

2. Tidy the data

The snapshot of the shows each athlete's results at individual track meets. Athletes have six throws, but in these meets only one – their longest – actually matters. If all we wanted to do was talk regular track and field, we would have a very easy task: create a new column taking the max of each row, arrange the data frame by that column in descending order and we'd be done.

But the managers need to do and know much more than that! This is a sport of strategy, where every throw matters. Managers need a deeper analysis to choose their teams, craft their plan and make decisions on match-day.

We first need to make this standard "wide" view tidy data. The tidy data will allow us to compute our summary statistics.

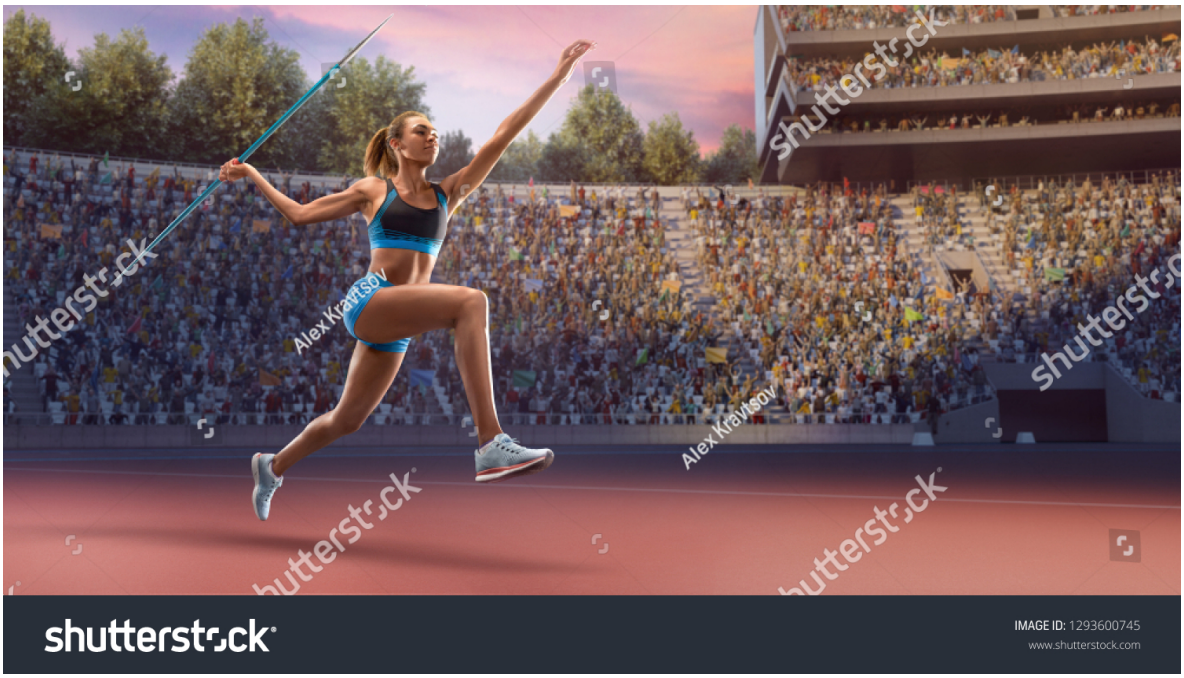
```
# Assign the tidy data to javelin_long
javelin_long <- javelin %>%
            gather(-c(1:2), key = "Flight", value = "Distance") %>%
```

```
            mutate(Flight = str_replace(Flight, pattern = "Flight",
                                        replacement = ""))

# Make Flight a numeric
javelin_long$Flight <- as.numeric(javelin_long$Flight)

# Examine the first 6 rows
head(javelin_long)
```

```
## # A tibble: 6 x 4
##   EventID Athlete            Flight Distance
##     <dbl> <chr>               <dbl>    <dbl>
## 1       8 Brittany Borman         1     54.0
## 2       8 Ariana Ince             1     49.0
## 3       8 Kara Patterson          1     50.1
## 4       8 Kimberley Hamilton      1     48.0
## 5       8 Laura Loht              1     44.4
## 6       8 Brianna Bain            1     49.3
```

3. Every throw matters

A throw is a foul if the athlete commits a technical violation during the throw. In javelin, the most common foul is stepping over the release line. Traditionally, the throw is scored as an "F" and it has no further significance. Athletes can also choose to pass on a throw – scored as a "P" – if they are content with their earlier throws and want to "save themselves" for later.



When we said every throw matters, the goal here is not for each player to have one great throw. All their throws in each event are summed together, and the team with the highest total distance wins the point. Fouls are scored as 0 and passing in which the manager and teammates would not be pleased.

Here, we examine which athletes cover the most distance in each of their meets, along with two ways to talk about their consistency.

```
# Filter out 0 Distance
# Find the total distance and standard deviation of Distance by Athlete & EventID
# Count the number of successes by Athlete & EventID
javelin_totals <- javelin_long %>%
                    filter(Distance > 0) %>%
                    group_by(Athlete, EventID) %>%
                    summarise(TotalDistance = sum(Distance),
                              StandardDev = round(sd(Distance), 3),
                              Success = n())

# View 10 rows of javelin_totals
javelin_totals[30:40, ]
```

```
## # A tibble: 11 x 5
## # Groups:   Athlete [5]
##    Athlete                     EventID TotalDistance StandardDev Success
##    <chr>                         <dbl>         <dbl>       <dbl>   <int>
##  1 Bethany Drake                  1766          195.        1.56       4
##  2 Bethany Drake                  1773          193.        1.06       4
##  3 Bethany Drake                  1859          206.        1.91       4
##  4 Bettie Wade                     174          103.        1.39       3
##  5 Betzabet Menendez Bejarano     1727          237.        1.25       6
##  6 Betzabet Menendez Bejarano     1734          209.        1.10       5
##  7 Brianna Bain                      8          202.        2.00       4
##  8 Brittany Borman                   8          333.        3.57       6
##  9 Brittany Borman                  20          238.        2.31       4
## 10 Brittany Borman                 173          223.        1.56       4
## 11 Brittany Borman                 218          219.        2.32       4
```

4. Find the clutch performers

In many traditional track meets, after the first three throws the leaders in the field are whittled down to the top eight (sometimes more, sometimes less) athletes. Like the meet overall, this is solely based on their best throw of those first three.

We give the choice to the managers. Of the three athletes who start each event, the manager chooses the two who will continue on for the last three throws. The manager will need to know which players tend to come alive – or at least maintain their form – in the late stages of a match. They also need to know if a player's first three throws are consistent with their playing history. Otherwise, they could make a poor decision about who stays in based only on the sample unfolding in front of them.

For now, let's examine just the top-line stat – total distance covered – for differences between early and late stages of the javelin match.

```
# Find the difference between the first three throws and last three throws by Athlete and EventID
javelin <- javelin %>%
            group_by(Athlete, EventID) %>%
            mutate(Early = sum(Flight1, Flight2, Flight3),
                   Late = sum(Flight4, Flight5, Flight6),
                   Diff = Late - Early)
# Examine the last ten rows
tail(javelin, 10)
```

```
## # A tibble: 10 x 11
## # Groups:   Athlete, EventID [10]
##    EventID Athlete   Flight1 Flight2 Flight3 Flight4 Flight5 Flight6 Early  Late
##      <dbl> <chr>       <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl> <dbl> <dbl>
##  1    1773 Melissa ~    47.6    48.7    47.5     0       0      45.6 144.   45.6
##  2    1773 Kaelyn C~    43.4    44.9    40      43.2    40.3    40.6 128.  124.
##  3    1859 Kara Win~    56.9    52.9    55.5    54.4    57.6    62.9 165.  175.
##  4    1859 Avione A~    56.5     0      54.4    51.6    54.3     0   111.  106.
##  5    1859 Ariana I~    51.9    53.5    52.4    56.0    55.2     0   158.  111.
##  6    1859 Bethany ~    49.9    51.0    54.2     0      50.6     0   155.   50.6
##  7    1859 Alyssa O~     0      53.7    52.1    51.5     0      52.8 106.  104.
##  8    1859 Dominiqu~    49.6    44.2    50.6    51.3    49.2    53.2 144.  154.
##  9    1859 Kristen ~    47.2    50.9     0      48.2    49.3    49.6  98.1 147.
## 10    1859 Rebekah ~    48.8     0      50.4    48.2     0      46.6  99.2  94.9
## # ... with 1 more variable: Diff <dbl>
```

5. Pull the pieces together for a new look at the athletes

The aggregate stats are in two data frame. By joining the two together, we can take our first rough look at how the athletes are compared to each other.

```
# Merge javelin_totals and javelin dataframes
javelin_totals <- javelin_totals %>%
                left_join(javelin, by = c("EventID", "Athlete")) %>%
                select(Athlete, TotalDistance,
                       StandardDev, Success, Diff) %>%
                mutate(StandardDev = round(StandardDev, 2),
                       Diff = round(Diff, 2))

# Examine the first ten rows
head(javelin_totals, 10)
```

```
## # A tibble: 10 x 5
## # Groups:   Athlete [4]
##    Athlete                 TotalDistance StandardDev Success    Diff
##    <chr>                           <dbl>       <dbl>   <int>   <dbl>
##  1 Abigail Gomez                    152.        1.23       3  -52.9
##  2 Abigail Gomez                    244.        1.63       5  -48
##  3 Abigail Gomez                    207.        2.97       4 -110.
##  4 Abigail Gomez                    222.        1.29       4   -3.11
##  5 Abigail Gomez                    155.        1.03       3   53.4
##  6 Abigail Gomez Hernandez          135.        0.72       3   45.6
##  7 Alicia DeShasier                 270.        2.15       5   60.0
##  8 Alicia DeShasier                 320.        2.26       6    0.74
##  9 Alicia DeShasier                 275.        1.53       5   53.5
## 10 Allison Updike                   147.        3.84       3  -46.6
```

6. Normalize the data to compare across stats

The four summary statistics - total distance, standard deviation, number of successful throws and the measure of early vs. late - are on different scales and measure very different things. Managers need to be

able to compare these to each other and then weigh them based on what is most important to their vision and strategy for the team. A simple normalization will allow for these comparisons.

```
# Create a normalize function
norm <- function(result) {
    (result - min(result)) / (max(result) - min(result))
}
```

```
# Agg Stats Column Headers
aggstats <- c("TotalDistance", "StandardDev", "Success", "Diff")

# Normalize the javelin_totals dataframe
javelin_norm <- javelin_totals %>%
                  ungroup() %>%
                  mutate_at(vars(aggstats), norm) %>%
                  group_by(Athlete) %>%
                  summarize_all(mean, na.rm = TRUE) %>%
                  mutate_if(is.numeric, round,4)

# Review javelin_norm
#dim(javelin_norm) ## 68 rows & 5 columns
head(javelin_norm)
```

```
## # A tibble: 6 x 5
##   Athlete                 TotalDistance StandardDev Success  Diff
##   <chr>                           <dbl>       <dbl>   <dbl> <dbl>
## 1 Abigail Gomez                   0.446       0.267   0.45  0.383
## 2 Abigail Gomez Hernandez         0.244       0.114   0.25  0.720
## 3 Alicia DeShasier                0.753       0.326   0.833 0.687
## 4 Allison Updike                  0.283       0.639   0.25  0.320
## 5 Alyssa Olin                     0.469       0.250   0.5   0.309
## 6 Ariana Ince                     0.660       0.342   0.692 0.446
```

7. What matters most when building the squad?

Managers have to decide what kind of players they want on their team - who matches their vision, who has the skills they need to play their style of athletics and - ultimately - who will deliver the wins. A risk-averse manager will want players who rarely foul. The steely-eyed manager will want the players who can deliver the win with their final throws.

Like any other sport (or profession), rarely will any one player be equally strong in all areas. Managers have to make trade-offs in selecting their teams. The first batch of managers have the added disadvantage of selecting players based on data from a related but distinct sport. The data comes from traditional track and field meets, where the motivations and goals are much different than our own.

This is why managers make the big money and get fired when results go south.


```
# Choose the weights to select players to be part of the squad
weights <- c(4, 1, 3, 2)

# Calculate a total score for each athlete
# Select the top 5 athletes
javelin_team <- javelin_norm %>%
                  mutate(TotalScore = round(
                    (TotalDistance * weights[1]) +
                    (StandardDev * weights[2]) +
                    (Success * weights[3]) +
                    (Diff * weights[4]), 2)
                    ) %>%
                  # Descending order
                  arrange(desc(TotalScore)) %>%
                  # Select Athlete & TotalScore
                  select(Athlete, TotalScore) %>%
                  # Top 5
                  slice(1:5)

javelin_team
```

```
## # A tibble: 5 x 2
##   Athlete             TotalScore
##   <chr>                    <dbl>
## 1 Dominique Ouellette       7.7
## 2 Madalaine Stulce          7.57
## 3 Heather Bergmann          7.39
## 4 Maggie Malone             7.24
## 5 Alicia DeShasier          7.21
```

8. Get to know your players

The data has spoken! Now we have our five javelin throwers, but we still don't really know them. The `javelin_totals` data frame has the data that went into the decision process earlier, so we will account that. This gives us an idea of what they each bring to the team.

We can also take a look at how they compare to the pool of athletes we started from by taking the mean and maximum of each statistic.

```
# Create a team_stats dataframe of players' avg stats from javeline_totals dataframe
team_stats <- javelin_totals %>%
                # Filter for the selected top 5 athletes
                filter(Athlete %in% javelin_team$Athlete) %>%
                # Find the avg
                summarize_all(mean, na.rm = TRUE) %>%
                mutate_if(is.numeric, round, 2)

# Examine team_stats
team_stats
```

```
## # A tibble: 5 x 5
##   Athlete             TotalDistance StandardDev Success  Diff
##   <chr>                       <dbl>       <dbl>   <dbl> <dbl>
## 1 Alicia DeShasier             288.        1.98    5.33 38.1
## 2 Dominique Ouellette          299.        2.91    6     2.88
## 3 Heather Bergmann             283.        2.28    6     2.41
## 4 Madalaine Stulce             275.        4.47    6    -6.42
## 5 Maggie Malone                293.        2       5    61.1
```

```
# Create the pool_stats dataframe of max and avg values for each stats acrosse the entire pool of athle

# pool_stats dataframe
pool_stats <- data.frame(do.call('cbind',
            sapply(javelin_totals,
              # Find the max and avg for each stats
              function(x) if(is.numeric(x)) c(max(x), mean(x)))))

# Create a column for pool_stats
pool_stats$MaxAvg <- c("Maximum", "Average")

# Long format
pool_stats <- pool_stats %>%
                gather(key="Statistic",
                       value="Aggregate", -MaxAvg) %>%
                mutate(Aggregate = round(Aggregate,2))

# Examine pool_stats
pool_stats
```

```
##    MaxAvg     Statistic Aggregate
## 1 Maximum TotalDistance    362.54
## 2 Average TotalDistance    222.05
## 3 Maximum   StandardDev      5.99
```

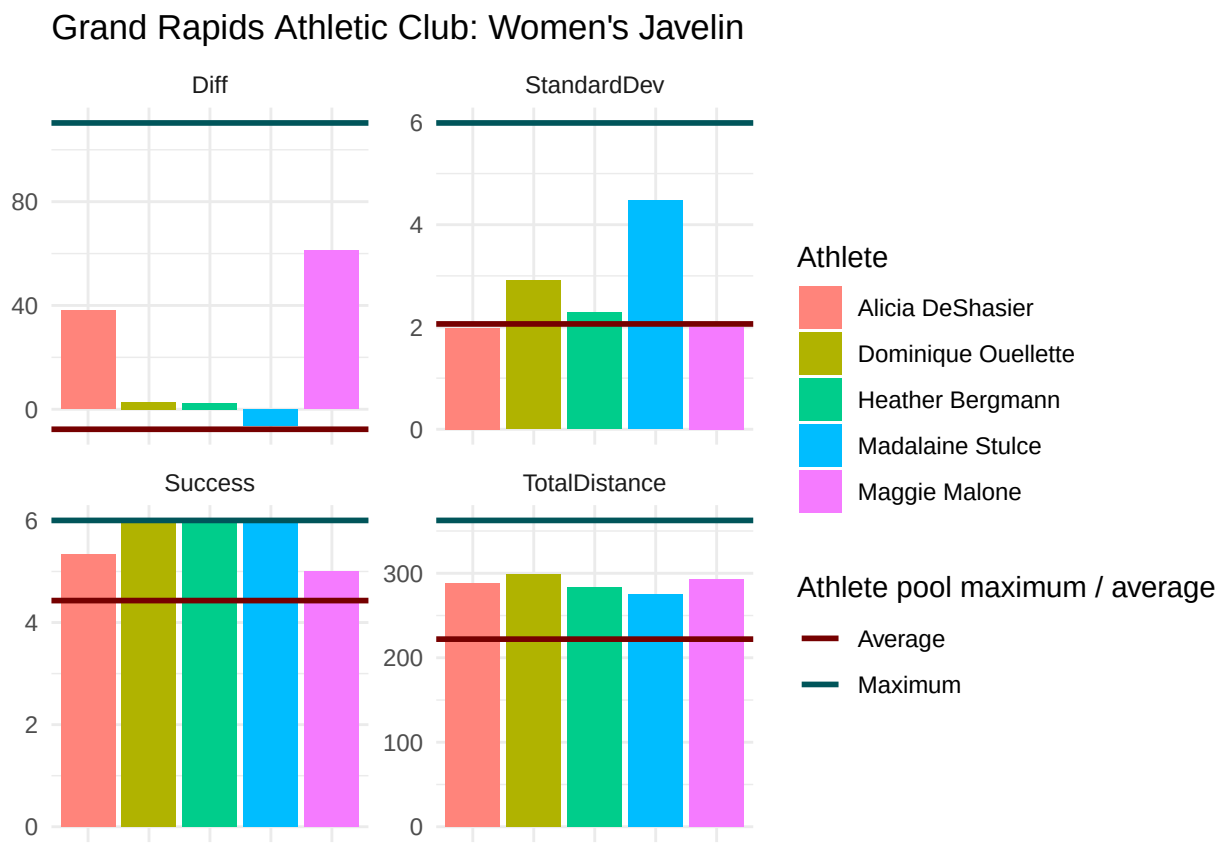
```
## 4 Average   StandardDev      2.06
## 5 Maximum       Success      6.00
## 6 Average       Success      4.43
## 7 Maximum          Diff    110.34
## 8 Average          Diff     -7.72
```

9. Make the case to the front office

The manager knows what she wants out of the team and has the data to support her choices, but she still needs to defend her decisions to the team owners. They do write the checks, after all.

The owners are busy people. Many of them work other jobs and own other companies. They trust their managers, so as long the manager can give them an easy-to-digest visual presentation of why they should sign these five athletes out of all the others, they will approve. A series of plots showing how each athlete compares to the maximum and the average of each statistic will be enough for them.

```
# Create a 2x2 grid to show different aggregate stats
# Each bar of the plot represent each athlete on the squad and line showing the max and avg values from
p <- team_stats %>%
        # Long format
        gather(-Athlete, key = "Statistic", value = "Aggregate") %>%
        # Bar plot
        ggplot(aes(x = Athlete, y = Aggregate, fill = Athlete)) +
        geom_bar(stat = "identity") +
        facet_wrap(~Statistic, scales = "free_y") +
        geom_hline(data=pool_stats,
                   aes(yintercept=Aggregate,
                       group=Statistic, color=MaxAvg), size=1) +
        labs(title="Grand Rapids Athletic Club: Women's Javelin",
             color="Athlete pool maximum / average") +
        scale_fill_hue(l=70) +
        scale_color_hue(l=20) +
      theme_minimal() +
      theme(axis.text.x=element_blank(),
            axis.title.x=element_blank(),
            axis.title.y=element_blank())
# Plot
p
```



10. Time to throw down

Before the athletics season opens, the manager will perform similar analyses for the other throws, the jumps, and running events. Then, we'll game out different permutations of the team and opponent to come up with the best lineup and make the best decisions on match day. For now, since it's what we know best and we're almost out of time, let's simulate a simple javelin match.

The winner is the team that throws the highest combined distance: six throws from each of our three players against six throws from each of the opponent's three players.

Our team selections are **Maggie Malone**, **Alicia DeShasier,** and **Dominique Ouellette.**

```
# Select 3 players from the top 5 shortlist who should compete
home <- c(5, 1, 2)
# Randomly select 3 players for the away team
away <- sample(1:nrow(javelin_totals), 3, replace=FALSE)

# Simulation
HomeTeam <- round(sum(team_stats$TotalDistance[home]),2)
AwayTeam <- round(sum(javelin_totals$TotalDistance[away]),2)

# Print Result
print(paste0("Javelin match, Final Score: ", HomeTeam, " - ",
             AwayTeam))
```

```
## [1] "Javelin match, Final Score: 880.35 - 611.55"
```

```
ifelse(HomeTeam > AwayTeam, print("Win!"), print("Sometimes you just have to take the L."))
```

```
## [1] "Win!"
```

```
## [1] "Win!"
```

In this simulation example, the team selection we chose won against a randomized opponent.