From run expectancy to WAR markdown version

03 August 2020

From Run Expectancy to Wins Above Replacement (WAR)

The Run Expectancy Matrix is a context-dependent measure of productivity. By introducing the context of a hit or a walk to the state of the game, it provides a more accurate measure of the contribution of a position player.

In the previous notebook we derived the Runs Value of an event, measured by the runs scored on the play, plus the Runs Expectancy at the end of the play minus the Run Expectancy at the beginning of the play. We also measured the season-long contribution of the position player as the sum of Runs Values across all plate appearances.

The Run Expectancy Matrix has been used in recent years to develop new ways of valuing player contributions. Because the underlying data has not been easily available until recently, and because deriving the matrix from the data was not easy in the past, the Run Expectancy Matrix has been adapted in simple ways.

In the previous notebooks we looked at On Base Percentage (OBP) and Slugging (SLG). In recent years an increasingly popular statistic for measuring performance has been On-base Plus Slugging (OPS). OPS = OBP + SLG (hence the name "On-base Plus Slugging"). This measure is useful since it recognizes both the value of getting on base and extra value of doubles, triples, and home runs.

However, as a measure of contribution, it implicitly imposes equal weights on each element of the formula. Is a double worth exactly twice a single in terms of runs scored? We shouldn't assume that - we should let the data tell us. This is exactly the idea behind the use of "linear weights" to calculate weighted on-base average (wOBA). The linear weights are taken from the Run Expectancy Matrix.

There are different versions of wOBA out there depending on different adjustments preferred by different analysts. Remember that the Run Expectancy Matrix changes from year to year, and hence analysts like to evaluate a player relative to the RE Matrix of the season from which their stats are derived.

For example, Fangraphs, a popular baseball statistics site, defined the following formula for wOBA in 2018:

wOBA = (0.69$NIBB$ + 0.72HBP + 0.881B + 1.2472B + 1.5783B + 2.031HR)/(AB + BB - IBB +SF + HBP)

where NIBB and IBB are non-intentional and intentional walks.

The weights in wOBA are taken from the Run Expectancy Matrix. Sites such as Fangraphs and Baseball Reference generate these statistics, and then use the coefficients to generate measures such as wOBA, which they consider an improvement on On-base Plus Slugging (OPS).

To take this further, analysts have created statistics to account for the total contribution of a player, one of the most popular being Wins Above Replacement (WAR). Most measures of WAR are based around calculations of wOBA for position players. We can summarize three aims in the creation of these statistics:

(i) To account for the four dimensions of on-field performance - batting, base-running, pitching, and fielding. The first two constitute offense, the last two defense. While base-running and fielding make a difference, it is batting and pitching which generate most value added in the game. One can think of pitching, in terms of the Run Expectancy Matrix, as the opposite of batting- if a play generates a positive Run Value for the batter, it creates an equal and opposite Run Value for the pitcher.

(ii) To adjust for differences in productivity which are unrelated to the skill of the player. The obvious adjustment here is for the size of the ballpark- some are easier for offense than than others, so player stats can be adjusted to take account of these differences. More subtly, a left-handed batter has an advantage over a right-handed batter when facing a right-handed pitcher - the so-called "Platoon Effect" - this advantage is often calculated and then deducted from the batter's Run Value.

(iii) To express the value of a player relative to a replacement player. If a player is injured, the team typically brings up a player from the minor leagues. Thus the true value of a player is his productivity relative to this likely replacement. Replacement value is usually calculated by averaging the statistics of players with a small number of appearances over the season, since these players are likely to be similar to players called up from the minors.

So far we have measured performance in terms of runs. For ease of interpretation (only), WAR is expressed in terms of wins, where a win is typically valued at around ten wins.

In what follows we're going to focus only on batting (which is essentially offensive WAR) and ignore all of the adjustments. Not adjusting the figures might reduce their reliability, but our use of the RE Matrix has one big advantage over WAR based on wOBA. While Run Expectancy takes into account the context of an event, the linear weights are applied to average performance statistics over the season and are therefore not context dependent. Since we have the Run Expectancy Matrix to work with, our performance stats are context dependent.

Runs Values

Here we are going to look at the seasons 2016 and 2017. As ever, we begin by loading the packages we need.

```
library("tidyverse",quietly = TRUE)
library("readxl",quietly = TRUE)
```

Writing a Function to Create the Runs Value of Each Event in a Season

In the previous workbook we used the MLBAM data for 2018 to derive the Run Expectancy Matrix and then create the Runs Value of every event. MLBAM has made available files in the same format for each season since 2014. Because the files are structured identically with identical names, we can turn the code into a Python function to produce the RE Matrix and Runs Values for any season, depending on the the file we choose.

The function always starts with 'def'. Here the name Run_Expectancy is given to it, and it is defined by a path, which is where the data is to be found on which to execute the function. The content of the function (which must be indented) is simply a copy of all the lines of code used in the previous notebook, the only difference now is that the specific name 'MLBAM18' is replaced with a generic name 'RE', while the actual name applied when the function is run is defined when we specify the path. **The function is completed by specifying 'return', the name of the output and a semi colon (;).**

```
Run_Expectancy <- function(path){
    RE = read.csv(path)
    RE <- RE %>% dplyr::select(-X)
    RE <- RE %>% dplyr::select('home_team','away_team','half','gameId','batterName',
                               'batterId','event', 'start1B', 'start2B', 'start3B',
        'end1B', 'end2B', 'end3B','startOuts','endOuts','runsFuture',
        'runsOnPlay','outsInInning','venueId','batterPos')
    RE[,'Start1'] = ifelse(is.na(RE$start1B),0,1)
    RE[,'Start2'] = ifelse(is.na(RE$start2B),0,1)
    RE[,'Start3'] = ifelse(is.na(RE$start3B),0,1)
    RE[,'Start_State'] = paste(RE[,'Start1'],RE[,'Start2'],
                               RE[,'Start3']," ",RE[,'startOuts'])
    RE[,'End1'] = ifelse(is.na(RE$end1B),0,1)
    RE[,'End2'] = ifelse(is.na(RE$end2B),0,1)
    RE[,'End3'] = ifelse(is.na(RE$end3B),0,1)
    RE[,'End_State'] =   paste(RE[,'End1'],RE[,'End2'],
                               RE[,'End3']," ",RE[,'endOuts'])
    RE <- RE %>% filter(((Start_State != End_State) | (runsOnPlay > 0))
                        & (outsInInning == 3))
    Start_RunExp <- RE %>%
```

```
                    group_by(Start_State)%>%
                    dplyr::summarise(runsFuture = mean(runsFuture)) %>%
                    rename(Start_RE = runsFuture)
    RE = left_join(RE, Start_RunExp, by='Start_State')
    Base_State_3 <- as.data.frame(rbind(c('0 0 0   3', 0),
                          c('0 0 1   3', 0),
                          c('0 1 0   3', 0),
                          c('0 1 1   3', 0),
                          c('1 0 0   3', 0),
                          c('1 0 1   3', 0),
                          c('1 1 0   3', 0),
                          c('1 1 1   3', 0)))
    Base_State_3 <- Base_State_3 %>%
                    rename( Start_State = V1, Start_RE = V2) %>%
                    mutate( Start_State = as.character(Start_State),
                            Start_RE = as.numeric(as.character(Start_RE)))
    Start_RunExp <- rbind(Start_RunExp,Base_State_3)
    End_RunExp  <- Start_RunExp %>% rename(End_State = Start_State, End_RE = Start_RE)
    RE <- left_join(RE, End_RunExp, by='End_State')
    RE[,'Run_Value'] = RE[,'runsOnPlay'] +
                       RE[,'End_RE'] - RE[,'Start_RE']
    return(RE)
}
```

```
RE_17 = Run_Expectancy("MLBAM17.csv")
head(RE_17)
```

```
##   home_team away_team   half                          gameId   batterName
## 1       tba       nya    top gid_2017_04_02_nyamlb_tbamlb_1      Gardner
## 2       tba       nya    top gid_2017_04_02_nyamlb_tbamlb_1   Sanchez, G
## 3       tba       nya    top gid_2017_04_02_nyamlb_tbamlb_1         Bird
## 4       tba       nya    top gid_2017_04_02_nyamlb_tbamlb_1     Holliday
## 5       tba       nya bottom gid_2017_04_02_nyamlb_tbamlb_1 Dickerson, C
## 6       tba       nya bottom gid_2017_04_02_nyamlb_tbamlb_1    Kiermaier
##   batterId     event start1B start2B start3B  end1B  end2B  end3B
## 1   458731    Flyout      NA      NA      NA     NA     NA     NA
## 2   596142 Groundout      NA      NA      NA     NA     NA     NA
## 3   595885      Walk      NA      NA      NA 595885     NA     NA
## 4   407812 Groundout  595885      NA      NA     NA     NA     NA
## 5   572816    Single      NA      NA      NA 572816     NA     NA
## 6   595281    Double  572816      NA      NA     NA 595281 572816
##   startOuts endOuts runsFuture runsOnPlay outsInInning venueId batterPos
## 1         0       1          0          0            3      12        LF
## 2         1       2          0          0            3      12         C
## 3         2       2          0          0            3      12        1B
```

```
## 4         2       3          0          0           3      12        DH
## 5         0       0          3          0           3      12        DH
## 6         0       0          3          0           3      12        CF
##   Start1 Start2 Start3 Start_State End1 End2 End3 End_State  Start_RE
## 1      0      0      0   0 0 0   0    0    0    0 0 0 0   1 0.5163748
## 2      0      0      0   0 0 0   1    0    0    0 0 0 0   2 0.2721761
## 3      0      0      0   0 0 0   2    1    0    0 1 0 0   2 0.1080383
## 4      1      0      0   1 0 0   2    0    0    0 0 0 0   3 0.2253648
## 5      0      0      0   0 0 0   0    1    0    0 1 0 0   0 0.5163748
## 6      1      0      0   1 0 0   0    0    1    1 0 1 1   0 0.9129211
##      End_RE  Run_Value
## 1 0.2721761 -0.2441987
## 2 0.1080383 -0.1641379
## 3 0.2253648  0.1173266
## 4 0.0000000 -0.2253648
## 5 0.9129211  0.3965462
## 6 2.0513761  1.1384551
```

We Now Repeat the Same Exercise for 2016

```
RE_16 = Run_Expectancy("MLBAM16.csv")
head(RE_16)
```

```
##   home_team away_team   half                          gameId   batterName
## 1       pit       sln    top gid_2016_04_03_slnmlb_pitmlb_1 Carpenter, M
## 2       pit       sln    top gid_2016_04_03_slnmlb_pitmlb_1         Pham
## 3       pit       sln    top gid_2016_04_03_slnmlb_pitmlb_1     Holliday
## 4       pit       sln bottom gid_2016_04_03_slnmlb_pitmlb_1         Jaso
## 5       pit       sln bottom gid_2016_04_03_slnmlb_pitmlb_1    McCutchen
## 6       pit       sln bottom gid_2016_04_03_slnmlb_pitmlb_1       Freese
##   batterId        event start1B start2B start3B  end1B end2B  end3B
## 1   572761    Groundout      NA      NA      NA     NA    NA     NA
## 2   502054    Groundout      NA      NA      NA     NA    NA     NA
## 3   407812    Strikeout      NA      NA      NA     NA    NA     NA
## 4   444379    Groundout      NA      NA      NA     NA    NA     NA
## 5   457705 Hit By Pitch      NA      NA      NA 457705    NA     NA
## 6   501896       Single  457705      NA      NA 501896    NA 457705
##   startOuts endOuts runsFuture runsOnPlay outsInInning venueId batterPos
## 1         0       1          0          0           3      31        3B
## 2         1       2          0          0           3      31        LF
## 3         2       3          0          0           3      31        1B
## 4         0       1          0          0           3      31        1B
## 5         1       1          0          0           3      31        CF
## 6         1       1          0          0           3      31        3B
```

```
##   Start1 Start2 Start3 Start_State End1 End2 End3 End_State  Start_RE
## 1      0      0      0   0 0 0   0    0    0    0 0 0 0   1 0.4983772
## 2      0      0      0   0 0 0   1    0    0    0 0 0 0   2 0.2686783
## 3      0      0      0   0 0 0   2    0    0    0 0 0 0   3 0.1063051
## 4      0      0      0   0 0 0   0    0    0    0 0 0 0   1 0.4983772
## 5      0      0      0   0 0 0   1    1    0    0 1 0 0   1 0.2686783
## 6      1      0      0   1 0 0   1    1    0    1 1 0 1   1 0.5122249
##      End_RE  Run_Value
## 1 0.2686783 -0.2296989
## 2 0.1063051 -0.1623732
## 3 0.0000000 -0.1063051
## 4 0.2686783 -0.2296989
## 5 0.5122249  0.2435466
## 6 1.1967773  0.6845524
```

Self Test

Repeat this exercise for the event data from 2015.

1. Comparing Event Run Values

To make comparisons between the two seasons we are going (a) use .groupby to generate a statistic for a particular variable (an event, a player, and a team) (b) merge the data for the two seasons (c) derive the correlation and, where suitable, create a scatter diagram.

We start by comparing the Runs Value of events in each season.

```
#2016

Event_Value16 <- RE_16 %>% group_by(event) %>%
                           dplyr::summarise(Run_Value = mean(Run_Value))%>%
                           rename(RV16 = Run_Value)
head(Event_Value16)
```

```
## # A tibble: 6 x 2
##   event                  RV16
##   <fct>                 <dbl>
## 1 Batter Interference  -0.285
## 2 Bunt Groundout       -0.219
## 3 Bunt Lineout         -0.352
## 4 Bunt Pop Out         -0.343
## 5 Catcher Interference  0.302
## 6 Double                0.743
```

```
tail(Event_Value16)
```

```
## # A tibble: 6 x 2
```

```
##   event            RV16
##   <fct>           <dbl>
## 1 Single          0.440
## 2 Strikeout      -0.260
## 3 Strikeout - DP -0.623
## 4 Triple          1.02
## 5 Triple Play    -1.64
## 6 Walk            0.299
```

```
#2017

Event_Value17 <- RE_17 %>% group_by(event) %>%
                           dplyr::summarise(Run_Value = mean(Run_Value))%>%
                           rename(RV17 = Run_Value)
head(Event_Value17)
```

```
## # A tibble: 6 x 2
##   event                 RV17
##   <fct>                <dbl>
## 1 Batter Interference  -0.430
## 2 Bunt Groundout       -0.209
## 3 Bunt Lineout         -0.328
## 4 Bunt Pop Out         -0.373
## 5 Catcher Interference  0.399
## 6 Double                0.779
```

```
tail(Event_Value17)
```

```
## # A tibble: 6 x 2
##   event            RV17
##   <fct>           <dbl>
## 1 Single          0.456
## 2 Strikeout      -0.270
## 3 Strikeout - DP -0.683
## 4 Triple          1.06
## 5 Triple Play    -1.47
## 6 Walk            0.323
```

```
# merge 2016 and 2017

RVseasons= merge(Event_Value16,Event_Value17,by = 'event')
head(RVseasons)
```

```
##                   event       RV16       RV17
## 1  Batter Interference -0.2846494 -0.4300188
## 2       Bunt Groundout -0.2188264 -0.2094114
## 3         Bunt Lineout -0.3522947 -0.3282921
```

```
## 4         Bunt Pop Out -0.3428022 -0.3732253
## 5 Catcher Interference  0.3016227  0.3990697
## 6               Double  0.7434674  0.7793378
```

```
tail(RVseasons)
```

```
##             event       RV16       RV17
## 25         Single  0.4402576  0.4557730
## 26      Strikeout -0.2599399 -0.2699846
## 27 Strikeout - DP -0.6227867 -0.6828187
## 28         Triple  1.0239115  1.0636067
## 29    Triple Play -1.6375429 -1.4712235
## 30           Walk  0.2994038  0.3226351
```

```
# the correlation coefficient
cor(RVseasons$RV16,RVseasons$RV17)
```

```
## [1] 0.9949636
```

Conclusion on Event Runs Value Comparison

Although the exact Runs Value of an event can vary from year to year, the pattern of Runs Values across different events appear remarkably consistent, with a +0.995 correlation coefficient.

Self Test

Use the Run Expectancy data from 2015 to calculate the correlation coefficient for event Run Values comparing 2015 with 2017.

2. Comparing Position Player Run Values

We now repeat this comparison, grouping the data by player rather than by event. We use batterId as the matching variable, since each player has a unique number in the data, rather than batterName, since some players may have the same name.

```
Player_Value16 <- RE_16 %>%
  group_by(batterId,batterName)%>%
  dplyr::summarise(Run_Value = sum(Run_Value))%>%
  rename(RV16 = Run_Value)

head(Player_Value16)
```

```
## # A tibble: 6 x 3
## # Groups:   batterId [6]
##   batterId batterName     RV16
##      <int> <fct>         <dbl>
```

```
## 1   112526 Colon        -15.4
## 2   120074 Ortiz         60.9
## 3   121347 Rodriguez, A -10.7
## 4   134181 Beltre        29.7
## 5   136860 Beltran       21.3
## 6   150029 Werth          6.37
```

```
tail(Player_Value16)
```

```
## # A tibble: 6 x 3
## # Groups:   batterId [6]
##   batterId batterName    RV16
##      <int> <fct>        <dbl>
## 1   645848 Alvarez, D   1.46
## 2   649557 Diaz, A     16.7
## 3   656546 Hoffman     -2.04
## 4   656941 Schwarber   -0.780
## 5   660162 Moncada     -2.72
## 6   666560 Park       -12.0
```

```
Player_Value17 <- RE_17 %>%
  group_by(batterId,batterName)%>%
  dplyr::summarise(Run_Value = sum(Run_Value))%>%
  rename(RV17 = Run_Value)

head(Player_Value17)
```

```
## # A tibble: 6 x 3
## # Groups:   batterId [6]
##   batterId batterName    RV17
##      <int> <fct>        <dbl>
## 1   112526 Colon       -6.05
## 2   134181 Beltre      27.2
## 3   136860 Beltran    -16.9
## 4   150029 Werth       -7.25
## 5   276520 Arroyo, B   -3.53
## 6   282332 Sabathia    -0.292
```

```
tail(Player_Value17)
```

```
## # A tibble: 6 x 3
## # Groups:   batterId [6]
##   batterId batterName  RV17
##      <int> <fct>      <dbl>
## 1   664023 Happ, I     1.10
## 2   664056 Bader      -1.44
## 3   664057 Stevenson  -8.07
```

```
## 4   664641 Miranda    -2.15
## 5   666561 Hwang      -3.51
## 6   669720 Hays       -3.33
```

```
Players <- merge(Player_Value16,Player_Value17, by = 'batterId')
head(Players)
```

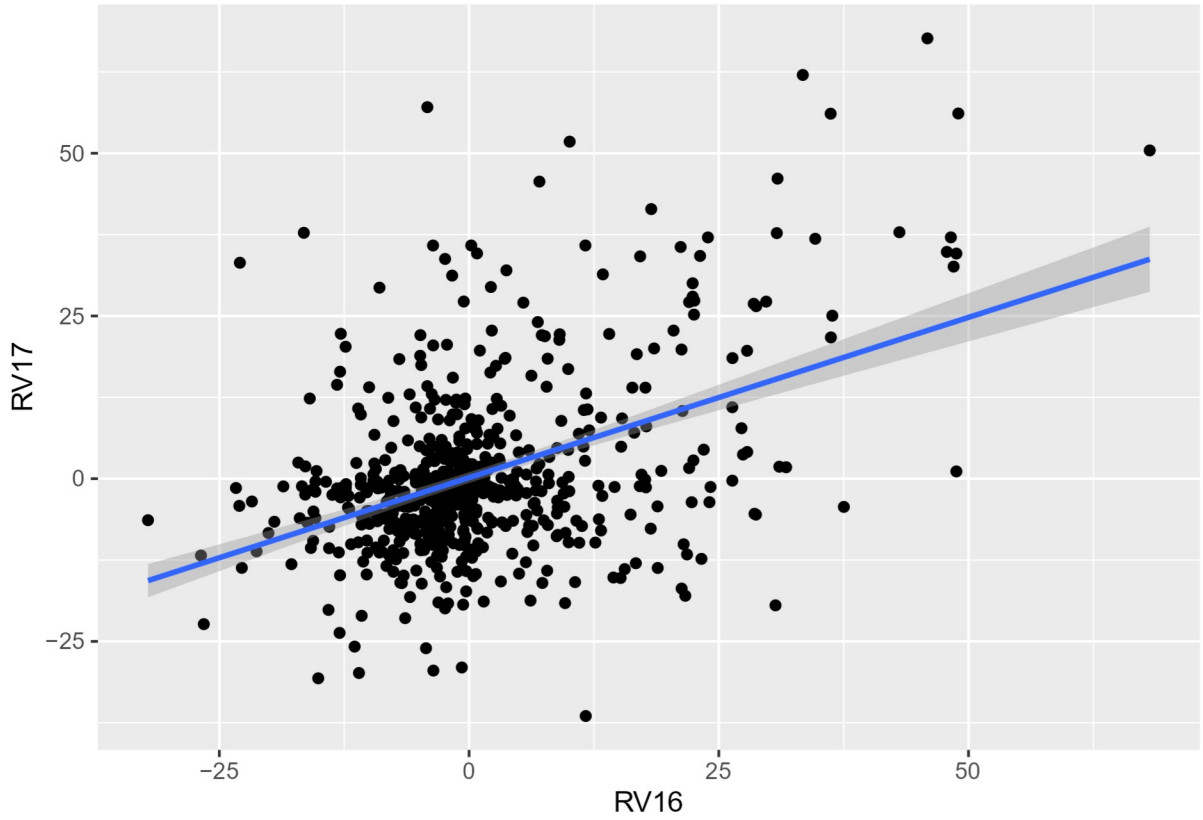
```
##   batterId batterName.x        RV16 batterName.y        RV17
## 1   112526        Colon -15.3808409        Colon  -6.0530604
## 2   134181       Beltre  29.7451145       Beltre  27.2050504
## 3   136860      Beltran  21.2793930      Beltran -16.8830416
## 4   150029        Werth   6.3684482        Werth  -7.2461447
## 5   282332     Sabathia  -0.9445797     Sabathia  -0.2924953
## 6   285079       Dickey  -0.2916857       Dickey -11.9554668
```

```
tail(Players)
```

```
##     batterId batterName.x       RV16 batterName.y       RV17
## 714   643393      Kemp, T -9.1368764      Kemp, T  -1.489757
## 715   643603        White -9.4823687        White   2.336055
## 716   649557      Diaz, A 16.6953138      Diaz, A -12.989348
## 717   656546      Hoffman -2.0367482      Hoffman  -8.617175
## 718   656941    Schwarber -0.7797263    Schwarber   1.261717
## 719   660162      Moncada -2.7220853      Moncada   3.754225
```

Using a scatter diagram is a good way to compare consistency between seasons. If performance from season to season were perfectly correlated, we would observe every dot, which indicates an individual player, ranged along a 45 degree line running for the bottom left to the top right of the chart. In reality, the data is far more dispersed.

```
ggplot(data = Players)+
  geom_point(aes(x= RV16, y = RV17),color= 'black')+
  geom_smooth(aes(x= RV16, y = RV17),method = "lm",formula = y ~ x)
```



As before, we calculate the correlation coefficient. There clearly is a positive correlation, but it is not nearly as strong as it is for the event values.

```
cor(Players$RV16, Players$RV17)
```

```
## [1] 0.4430088
```

Self Test

One explanation for the low correlation may be that in one or both years some players have relatively few at-bats, and hence the performance measure is noisy. Identify the number of appearances by players in the 2016 and 2017 data, and then derive the correlation coefficient. What do you conclude from this?

3. Comparing Team Run Values

The next step is to compare the runs values for each team across the season, to see how closely these correlate with winning records of the teams.

```
# Since each event is identified as occurring at
# the top of the inning (away team) or bottom of the inning (home team),
# we can use this variable to associate each event with the offensive record of the ba
RE_17$away_team <- as.character(RE_17$away_team)
```

```
RE_17$home_team <- as.character(RE_17$home_team)

RE_17[,'team']= ifelse(RE_17$half=='top',RE_17$away_team,RE_17$home_team)
head(RE_17)
```

```
##   home_team away_team   half                          gameId   batterName
## 1       tba       nya    top gid_2017_04_02_nyamlb_tbamlb_1      Gardner
## 2       tba       nya    top gid_2017_04_02_nyamlb_tbamlb_1   Sanchez, G
## 3       tba       nya    top gid_2017_04_02_nyamlb_tbamlb_1         Bird
## 4       tba       nya    top gid_2017_04_02_nyamlb_tbamlb_1     Holliday
## 5       tba       nya bottom gid_2017_04_02_nyamlb_tbamlb_1 Dickerson, C
## 6       tba       nya bottom gid_2017_04_02_nyamlb_tbamlb_1    Kiermaier
##   batterId     event start1B start2B start3B  end1B  end2B  end3B
## 1   458731    Flyout      NA      NA      NA     NA     NA     NA
## 2   596142 Groundout      NA      NA      NA     NA     NA     NA
## 3   595885      Walk      NA      NA      NA 595885     NA     NA
## 4   407812 Groundout  595885      NA      NA     NA     NA     NA
## 5   572816    Single      NA      NA      NA 572816     NA     NA
## 6   595281    Double  572816      NA      NA     NA 595281 572816
##   startOuts endOuts runsFuture runsOnPlay outsInInning venueId batterPos
## 1         0       1          0          0            3      12        LF
## 2         1       2          0          0            3      12         C
## 3         2       2          0          0            3      12        1B
## 4         2       3          0          0            3      12        DH
## 5         0       0          3          0            3      12        DH
## 6         0       0          3          0            3      12        CF
##   Start1 Start2 Start3 Start_State End1 End2 End3 End_State  Start_RE
## 1      0      0      0   0 0 0   0    0    0    0 0 0 0   1 0.5163748
## 2      0      0      0   0 0 0   1    0    0    0 0 0 0   2 0.2721761
## 3      0      0      0   0 0 0   2    1    0    0 1 0 0   2 0.1080383
## 4      1      0      0   1 0 0   2    0    0    0 0 0 0   3 0.2253648
## 5      0      0      0   0 0 0   0    1    0    0 1 0 0   0 0.5163748
## 6      1      0      0   1 0 0   0    0    1    1 0 1 1   0 0.9129211
##      End_RE  Run_Value team
## 1 0.2721761 -0.2441987  nya
## 2 0.1080383 -0.1641379  nya
## 3 0.2253648  0.1173266  nya
## 4 0.0000000 -0.2253648  nya
## 5 0.9129211  0.3965462  tba
## 6 2.0513761  1.1384551  tba
```

```
tail(RE_17)
```

```
##        home_team away_team   half                          gameId
## 184746       chn       cin    top gid_2017_10_01_cinmlb_chnmlb_1
```

```
## 184747       chn       cin    top gid_2017_10_01_cinmlb_chnmlb_1
## 184748       chn       cin bottom gid_2017_10_01_cinmlb_chnmlb_1
## 184749       chn       cin bottom gid_2017_10_01_cinmlb_chnmlb_1
## 184750       chn       cin bottom gid_2017_10_01_cinmlb_chnmlb_1
## 184751       chn       cin bottom gid_2017_10_01_cinmlb_chnmlb_1
##        batterName batterId      event start1B start2B start3B  end1B
## 184746   Hamilton   571740       Walk  608385      NA      NA 571740
## 184747     Vincej   572227 Runner Out  571740  608385      NA     NA
## 184748   Davis, T   543089  Groundout      NA      NA      NA     NA
## 184749 Almora Jr.   546991   Home Run      NA      NA      NA     NA
## 184750    Freeman   502273  Groundout      NA      NA      NA     NA
## 184751    Zobrist   450314  Groundout      NA      NA      NA     NA
##         end2B end3B startOuts endOuts runsFuture runsOnPlay outsInInning
## 184746 608385    NA         2       2          0          0            3
## 184747     NA    NA         2       3          0          0            3
## 184748     NA    NA         0       1          1          0            3
## 184749     NA    NA         1       1          1          1            3
## 184750     NA    NA         1       2          0          0            3
## 184751     NA    NA         2       3          0          0            3
##        venueId batterPos Start1 Start2 Start3 Start_State End1 End2 End3
## 184746      17        CF      1      0      0   1 0 0   2    1    1    0
## 184747      17        UN      1      1      0   1 1 0   2    0    0    0
## 184748      17        UN      0      0      0   0 0 0   0    0    0    0
## 184749      17        UN      0      0      0   0 0 0   1    0    0    0
## 184750      17        UN      0      0      0   0 0 0   1    0    0    0
## 184751      17        LF      0      0      0   0 0 0   2    0    0    0
##        End_State  Start_RE    End_RE  Run_Value team
## 184746 1 1 0   2 0.2253648 0.4497824  0.2244176  cin
## 184747 0 0 0   3 0.4497824 0.0000000 -0.4497824  cin
## 184748 0 0 0   1 0.5163748 0.2721761 -0.2441987  chn
## 184749 0 0 0   1 0.2721761 0.2721761  1.0000000  chn
## 184751 0 0 0   3 0.1080383 0.0000000 -0.1080383  chn
```

```
RE_16$away_team <- as.character(RE_16$away_team)
RE_16$home_team <- as.character(RE_16$home_team)

RE_16[,'team']= ifelse(RE_16$half=='top',RE_16$away_team,RE_16$home_team)
head(RE_16)
```

```
##   home_team away_team   half                          gameId   batterName
## 1       pit       sln    top gid_2016_04_03_slnmlb_pitmlb_1 Carpenter, M
## 2       pit       sln    top gid_2016_04_03_slnmlb_pitmlb_1         Pham
## 3       pit       sln    top gid_2016_04_03_slnmlb_pitmlb_1     Holliday
## 4       pit       sln bottom gid_2016_04_03_slnmlb_pitmlb_1         Jaso
```

```
## 5       pit       sln bottom gid_2016_04_03_slnmlb_pitmlb_1    McCutchen
## 6       pit       sln bottom gid_2016_04_03_slnmlb_pitmlb_1       Freese
##   batterId        event start1B start2B start3B  end1B end2B  end3B
## 1   572761    Groundout      NA      NA      NA     NA    NA     NA
## 2   502054    Groundout      NA      NA      NA     NA    NA     NA
## 3   407812    Strikeout      NA      NA      NA     NA    NA     NA
## 4   444379    Groundout      NA      NA      NA     NA    NA     NA
## 5   457705 Hit By Pitch      NA      NA      NA 457705    NA     NA
## 6   501896       Single  457705      NA      NA 501896    NA 457705
##   startOuts endOuts runsFuture runsOnPlay outsInInning venueId batterPos
## 1         0       1          0          0            3      31        3B
## 2         1       2          0          0            3      31        LF
## 3         2       3          0          0            3      31        1B
## 4         0       1          0          0            3      31        1B
## 5         1       1          0          0            3      31        CF
## 6         1       1          0          0            3      31        3B
##   Start1 Start2 Start3 Start_State End1 End2 End3 End_State  Start_RE
## 1      0      0      0   0 0 0   0    0    0    0 0 0 0   1 0.4983772
## 2      0      0      0   0 0 0   1    0    0    0 0 0 0   2 0.2686783
## 3      0      0      0   0 0 0   2    0    0    0 0 0 0   3 0.1063051
## 4      0      0      0   0 0 0   0    0    0    0 0 0 0   1 0.4983772
## 5      0      0      0   0 0 0   1    1    0    0 1 0 0   1 0.2686783
## 6      1      0      0   1 0 0   1    1    0    1 1 0 1   1 0.5122249
##      End_RE  Run_Value team
## 1 0.2686783 -0.2296989  sln
## 2 0.1063051 -0.1623732  sln
## 3 0.0000000 -0.1063051  sln
## 4 0.2686783 -0.2296989  pit
## 5 0.5122249  0.2435466  pit
## 6 1.1967773  0.6845524  pit
```

```
tail(RE_16)
```

```
##        home_team away_team   half                          gameId
## 184147       sln       pit bottom gid_2016_10_02_pitmlb_slnmlb_1
## 184148       sln       pit bottom gid_2016_10_02_pitmlb_slnmlb_1
## 184149       sln       pit    top gid_2016_10_02_pitmlb_slnmlb_1
## 184150       sln       pit    top gid_2016_10_02_pitmlb_slnmlb_1
## 184151       sln       pit    top gid_2016_10_02_pitmlb_slnmlb_1
## 184152       sln       pit    top gid_2016_10_02_pitmlb_slnmlb_1
##        batterName batterId     event start1B start2B start3B  end1B end2B
## 184147  Garcia, G   594824    Single      NA      NA      NA 594824    NA
## 184148       Moss   461235 Strikeout  594824      NA      NA     NA    NA
## 184149  Rogers, J   595386 Groundout      NA      NA      NA     NA    NA
## 184150       Jaso   444379      Walk      NA      NA      NA 444379    NA
```

```
## 184151 Polanco, G   570256 Strikeout  444379      NA      NA 444379    NA
## 184152  McCutchen   457705 Groundout  444379      NA      NA     NA    NA
##        end3B startOuts endOuts runsFuture runsOnPlay outsInInning venueId
## 184147    NA         2       2          0          0            3    2889
## 184148    NA         2       3          0          0            3    2889
## 184149    NA         0       1          0          0            3    2889
## 184150    NA         1       1          0          0            3    2889
## 184151    NA         1       2          0          0            3    2889
## 184152    NA         2       3          0          0            3    2889
##        batterPos Start1 Start2 Start3 Start_State End1 End2 End3 End_State
## 184147        UN      0      0      0   0 0 0   2    1    0    0 1 0 0   2
## 184148        LF      1      0      0   1 0 0   2    0    0    0 0 0 0   3
## 184149        UN      0      0      0   0 0 0   0    0    0    0 0 0 0   1
## 184150        1B      0      0      0   0 0 0   1    1    0    0 1 0 0   1
## 184151        RF      1      0      0   1 0 0   1    1    0    0 1 0 0   2
## 184152        CF      1      0      0   1 0 0   2    0    0    0 0 0 0   3
##         Start_RE    End_RE  Run_Value team
## 184147 0.1063051 0.2205392  0.1142341  sln
## 184148 0.2205392 0.0000000 -0.2205392  sln
## 184149 0.4983772 0.2686783 -0.2296989  pit
## 184150 0.2686783 0.5122249  0.2435466  pit
## 184151 0.5122249 0.2205392 -0.2916857  pit
## 184152 0.2205392 0.0000000 -0.2205392  pit
```

```
# We use .groupby to calculate the sum of runs values
# for each team in each season and then merge to two frames to make
# comparisons

REteam_17 <- RE_17 %>% group_by(team)%>%
  dplyr::summarise(Run_Value = sum(Run_Value))%>%
  rename(RV17 = Run_Value)


REteam_16 <- RE_16 %>% group_by(team)%>%
  dplyr::summarise(Run_Value = sum(Run_Value))%>%
  rename(RV16 = Run_Value)

REteam1617 <- merge(REteam_16,REteam_17, by='team')
head(REteam1617)
```

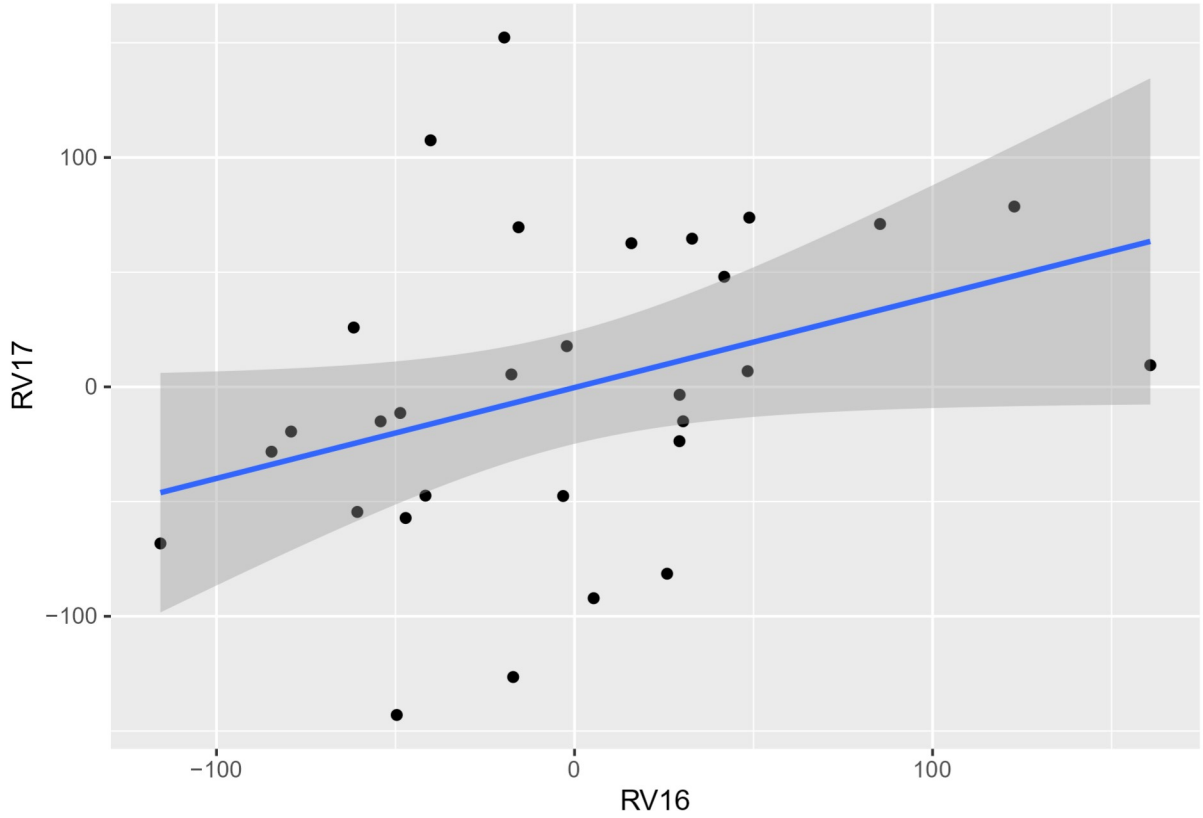
```
##   team       RV16       RV17
## 1  ana  -3.184209 -47.579782
## 2  ari  15.870968  62.616716
## 3  atl -84.633901 -28.259905
## 4  bal  29.297940 -23.678031
```

```
## 5  bos 160.812546   9.445347
## 6  cha -41.643638 -47.416033
```

```
tail(REteam1617)
```

```
##    team      RV16        RV17
## 25  sfn -17.14201 -126.489153
## 26  sln  48.33689    6.838344
## 27  tba -47.16798  -57.161656
## 28  tex  41.81255   48.002092
## 29  tor  25.85149  -81.456403
## 30  was  32.82553   64.616716
```

```
ggplot(data = REteam1617)+
  geom_point(aes(x= RV16, y = RV17),color= 'black')+
  geom_smooth(aes(x= RV16, y = RV17),method = "lm",formula = y ~ x)
```



We find a modest correlation in Runs Values between the two seasons.

```
cor(REteam1617$RV16, REteam1617$RV17)
```

```
## [1] 0.3517082
```

Self Test

Compare the correlation in Runs Values for the seasons 2015 and 2017. Do you find the same pattern? Why or why not?

4. Comparing Run Values, Win Percentage, and Salary Data

Runs values can be considered both as a factor that explains winning - teams that achieve a higher Runs Value are performing better than expected, and hence are more likely to win - or as a predictor of winning in the future. To the extent that the market for players is efficient in an economic sense, player salaries will also reflect both current and future performance. So we can use our data to make these comparisons.

We take the salary data for 2016 from the Lahman database. We download a file which includes the regular season win percentage for each team in 2016 and 2017 and the total paid out in salaries in 2016.

```
wpcsal1617 = read_excel("wpcsal2016-17.xlsx")
head(wpcsal1617)
```

```
## # A tibble: 6 x 5
##   team  wpc2016 wpc2017   Sal2016 lahteamname
##   <chr>   <dbl>   <dbl>     <dbl> <chr>
## 1 ari     0.426   0.574  87439063 ARI
## 2 atl     0.422   0.444  68498291 ATL
## 3 bal     0.549   0.463 161863456 BAL
## 4 bos     0.574   0.574 188545761 BOS
## 5 cha     0.481   0.414 112998667 CHA
## 6 chn     0.636   0.568 154067668 CHN
```

Self Test

The win percentages for each can be derived directly from RE_16 and RE_17. To do this, .groupby to sum the runs scored by each team in each game and denote the winner as the team with more runs (1 if the winner, 0 otherwise) and then use .groupby to find the mean of wins - that is win percentage. Check that your answer matches the data for win percentage downloaded here.

```
# Here we merge the Runs Value df with the Win Percentage df
REteam1617 <- merge(REteam1617,wpcsal1617, by = 'team')
head(REteam1617)
```

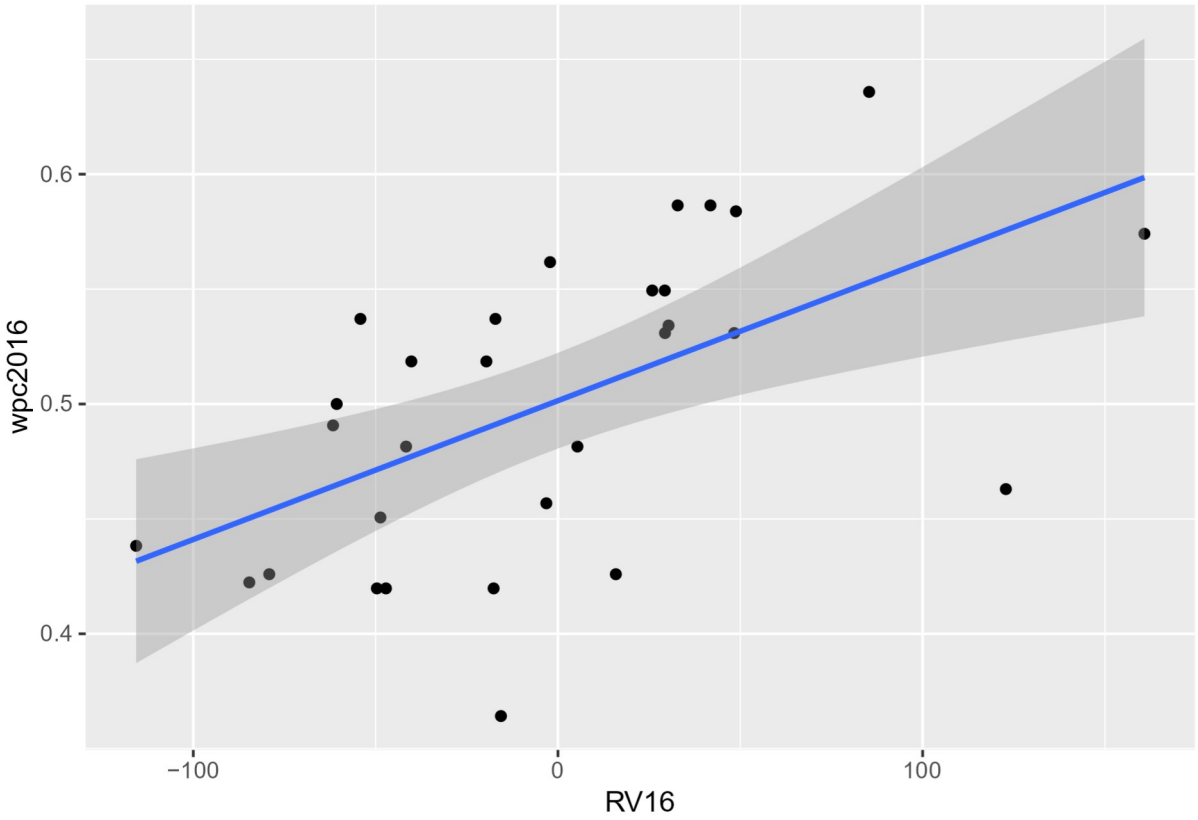
```
##   team       RV16       RV17   wpc2016   wpc2017   Sal2016 lahteamname
## 1  ana  -3.184209 -47.579782 0.4567901 0.4938272 137251333         LAA
## 2  ari  15.870968  62.616716 0.4259259 0.5740741  87439063         ARI
## 3  atl -84.633901 -28.259905 0.4223602 0.4444444  68498291         ATL
## 4  bal  29.297940 -23.678031 0.5493827 0.4629630 161863456         BAL
```

```
## 5   bos 160.812546   9.445347 0.5740741 0.5740741 188545761          BOS
## 6   cha -41.643638 -47.416033 0.4814815 0.4135802 112998667          CHA
```

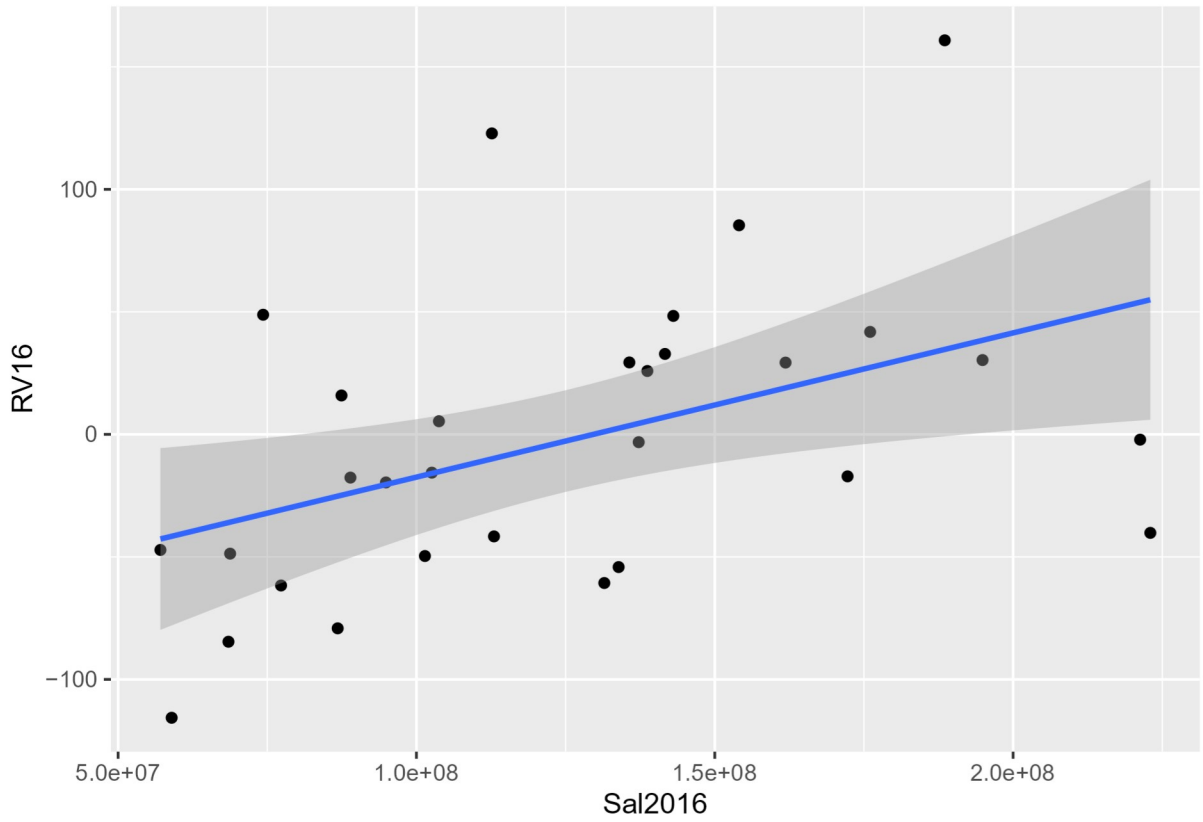
Scatterplots

The extent of correlation among the variables can be examined using scatterplots for each pair of variables

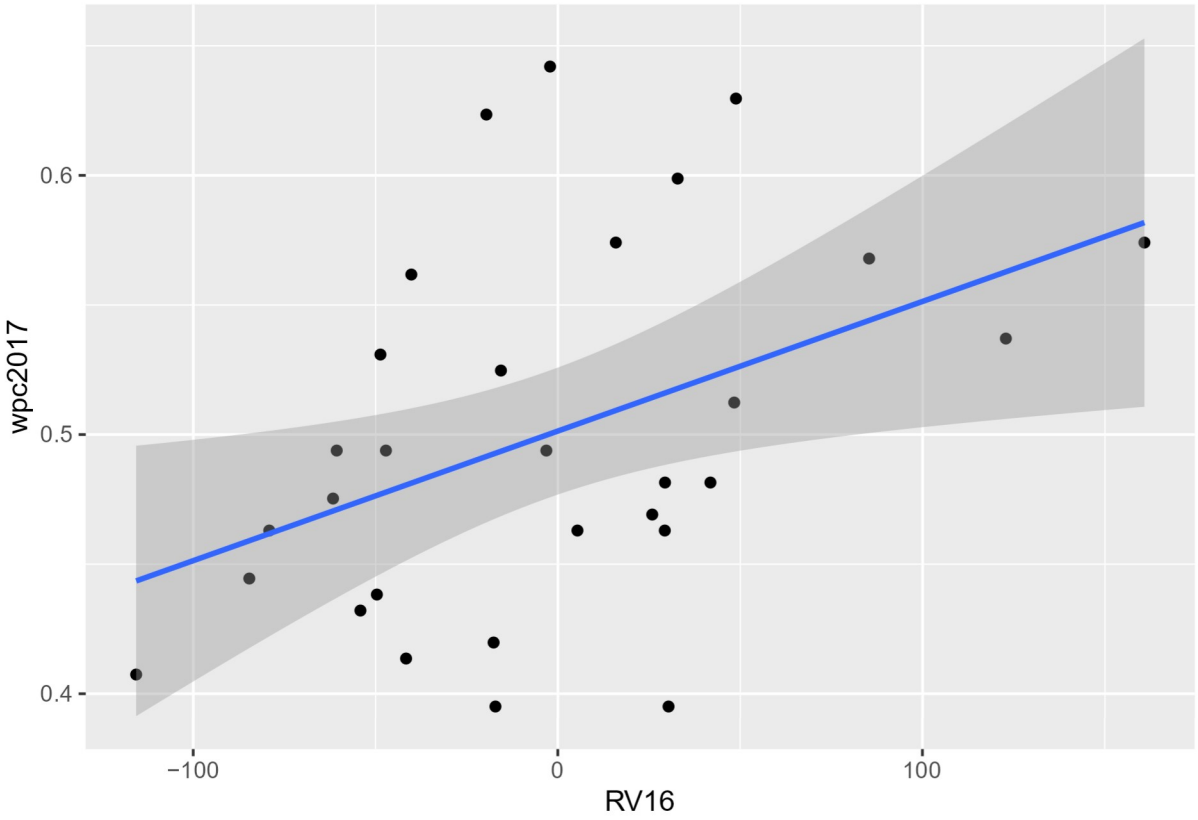
```
ggplot(data = REteam1617)+
  geom_point(aes(x= RV16, y = wpc2016),color= 'black')+
  geom_smooth(aes(x= RV16, y = wpc2016),method = "lm",formula = y ~ x)
```



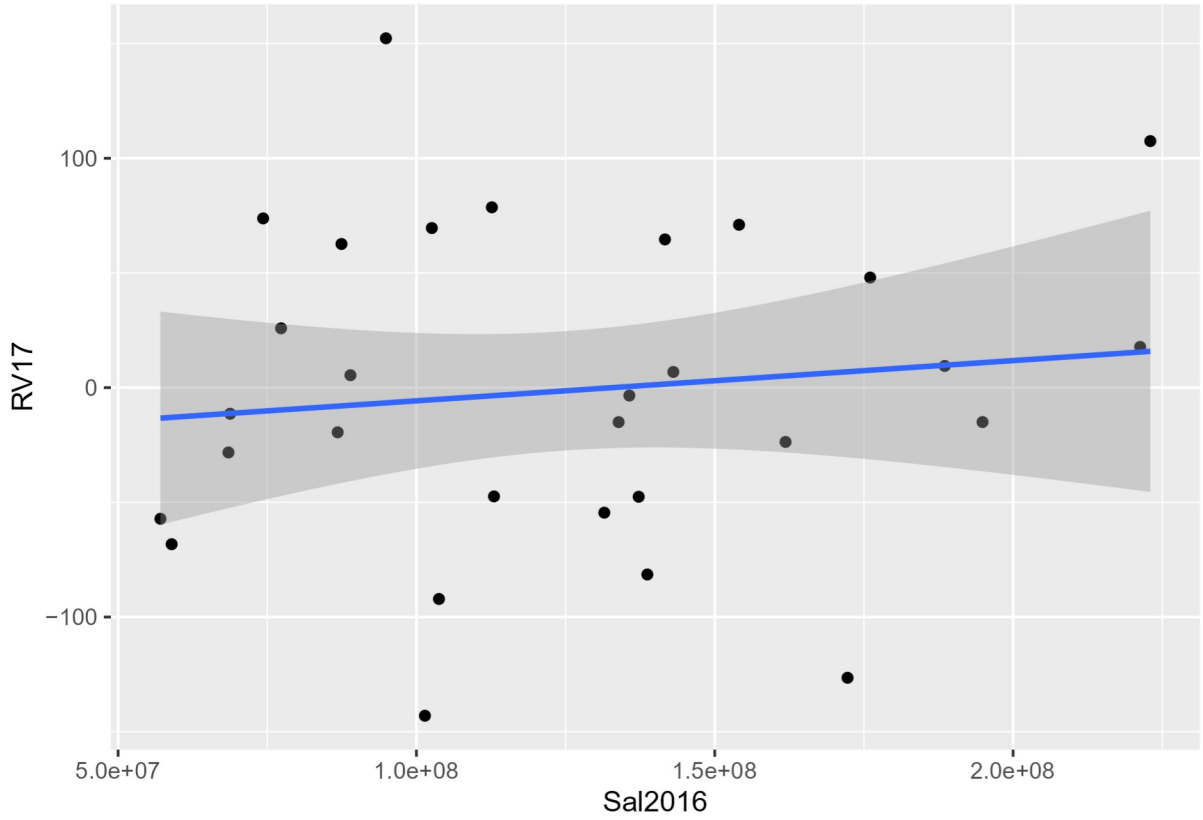

```
ggplot(data = REteam1617)+
  geom_point(aes(x= Sal2016, y = RV16),color= 'black')+
  geom_smooth(aes(x= Sal2016, y = RV16),method = "lm",formula = y ~ x)
```




```
ggplot(data = REteam1617)+
  geom_point(aes(x= RV16, y = wpc2017),color= 'black')+
  geom_smooth(aes(x= RV16, y = wpc2017),method = "lm",formula = y ~ x)
```




```
ggplot(data = REteam1617)+
  geom_point(aes(x= Sal2016, y = RV17),color= 'black')+
  geom_smooth(aes(x= Sal2016, y = RV17),method = "lm",formula = y ~ x)
```



Correlations

We can generate a matrix of correlation coefficients using cor()

```
cor(REteam1617 %>% select_if(is.numeric))
```

```
##              RV16      RV17   wpc2016   wpc2017   Sal2016
## RV16    1.0000000 0.3517082 0.5590354 0.4288187 0.4497020
## RV17    0.3517082 1.0000000 0.2491733 0.7167627 0.1189887
## wpc2016 0.5590354 0.2491733 1.0000000 0.3444263 0.6326956
## wpc2017 0.4288187 0.7167627 0.3444263 1.0000000 0.1552630
## Sal2016 0.4497020 0.1189887 0.6326956 0.1552630 1.0000000
```

Conclusions

There is high correlation between Runs Value and Win Percentage, especially in 2017. There is also a fairly high level of correlation between Runs Value in 2016 and Win Percentage in 2017 - it appears predictive to some extent. Runs Value is also fairly highly correlated with salary in a given year, but salaries in 2016 are not very closely correlated with Win Percentage in 2017.

There are several possible explanations for this. Salaries are not as correlated with productivity (Runs Value) as one might expect in an open market, since rookies are not able to sell their

services competitively and even 5 year veterans are constrained by arbitration (recall our analysis of salaries from Hakes and Sauer).

It is also possible that salaries are more about rewarding past performance than future performance. Your opinion on this depends on how you think the markets work.

It may also be the case that salary data is partial and incomplete.

Runs Value is a very good measure of player productivity, which improves upon traditional statistics by taking account of context. It's no accident that statistical analysis in baseball is more advanced than in any other sport - the natural measurement of performance as the product of a series of discrete homogeneous events (at-bats) is unparalleled in sports. Baseball analysts have gone further and made adjustments to these statistics to generate a single, intuitive measure of productivity such as Wins Above Replacement (WAR). There is debate about how to make these adjustments and different versions of WAR exist. In many cases WAR is calculated on the basis of season average player stats adjusted for Run Expectancy - a method which then loses the context dependent values we have derived here. In any case, equipped with the capacity to produce Runs Value data, you should now be able to go on and produce your own version of WAR, if you so wish. An example of how to do this is given by Baumer, Jensen, and Matthews (2015) "openWAR: An open source system for evaluating overall player performance in major league baseball", which you can download online for free.

Throughout this course we have focused on the relationship between productivity, performance and valuation. We have focused mainly on using data to *explain* the relationships, but we have also seen that it can have *predictive* value. This is the underlying story behind Moneyball, but is is also the story of sports analytics in general. In the next course we will focus in more detail on prediction and forecasting.