Lab 2

Nate Beebe (Worked with Matthew Raitano and Zack Barnes)

Due on 02/21 at 11:59 pm

Question 1 The 2014 and 2015 Royals surprised a lot of people when they seemingly came out of nowhere with back-to-back world series including a title in 2015. In this problem and in the next problem we will investigate aspects of weirdness surrounding these Royals teams. See this Foolish Baseball video, this Keith Law article, and this article about the failure of projection systems for background. In this problem you will construct a relevant data set for analysis with the ultimate goal of describing just how unique these Royals were. Do the following:

- Construct a data frame which includes the following variables from the `Teams` data frame in the `Lahman` package: `yearID`, `teamID`, `AB`, `SO`, `H`, `HR`, `R`, `RA`, `W`, and `L`. Only keep seasons dating back to 1990, and remove the 1994, 1995, and 2020 seasons.

```
library(Lahman)
library(tidyverse)
Set = Teams %>%
  filter(yearID >= 1990) %>%
  filter(yearID != 2020) %>%
  filter(yearID != 1995) %>%
  filter(yearID != 1994) %>%
  select(yearID, teamID, AB, SO, H, HR, R, RA, W, L, franchID)

Set_Save = Teams %>%
  filter(yearID >= 1990) %>%
  filter(yearID != 2020) %>%
  filter(yearID != 1995) %>%
  filter(yearID != 1994) %>%
  group_by(yearID)
head(Set)
```

```
##   yearID teamID   AB   SO    H  HR   R  RA  W  L franchID
## 1   1990    BAL 5410  962 1328 132 669 698 76 85      BAL
## 2   1990    BOS 5516  795 1502 106 699 664 88 74      BOS
## 3   1990    CAL 5570 1000 1448 147 690 706 80 82      ANA
## 4   1990    CHA 5402  903 1393 106 682 633 94 68      CHW
## 5   1990    CLE 5485  836 1465 110 732 737 77 85      CLE
## 6   1990    DET 5479  952 1418 172 750 754 79 83      DET
```

I found it easier to work with "franchID" when merging the data from each season, so that's why it is also included here.

- Run the code below to scrape data from baseball reference, and only keep seasons dating back to 1990, and remove the 1994, 1995, and 2020 seasons.

```r
bwar_bat = readr::read_csv("https://www.baseball-reference.com/data/war_daily_bat.txt", na = "NULL")
bwar_pit = readr::read_csv("https://www.baseball-reference.com/data/war_daily_pitch.txt", na = "NULL")

#Only keeping instructed seasons
bat_war = bwar_bat %>%
  filter(year_ID != 2020) %>%
  filter(year_ID != 1995) %>%
  filter(year_ID != 1994) %>%
  filter(year_ID >= 1990)
pit_war = bwar_pit %>%
  filter(year_ID != 2020) %>%
  filter(year_ID != 1995) %>%
  filter(year_ID != 1994) %>%
  filter(year_ID >= 1990)
```

- Obtain total team defensive WAR `WAR_def`, bullpen WAR, and base running runs `runs_br` for each year and add these quantities to the data frame that you previously constructed from the `Teams` data frame. Call these variables, respectively, `dWAR`, `penWAR`, `BRruns`.

```r
D_R_WAR = bat_war %>%
  rename(franchID = team_ID) %>%
  group_by(year_ID, franchID) %>%
  summarise(dWAR = sum(WAR_def, na.rm = T), BRruns = sum(runs_br, na.rm = T)) %>%
  mutate(franchID = ifelse(franchID == "LAA", "ANA", franchID)) %>%
  mutate(franchID = ifelse(franchID == "MON", "WSN", franchID)) %>%
  mutate(franchID = ifelse(franchID == "TBR", "TBD", franchID)) %>%
  mutate(franchID = ifelse(franchID == "MIA", "FLA", franchID)) %>%
  mutate(franchID = ifelse(franchID == "CAL", "ANA", franchID))


Pen_WAR = pit_war %>%
  rename(franchID = team_ID) %>%
  #For this exercise, I am defining a "bullpen pitcher" as someone who has more outs recorded in relief
  filter(IPouts_relief > IPouts_start) %>%
  group_by(year_ID, franchID) %>%
  summarise(penWAR = sum(WAR, na.rm = T)) %>%
  mutate(franchID = ifelse(franchID == "LAA", "ANA", franchID)) %>%
  mutate(franchID = ifelse(franchID == "MON", "WSN", franchID)) %>%
  mutate(franchID = ifelse(franchID == "TBR", "TBD", franchID)) %>%
  mutate(franchID = ifelse(franchID == "MIA", "FLA", franchID)) %>%
  mutate(franchID = ifelse(franchID == "CAL", "ANA", franchID))

WARSet = left_join(Pen_WAR, D_R_WAR) %>%
  rename(yearID = year_ID)


NewSet = Set %>%
  left_join(WARSet) %>%
  mutate(BRruns) %>%
  select(!franchID)

head(NewSet)
```

```
##   yearID teamID   AB   SO    H  HR   R  RA  W  L penWAR  dWAR BRruns
## 1   1990    BAL 5410  962 1328 132 669 698 76 85   3.70  2.61  -4.60
## 2   1990    BOS 5516  795 1502 106 699 664 88 74   1.94 -4.75 -21.64
## 3   1990    CAL 5570 1000 1448 147 690 706 80 82   4.15 -5.86  -4.46
## 4   1990    CHA 5402  903 1393 106 682 633 94 68   7.40  5.40   6.39
## 5   1990    CLE 5485  836 1465 110 732 737 77 85   3.59  0.48  -0.84
## 6   1990    DET 5479  952 1418 172 750 754 79 83   9.29 -3.24  -4.12
```

- The 2014-2015 Royals were known for elite base running, an elite bullpen, and elite defense. They were also known for not striking out and not hitting home runs. Add the following scaled variables separately for each season to the data frame that you constructed in the previous step:
  - `scaledSO = scale(SO/AB)`,
  - `scaledBA = scale(H/AB)`,
  - `scaledABpHR = scale(AB/HR)`,
  - `scaledpenWAR = scale(penWAR)`,
  - `scaleddWAR = scale(dWAR)`,
  - `scaledBRruns = scale(BRruns)`

```
ScaledSet = NewSet %>%
  group_by(yearID) %>%
  mutate(scaledSO = scale(SO/AB),
         scaledBA = scale(H/AB),
         scaledAPpHR = scale(AB/HR),
         scaledpenWAR = scale(penWAR),
         scaleddWAR = scale(dWAR),
         scaledBRruns = scale(BRruns))
head(ScaledSet)
```

```
## # A tibble: 6 x 19
## # Groups:   yearID [1]
##   yearID teamID    AB    SO     H    HR     R    RA     W     L penWAR  dWAR
##    <dbl> <fct>  <int> <int> <int> <int> <int> <int> <int> <int>  <dbl> <dbl>
## 1   1990 BAL     5410   962  1328   132   669   698    76    85   3.7   2.61
## 2   1990 BOS     5516   795  1502   106   699   664    88    74   1.94 -4.75
## 3   1990 CAL     5570  1000  1448   147   690   706    80    82   4.15 -5.86
## 4   1990 CHA     5402   903  1393   106   682   633    94    68   7.4   5.4
## 5   1990 CLE     5485   836  1465   110   732   737    77    85   3.59  0.48
## 6   1990 DET     5479   952  1418   172   750   754    79    83   9.29 -3.24
## # i 7 more variables: BRruns <dbl>, scaledSO <dbl[,1]>, scaledBA <dbl[,1]>,
## #   scaledAPpHR <dbl[,1]>, scaledpenWAR <dbl[,1]>, scaleddWAR <dbl[,1]>,
## #   scaledBRruns <dbl[,1]>
```

- Compute and add winning percentage `Wpct` to your data frame. Use an equation in your notes and linear regression to compute the optimal k so that `Wpct` is well-explained by `Wpytk` $= R^k/(R^k + RA^k)$. Add `Wpytk` and `residuals_pytk = Wpct - Wpytk` to your data frame.

```
RoyalsSet = ScaledSet %>%
  mutate(Wpct = round(W/(W + L), 3))

Royals_aug = RoyalsSet %>%
mutate(logWratio = log(W / L),
       logRratio = log(R / RA))
```

```r
RoyalpyFit = lm(logWratio ~ 0 + logRratio, data = Royals_aug)
RoyalpyFit$coefficients
```

```
## logRratio
##  1.841277
```

```r
k_Royal = 1.841277
RoyalPythagSet = Royals_aug %>%
  mutate(Wpytk = R^k_Royal/(R^k_Royal +RA^k_Royal)) %>%
  mutate(residuals_pytk = Wpct - Wpytk)

FinalSet = RoyalsSet %>%
  left_join(RoyalPythagSet) %>%
  select(-c(logWratio, logRratio))
head(FinalSet)
```

```
## # A tibble: 6 x 22
## # Groups:   yearID [1]
##   yearID teamID    AB    SO     H    HR     R    RA     W     L penWAR  dWAR
##    <dbl> <fct>  <int> <int> <int> <int> <int> <int> <int> <int>  <dbl> <dbl>
## 1   1990 BAL     5410   962  1328   132   669   698    76    85   3.7   2.61
## 2   1990 BOS     5516   795  1502   106   699   664    88    74   1.94 -4.75
## 3   1990 CAL     5570  1000  1448   147   690   706    80    82   4.15 -5.86
## 4   1990 CHA     5402   903  1393   106   682   633    94    68   7.4   5.4
## 5   1990 CLE     5485   836  1465   110   732   737    77    85   3.59  0.48
## 6   1990 DET     5479   952  1418   172   750   754    79    83   9.29 -3.24
## # i 10 more variables: BRruns <dbl>, scaledSO <dbl[,1]>, scaledBA <dbl[,1]>,
## #   scaledAPpHR <dbl[,1]>, scaledpenWAR <dbl[,1]>, scaleddWAR <dbl[,1]>,
## #   scaledBRruns <dbl[,1]>, Wpct <dbl>, Wpytk <dbl>, residuals_pytk <dbl>
```

- Display the rows of this data frame corresponding to the 2014-2015 Royals seasons.

```r
Royals = FinalSet %>%
  filter(yearID == 2014 | yearID == 2015) %>%
  filter(teamID == "KCA")

Royals
```

```
## # A tibble: 2 x 22
## # Groups:   yearID [2]
##   yearID teamID    AB    SO     H    HR     R    RA     W     L penWAR  dWAR
##    <dbl> <fct>  <int> <int> <int> <int> <int> <int> <int> <int>  <dbl> <dbl>
## 1   2014 KCA     5545   985  1456    95   651   624    89    73   8.17  4.95
## 2   2015 KCA     5575   973  1497   139   724   641    95    67   9.99  5.22
## # i 10 more variables: BRruns <dbl>, scaledSO <dbl[,1]>, scaledBA <dbl[,1]>,
## #   scaledAPpHR <dbl[,1]>, scaledpenWAR <dbl[,1]>, scaleddWAR <dbl[,1]>,
## #   scaledBRruns <dbl[,1]>, Wpct <dbl>, Wpytk <dbl>, residuals_pytk <dbl>
```

Question 2 In this problem we will perform analyses that investigate strengths and peculiarities of the 2014-2015 Royals. Do the following:

- Fit and analyze a regression model of `residuals_pytk` on `penWAR`. Determine how many wins one would expect the Royals to obtain above their Pythagorean expectations on the basis of their bullpen.

```
PenModel = lm(residuals_pytk ~ penWAR, FinalSet)
summary(PenModel)
```

```
##
## Call:
## lm(formula = residuals_pytk ~ penWAR, data = FinalSet)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.084613 -0.016752  0.000356  0.016183  0.086652
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) -0.004569   0.001340  -3.410 0.000678 ***
## penWAR       0.001024   0.000264   3.878 0.000113 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.02455 on 910 degrees of freedom
## Multiple R-squared:  0.01626,    Adjusted R-squared:  0.01518
## F-statistic: 15.04 on 1 and 910 DF,  p-value: 0.0001128
```

```
(0.001116*8.17-0.005293)*162
```

```
## [1] 0.6196046
```

```
(0.001116*9.99-0.005293)*162
```

```
## [1] 0.9486461
```

Based on this model, the 2014 Royals would be expected to outperform their Pythagorean expectancy by 0.62 wins. The 2015 Royals would be expected to outperform their Pythagorean expectancy by 0.95 wins.

- Total bullpen WAR is just one aspect of what made the 2014-2015 Royals what they were. We will now use k-means clustering implemented via the kmeans function to determine whether or not teams similar to the 2014-2015 Royals beat their Pythagorean expectations. Do the following with the number of clusters ranging from $k = 30, ..., 50$: 1) run kmeans on a data set containing the six scaled variables that you previously constructed with k centers; 2) add the cluster assignments to the original dataset; 3) extract the average of `residuals_pytk` for the clusters containing the 2014 or 2015 Royals after removing the Royals from consideration. When finished, compute the average `residuals_pytk` value for the 2014 and 2015 Royals and then multiply this number by 162. This is the number of expected wins above/below their Pythagorean expectations that similar teams produced. Report this value and compare it with the 2014-2015 Royals.

```
KSet = FinalSet %>%
  ungroup(yearID) %>%
  select(scaledSO:scaledBRruns)
```

```r
for (i in 30:50) {
  k_means = kmeans(centers = i, x = KSet)

  clusteredSet = FinalSet %>%
    ungroup(yearID) %>%
    mutate(cluster = k_means$cluster)

  RoyalsClustered = clusteredSet %>%
    filter(teamID == "KCA", yearID == 2014 | yearID == 2015)

  clusters2014 = clusteredSet %>%
    filter(cluster == RoyalsClustered$cluster[1], teamID != "KCA") %>%
    summarise(residuals_pytk = sum(residuals_pytk)/n())

  clusters2015 = clusteredSet %>%
    filter(cluster == RoyalsClustered$cluster[2], teamID != "KCA") %>%
    summarise(residuals_pytk = sum(residuals_pytk)/n())


}


clusters2014$residuals_pytk*162
```

```
## [1] 0.7755481
```

```r
clusters2015$residuals_pytk*162
```

```
## [1] 0.7755481
```

```r
Royals$W[1] - Royals$Wpytk[1]*162
```

```
## [1] 4.842796
```

```r
Royals$W[2] - Royals$Wpytk[2]*162
```

```
## [1] 4.957846
```

Both the 2014 and the 2015 Royals were placed in the same cluster. Teams similar to both those Royals teams produced typically 0.7755 wins more than their Pythagorean win percentage. The 2014 Royals actually produced 4.84 more wins than their Pythagorean expectancy. The 2015 Royals outperformed this expectation by 4.96 wins.

- Add the `OPSscale` and `WHIPscale` variables that you computed in Question 1 of Lab 1 to the data frame. Run a regression with `Wpct` as the response variable and all eight scaled variables as predictors (you can drop terms if you want to). Does this model over/under estimate the success of the 2014-2015 Royals?

```r
PredictSt = FinalSet %>%
  ungroup(yearID) %>%
  mutate(BB = Set_Save$BB, HBP = Set_Save$HBP, SF = Set_Save$SF, X2B = Set_Save$X2B, X3B = Set_Save$X3B
  mutate(X1B = H - X2B - X3B - HR) %>%
  mutate(OBP = (H + BB + HBP)/(AB + BB + HBP + SF)) %>%
    mutate(SLG = (X1B + 2*X2B + 3*X3B + 4*HR)/AB) %>%
    mutate(OPS = OBP + SLG) %>%
    mutate(WHIP = 3*(HA + BBA)/IPouts)%>%
  mutate(IP = round(IPouts/3 ,2)) %>%
  group_by(yearID) %>%
  mutate(avgOBP = round((sum(H) + sum(BB) + sum(HBP))/ (sum(AB) + sum(HBP) + sum(BB) + sum(SF)),3),
         avgSLG = round((sum(X1B)+2*sum(X2B)+3*sum(X3B)+4*sum(HR))/sum(AB), 3),
         avgOPS = avgSLG + avgOBP,
         avgWHIP = round((sum(BB)+sum(H))/sum(IP),3)) %>%
  mutate(OPSscale = OPS/avgOPS,
         WHIPscale = avgWHIP/WHIP)
finalmodel = lm(Wpct ~ OPSscale + WHIPscale + scaledSO + scaledBA + scaledAPpHR + scaledpenWAR + scaled

FinalRoyalsSet = PredictSt %>%
  filter(yearID == 2014 | yearID == 2015, teamID == "KCA")

Royals14 = 162 * (finalmodel$coefficients[1] + finalmodel$coefficients[2]*FinalRoyalsSet$OPSscale[1]+ fi


Royals15 = 162 * (finalmodel$coefficients[1] + finalmodel$coefficients[2]*FinalRoyalsSet$OPSscale[2]+ fi


FinalRoyalsSet$W[1] - Royals14
```

```
## (Intercept)
##    4.972984
```

```r
FinalRoyalsSet$W[2] - Royals15
```

```
## (Intercept)
##    6.160265
```

This model underestimates the Royals. In both years the Royals actually won more than expected by this model (4.97 more wins in 2014 and 6.16 more wins in 2015).

Question 3 Do the following:

- Select a period of your choice (at least 20 years) and fit the Pythagorean formula model (after finding the optimal exponent) to the run-differential, win-loss data.

```r
MySet = Teams %>%
  filter(yearID >= 2000) %>%
  filter(yearID <= 2019) %>%
  mutate(Wpct = round(W/(W+L), 3))
dat_aug = MySet %>%
mutate(logWratio = log(W / L),
       logRratio = log(R / RA))
```

```r
pyFit = lm(logWratio ~ 0 + logRratio, data = dat_aug)
k = pyFit$coefficients
PythagSet = dat_aug %>%
  mutate(Wpct_pytk = R^k/(R^k +RA^k)) %>%
  select(Wpct, Wpct_pytk)
PythagModel = lm(Wpct ~ 0 + Wpct_pytk, data = PythagSet)
summary(PythagModel)
```

```
##
## Call:
## lm(formula = Wpct ~ 0 + Wpct_pytk, data = PythagSet)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.067671 -0.017276 -0.001122  0.015378  0.081605
##
## Coefficients:
##           Estimate Std. Error t value Pr(>|t|)
## Wpct_pytk 0.999071   0.002025   493.4   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.02505 on 599 degrees of freedom
## Multiple R-squared:  0.9975, Adjusted R-squared:  0.9975
## F-statistic: 2.434e+05 on 1 and 599 DF,  p-value: < 2.2e-16
```

- On the basis of your fit in the previous part and the list of managers obtained from Retrosheet, compile a top 10 list of managers who most overperformed their Pythagorean winning percentage and a top 10 list of managers who most underperformed their Pythagorean winning percentage.

In class, Dr. Eck said that it may be easier to do this exercise using the data from Lahman rather than finding and pulling all the data from Retrosheet, so that's what I did. For managers that didn't manage a full season, I assigned the runs and runs allowed that were proportional to the time that they did manage to them. For example, if two managers each managed 81 games, and the team scored 200 runs and allowed 150, both managers would have 100 runs scored and 75 runs allowed contributed to them.

```r
Runs = Teams %>%
  select(teamID, yearID, R, RA)
List = Managers %>%
  filter(yearID>=2000) %>%
  filter(yearID<=2019) %>%
  left_join(People) %>%
  left_join(Runs) %>%
  select(nameFirst, nameLast, G, W, L, R, RA, yearID, playerID, teamID, inseason) %>%
  mutate(Rportion = round(R * G/162,2),
         RAportion = round(RA * G/162,2)) %>%
  select(nameFirst, nameLast, G, W, L, R, Rportion, RA, RAportion, yearID, playerID, teamID) %>%
  group_by(playerID) %>%
  summarise(R = sum(Rportion), RA = sum(RAportion), G = sum(G), W = sum(W), L = sum(L)) %>%
  mutate(pythagpct = round((R^k) / (R^k + RA^k),3),
         Wpct = round(W/G, 3)) %>%
  mutate(diff = Wpct - pythagpct) %>%
```

```r
  group_by(playerID) %>%
  summarise(difference = sum(diff)) %>%
  left_join(People) %>%
  select(nameFirst, nameLast, difference) %>%
  arrange(desc(difference))


#Ten best
List[1:10,]
```

```
## # A tibble: 10 x 3
##    nameFirst nameLast  difference
##    <chr>     <chr>          <dbl>
##  1 Ray       Knight         0.612
##  2 Terry     Steinbach      0.538
##  3 Gary      Tuck           0.509
##  4 John      Tamargo        0.466
##  5 Ted       Simmons        0.439
##  6 Dino      Ebel           0.407
##  7 Pat       Corrales       0.288
##  8 Bill      Russell        0.231
##  9 Chris     Speier         0.226
## 10 Tim       Bogar          0.225
```

```r
#Ten worst
List[169:160,]
```

```
## # A tibble: 10 x 3
##    nameFirst nameLast  difference
##    <chr>     <chr>          <dbl>
##  1 Josh      Bard          -0.609
##  2 Dick      Scott         -0.538
##  3 Rene      Lachemann     -0.466
##  4 Jamie     Quirk         -0.425
##  5 Mark      Parent        -0.413
##  6 Rod       Barajas       -0.308
##  7 Harold    Baines        -0.269
##  8 Rob       Thomson       -0.205
##  9 Gary      Varsho        -0.199
## 10 Joe       Nossek        -0.196
```

Question 4 The first question in Section 1.4.3 of Analyzing Baseball Data with R. Your answer to this question must include the code to obtain the answer. You cannot copy the answer directly from the book.

```r
data = read.csv("C:/Users/STP/Desktop/stat430/stat430sp25/all1998.csv")
McGwireID = People %>%
  filter(nameFirst == "Mark" & nameLast == "McGwire") %>%
  pull(retroID)

SosaID = People %>%
  filter(nameFirst == "Sammy" & nameLast == "Sosa") %>%
  pull(retroID)
```

```r
MMHomers = data %>%
  filter(BAT_ID == McGwireID) %>%
  filter(BASE1_RUN_ID != "" | BASE2_RUN_ID != "" | BASE3_RUN_ID != "") %>%
  filter(EVENT_CD != 15) %>%
  filter(EVENT_CD != 16) %>%
  #Note of 317 events total
  filter(EVENT_CD == 23)
  #37 home runs

SosaHomers = data %>%
  filter(BAT_ID == SosaID) %>%
  filter(BASE1_RUN_ID != "" | BASE2_RUN_ID != "" | BASE3_RUN_ID != "") %>%
  filter(EVENT_CD != 15) %>%
  filter(EVENT_CD != 16) %>%
  #368 opportunities
  filter(EVENT_CD == 23)
  #29 home runs
```

I opted to include non-intentional walks in my count as opposed to the book because unlike being hit by a pitch or intentionally walked, the batter does have a choice in that outcome.

My conclusion is the same as the book's. McGwire had more home runs total, and he had less opportunities to do so, making him more successful at hitting home runs with runners on base in the 1998 season.