Lesson 9 Boardsheet

Kevin Cummiskey

2/2/2020

Review

During the last two lessons, we discussed three models for predicting the number of wins a team should expect to have in a season. Here are three models:

$$Wpct = \beta_0 + \beta_1 RD + \epsilon \tag{1}$$
$$Wpct = \frac{R^2}{R^2 + RA^2} + \epsilon \tag{2}$$
$$Wpct = \frac{R^k}{R^k + RA^k} + \epsilon \tag{3}$$

where $Wpct$ is Win Percentage, R is Runs Scored, RA is Runs Allowed, and ϵ is the random error. Recall that we fit Model 1 to the 1997-2001 seasons.

Briefly discuss the strengths/limitations of each model.

How would you assess which model is the best? Please be specific.

Model 2 Pythagorean Formula (Bill James)

First, let's create a function to calculate the expected wins under a Pythagorean Formula>

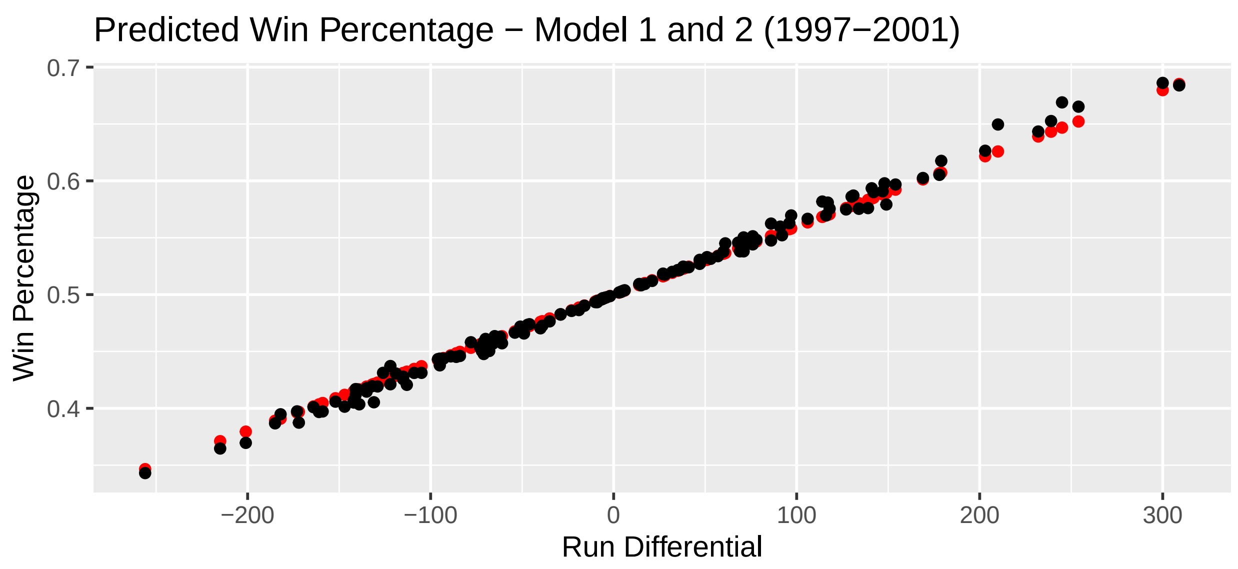
```r
#function to calculate expected wins
#using pythagorean formula
#arguments:
# R - runs scored
# RA - runs allowed
# k - exponent
# values:
# expected win percentage
pyt_wins <- function(R, RA, k = 2){
  return(R^k/(R^k + RA^k))
}
```

Using Model 2, let's calculate the expected number of wins for each team (1997-2001).

```r
library(tidyverse)
library(Lahman)

my_teams = Teams %>%
  filter(yearID >= 1997, yearID <= 2001) %>%
  mutate(RD = R - RA,
         Wpct = W/(W+L),
         Wpct.pyt2 = pyt_wins(R,RA,2))
```

Next, let's compare graphically the predictions from Model 1 and 2.

```r
library(broom)

#get Model 1 predicted win percentages
lin.fit <- lm(Wpct ~ RD, data = my_teams)
my_teams <- augment(lin.fit, data = my_teams)

#plot Model 1 and Model 2 predictions
my_teams %>%
  ggplot(aes(x = RD, y = .fitted)) +
  geom_point(col = "red") +
  geom_point(aes(x = RD, y = Wpct.pyt2)) +
  labs(x = "Run Differential",
       y = "Win Percentage",
       title = "Predicted Win Percentage - Model 1 and 2 (1997-2001)")
```



Briefly discuss how the models differ.

Next, let's compare the root mean square error (RMSE). Write an equation for the RMSE.

```r
# Model 1 RMSE
sqrt(mean(my_teams$.resid^2))
```

```
## [1] 0.0228099
```

```r
# Note this is very close to the Residual Standard Error
# which you could also use
summary(lin.fit)$sigma
```

```
## [1] 0.02296561
```

```r
# Model 2 RMSE
# calculate residuals
my_teams = my_teams %>%
  mutate(.resid.pyt2 = Wpct - Wpct.pyt2)
# calculate RMSE for Model 2
sqrt(mean(my_teams$.resid.pyt2^2))
```

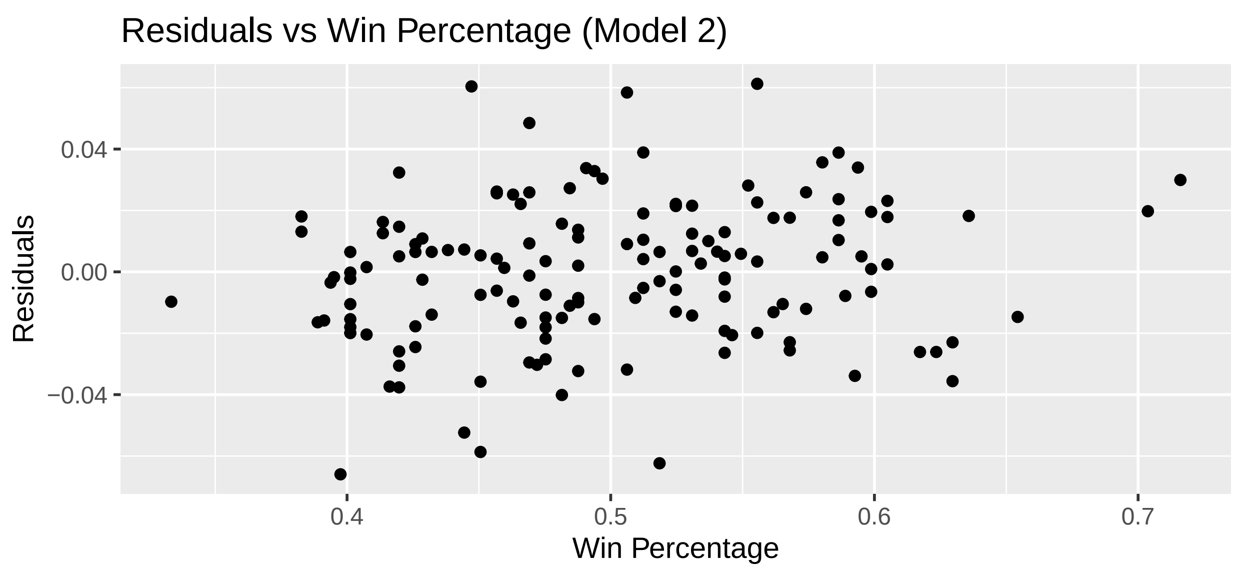
```
## [1] 0.023058
```

Next, let's take a look at the residuals from Model 2.

```r
# let's look at min, max, 1Q, 3Q, median residual
my_teams %>%
  summarise(min = min(.resid.pyt2),
            Q1 = quantile(.resid.pyt2, 0.25),
            median = median(.resid.pyt2),
            Q3 = quantile(.resid.pyt2, 0.75),
            max = max(.resid.pyt2))
```

```
## # A tibble: 1 x 5
##       min      Q1  median     Q3    max
##     <dbl>   <dbl>   <dbl>  <dbl>  <dbl>
## 1 -0.0660 -0.0154 0.00143 0.0158 0.0613
```

```r
my_teams %>%
  ggplot(aes(x = Wpct,
             y = .resid.pyt2)) +
  geom_point() +
  labs(x = "Win Percentage", y = "Residuals",
       title = "Residuals vs Win Percentage (Model 2)")
```



Next, let's try Model 3.

How do we find an estimate of k in Model 3?

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

#0 in formula means we don't include an intercept
pytFit <- lm(logWratio ~ 0 + logRratio, data = my_teams)
pytFit
```

```
##
## Call:
## lm(formula = logWratio ~ 0 + logRratio, data = my_teams)
##
## Coefficients:
## logRratio
##     1.904
```

Why don't we want an intercept in this model? (Warning: this is very unusual!!)

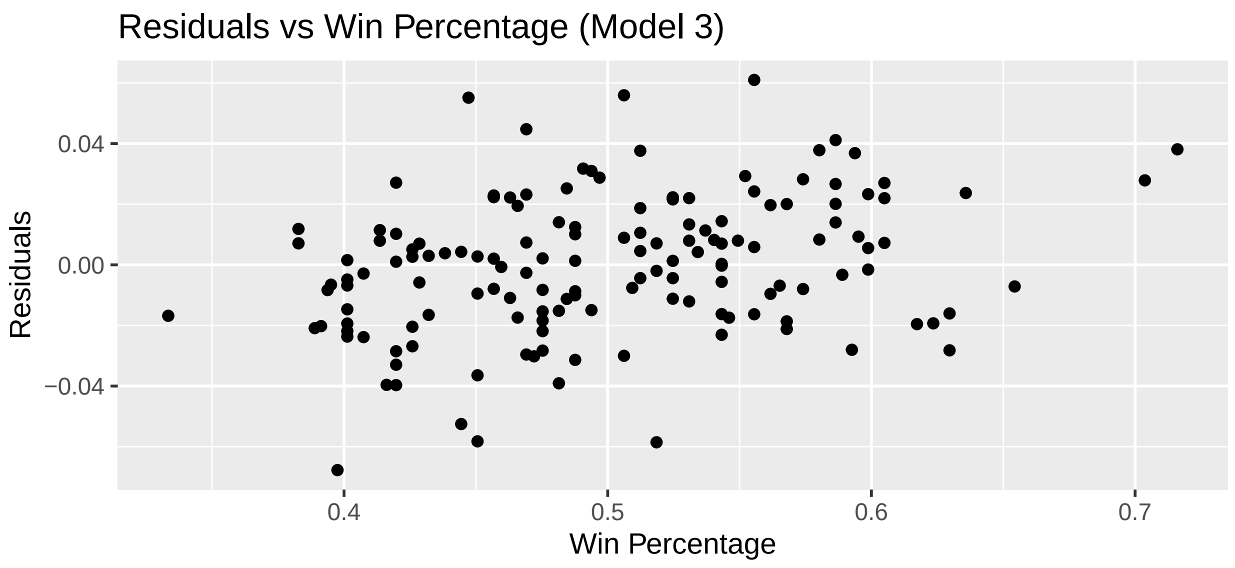
```r
k = pytFit$coefficients[1]

#get predictions
my_teams = my_teams %>%
  mutate(Wpct.pyt_k = pyt_wins(R,RA,k = k))

#get residuals
my_teams = my_teams %>%
  mutate(.resid.pyt_k = Wpct - Wpct.pyt_k)

#calculate RMSE
sqrt(mean(my_teams$.resid.pyt_k^2))
```

```
## [1] 0.02279093
```

```
my_teams %>%
  summarise(min = min(.resid.pyt_k),
            Q1 = quantile(.resid.pyt_k, 0.25),
            median = median(.resid.pyt_k),
            Q3 = quantile(.resid.pyt_k, 0.75),
            max = max(.resid.pyt_k))
```

```
## # A tibble: 1 x 5
##       min      Q1  median     Q3    max
##     <dbl>   <dbl>   <dbl>  <dbl>  <dbl>
## 1 -0.0677 -0.0163 0.00116 0.0140 0.0610
```

```
my_teams %>%
  ggplot(aes(x = Wpct,
             y = .resid.pyt_k)) +
  geom_point() +
  labs(x = "Win Percentage", y = "Residuals",
       title = "Residuals vs Win Percentage (Model 3)")
```



How many runs for a win (pg 106)

Previously, we learned about the "10-to-1" rule of thumb (number of required runs scored for one additional win) using Model 1. Using Model 2, this quantity changes based on runs allowed. How can we find an expression for the number of required runs scored for one additional win?

Let's explore this for various combinations of runs scored and runs allowed. (*Note R and RA below are runs scored per game and runs allowed per game.*)

```r
#function to compute incremental runs per win
IR <- function(R = 5, RA = 5){
  (R^2 + RA^2)^2 /(2*R*RA^2)
}

ir_table <- expand.grid(R = seq(3,6,0.5),
                        RA = seq(3,6,0.5))

ir_table %>%
  mutate(IRW = IR(R,RA)) %>%
  spread(key = RA, value = IRW, sep = "=") %>%
  round(1)
```

```
##     R RA=3 RA=3.5 RA=4 RA=4.5 RA=5 RA=5.5 RA=6
## 1 3.0  6.0    6.1  6.5    7.0  7.7    8.5  9.4
## 2 3.5  7.2    7.0  7.1    7.5  7.9    8.5  9.2
## 3 4.0  8.7    8.1  8.0    8.1  8.4    8.8  9.4
## 4 4.5 10.6    9.6  9.1    9.0  9.1    9.4  9.8
## 5 5.0 12.8   11.3 10.5   10.1 10.0   10.1 10.3
## 6 5.5 15.6   13.4 12.2   11.4 11.1   11.0 11.1
## 7 6.0 18.8   15.8 14.1   13.0 12.4   12.1 12.0
```

How do the results of Model 2 compare to Model 1 in terms of incremental runs per win?