Moneyball

In this project you will apply your data wrangling and exploratory data analysis skills to baseball. In particular, we want to know how well did Moneyball work for the Oakland A's. Was it worthy of a movie?

A bit of background

We'll be looking at data about teams in Major League Baseball. A couple of important points:

- Major League Baseball is a professional baseball league, where teams pay players to play baseball.
- The goal of each team is to win as many games out of a 162-game season as possible.
- Teams win games by scoring more runs than their adversary.
- In principle, better players are costlier, so teams that want good players need to spend more money.
- Teams that spend the most, frequently win the most.

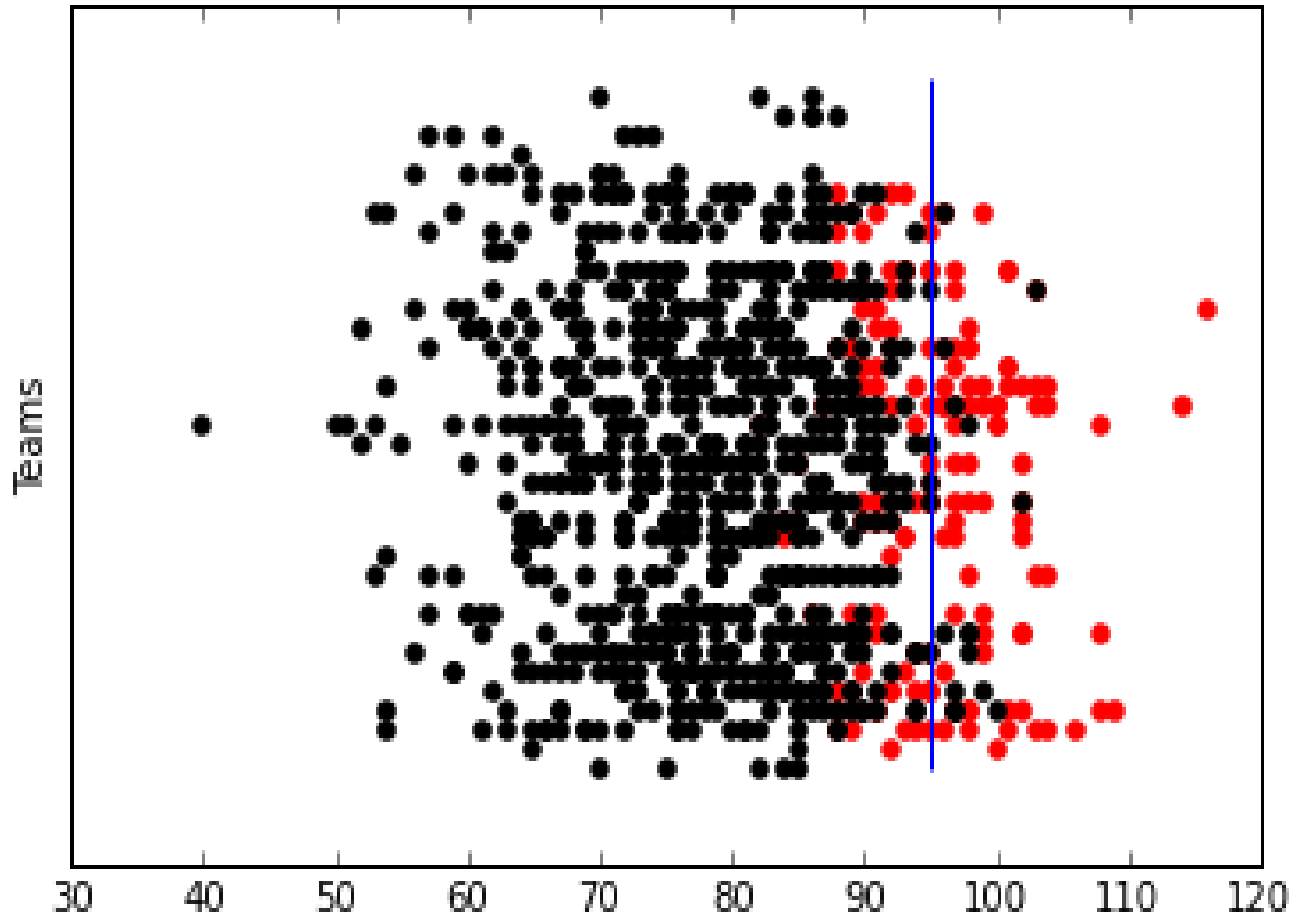
So, the question is, how can a team that can't spend so much win? The basic idea that Oakland (and other teams) used is to *redefine* what makes a player *good*. I.e., figure out what player characteristics translated into *wins*. Once they realized that teams were not really pricing players using these characteristics, they could exploit this to pay for undervalued players, players that were *good* according to their metrics, but were not recognized as such by other teams, and therefore not as expensive.

> "The statistics enable you to find your way past all sorts of sight-based scouting prejudices."

The goal of a baseball team is to make the playoffs.The A's approach was to get to the playoffs by using analytics.We'll first show how we can predict whether or not a team will make the playoffs by knowing how many games they won in the regular season. We'll then use linear regression to predict how many games a team will win using the difference between runs scored and runs allowed, or opponent runs. We'll then use linear regression again to predict the number of runs a team will score using batting statistics, and the number of runs a team will allow using fielding and pitching statistics. We'll start by figuring out how many games a team needs to win to make the playoffs, and then how many more runs a team needs to score than their opponent to win that many games.

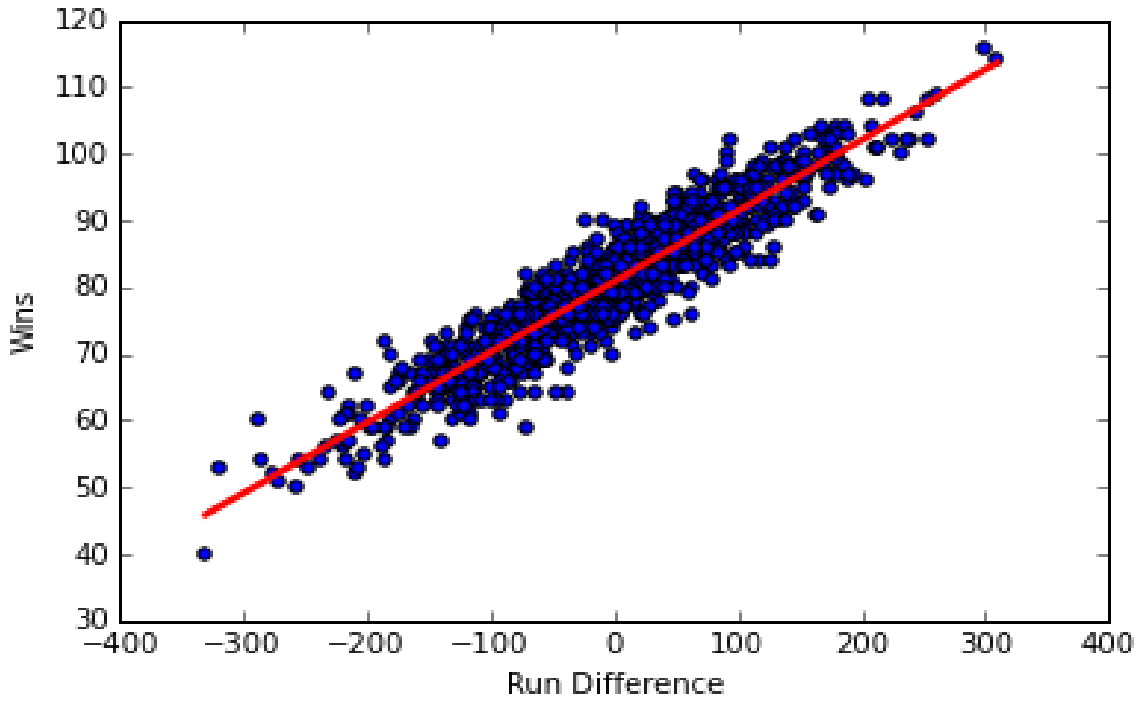
So our first question is: how many games does a team need to win in the regular season to make it to the playoffs?

In Moneyball, Paul DePodesta reduced the regular season to a math problem.He judged that it would take 95 wins for the A's to make it to the playoffs. Let's see if we can verify this using data.



Red dot depicts the number of wins, a team making to the playoffs scored and black dot are for the teams not making to the playoff.
This graph shows us that historically, if a team won 95 or more games, or is on the right side of this line, then they almost all made it to the playoffs. But if a team only won, say, 85 or more games, then they were likely to make it to the playoffs but it wasn't as certain. And if a team won, say, 100 or more games, than they definitely made it to the playoffs. So this plot shows us that historically, if a team won 95 or more games, then they had a strong chance of making it to the playoffs, which confirms Paul DePodesta's claim.
So we know that a team wants to win 95 or more games,
but how does a team win games?

They analyzed run difference between the runs score and run conceded overall.



The A's calculated that they needed to score 135 more runs than they allowed during the regular season to expect to win 95 games. Let's see if we can verify this using linear regression in R.

```
b = read.csv('baseball.csv')
str(b)
```

```
'data.frame':  1232 obs. of  15 variables:
 $ Team        : Factor w/ 39 levels "ANA","ARI","ATL",..: 2 3 4 5 7 8 9 10 1
1 12 ...
 $ League      : Factor w/ 2 levels "AL","NL": 2 2 1 1 2 1 2 1 2 1 ...
 $ Year        : int  2012 2012 2012 2012 2012 2012 2012 2012 2012 2012 ...
 $ RS          : int  734 700 712 734 613 748 669 667 758 726 ...
 $ RA          : int  688 600 705 806 759 676 588 845 890 670 ...
 $ W           : int  81 94 93 69 61 85 97 68 64 88 ...
 $ OBP         : num  0.328 0.32 0.311 0.315 0.302 0.318 0.315 0.324 0.33 0.3
35 ...
 $ SLG         : num  0.418 0.389 0.417 0.415 0.378 0.422 0.411 0.381 0.436 0
.422 ...
 $ BA          : num  0.259 0.247 0.247 0.26 0.24 0.255 0.251 0.251 0.274 0.2
68 ...
 $ Playoffs    : int  0 1 1 0 0 0 1 0 0 1 ...
 $ RankSeason  : int  NA 4 5 NA NA NA 2 NA NA 6 ...
 $ RankPlayoffs: int  NA 5 4 NA NA NA 4 NA NA 2 ...
 $ G           : int  162 162 162 162 162 162 162 162 162 162 ...
 $ OOBP        : num  0.317 0.306 0.315 0.331 0.335 0.319 0.305 0.336 0.357 0
.314 ...
```

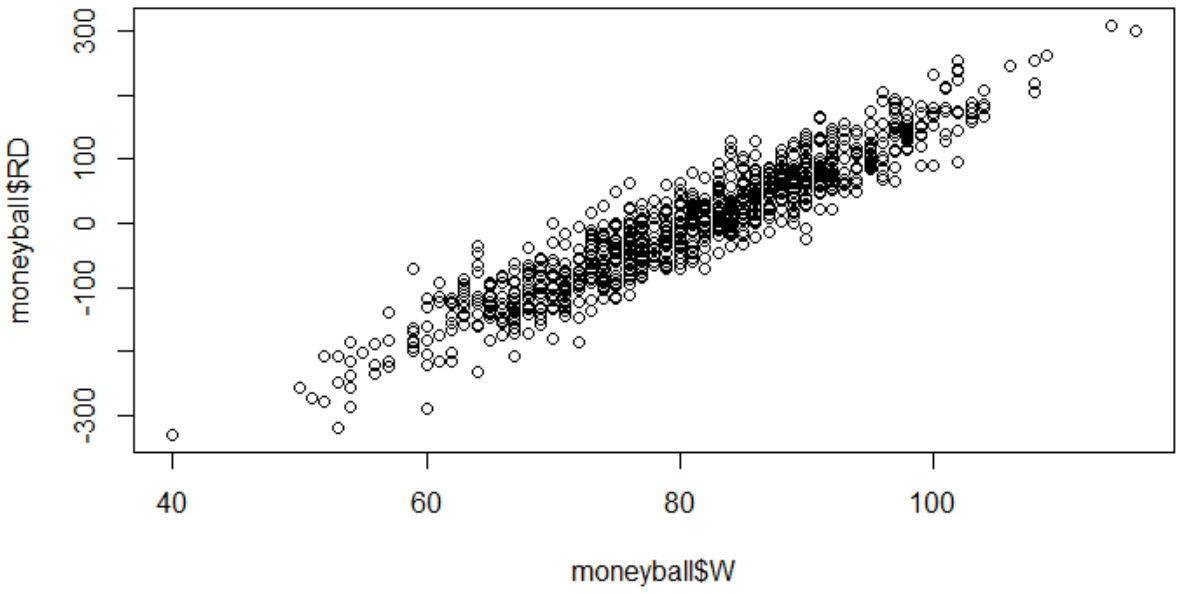
```
 $ OSLG        : num  0.415 0.378 0.403 0.428 0.424 0.405 0.39 0.43 0.47 0.40
2 ...
```

```
#Taking data till 2002(year of Moneyball)
moneyball = subset(b , Year<2002)

#Now observations are reduced to 902 from 1232.


#Lets create a new column in our dataset for run difference.
moneyball$RD = moneyball$RS - moneyball$RA

#chechking structure of data again.
str(moneyball)
```

We have new column RD in our dataset.

#relation between wins and run difference

```
plot(moneyball$W,moneyball$RD)
```



#lets run a regression model on win using only RD as a variable

```
win = lm(W~ RD, data = moneyball)
summary(win)
```

```
Coefficients:
             Estimate Std. Error t value Pr(>|t|)
```

```
(Intercept) 80.881375   0.131157  616.67   <2e-16 ***
RD           0.105766   0.001297   81.55   <2e-16 ***

Multiple R-squared:  0.8808
```

That's a strong model.

Coefficient table tells that

W= 80.88 + 0.105*RD

As per Moneyball analysis, to make it to playoffs we need more than 95 wins.
W>=95

80.88 + 0.105*RD >= 95

By rearranging we can say that as team needs a run difference of more than 133.5 to make it to the playoffs.

RD >=133.5 which almost equal to 135(Paul Depodesta claim)

Let's start by creating a linear regression model to predict runs scored. The Oakland A's were interested in answering the question: how does a team score more runs?
They discovered that two particular baseball statistics were very predictive of runs scored:
on-base percentage, or OBP, and slugging percentage, or SLG. On-base percentage is the percentage of time a player gets on base, including walks. Slugging percentage measures how far a player gets around the bases on his turn, and measures the power of a hitter.
Most baseball teams and experts have historically focused on batting average, which measures how often a hitter gets on base by hitting the ball. It's similar to on-base percentage but excludes walks.
The Oakland A's claimed that on-base percentage was the most important statistic for predicting runs. That it doesn't matter if a player gets on base by hitting the ball or walking, just that they got on base. They also claimed that slugging percentage was important, and batting average was overvalued.
Let's see if we can use linear regression in R, to verify which baseball statistics are important for predicting runs scored.

Our data set has many variables, including runs scored, RS,on-base percentage, OBP, slugging percentage, SLG, and batting average, BA. We want to see if we can use linear regression to predict runs scored, RS, using these three hitting statistics-- on-base percentage, slugging percentage and batting average.

So let's build a linear regression equation. We'll call it Runs, and we'll use the lm function to predict runs scored, RS, using OBP, SLG, and BA.
We'll use the data set moneyball.

#lets run regression using three variables mentioned

```
Runs = lm(RS ~ OBP + SLG + BA, data = moneyball)
summary(Runs)
```

```
Estimate Std. Error t value Pr(>|t|)
(Intercept)  -788.46      19.70 -40.029  < 2e-16 ***
OBP          2917.42     110.47  26.410  < 2e-16 ***
SLG          1637.93      45.99  35.612  < 2e-16 ***
BA           -368.97     130.58  -2.826  0.00482 **
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
Multiple R-squared:  0.9302
```

If we take a look at the summary of our regression equation, we can see that all of our independent variables are significant, and our R-squared is 0.93. But if we look at our coefficients, we can see that the coefficient for batting average is negative. This implies that, all else being equal, a team with a lower batting average will score more runs, which is a little counterintuitive. What's going on here is a case of multicollinearity.
These three hitting statistics are highly correlated, so it's hard to interpret the coefficients of our model. Let's try removing batting average, the variable with the least significance, to see what happens to our model.

running regression without batting average.

```
Runs = lm(RS ~ OBP + SLG , data = moneyball)
summary(Runs)
```

```
Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept)  -804.63      18.92  -42.53   <2e-16 ***
OBP          2737.77      90.68   30.19   <2e-16 ***
SLG          1584.91      42.16   37.60   <2e-16 ***
Multiple R-squared:  0.9296
```

We can see that our independent variables are still very significant, the coefficients are both positive as we expect, and our R-squared is still about 0.93. So this model is simpler, with only two independent variables, and has about the same R-squared.
Overall a better model.
You could experiment and see that if we'd removed on-base percentage or slugging percentage instead of batting average, our R-squared would have decreased.
If we look at the coefficients of our model, we can see that on-base percentage has a larger coefficient than slugging percentage. Since these variables are on about the same scale, this tells us that on-base percentage is probably worth more than slugging percentage.

So by using linear regression, we're able to verify the claims made in Moneyball:
that batting average is overvalued, on-base percentage is the most important, and slugging percentage is important for predicting runs scored.

We can create a very similar model to predict runs allowed, or opponent runs. This model uses pitching statistics: opponents on-base percentage, or OOBP, and opponents slugging percentage, or OSLG. These statistics are computed in the same way as on-base percentage and slugging percentage, but they use the actions of the opposing batters against our team's pitcher and fielders.
Using our data set in R, we can build a linear regression model to predict runs allowed, using opponents on-base percentage and opponents slugging percentage.

```
#using regression to understand runs allowed.
Runsallowed = lm(RA ~ OOBP + OSLG ,data = moneyball)
summary(Runsallowed)
```

```
Estimate Std. Error t value Pr(>|t|)
(Intercept)  -837.38      60.26 -13.897  < 2e-16 ***
OOBP         2913.60     291.97   9.979 4.46e-16 ***
OSLG         1514.29     175.43   8.632 2.55e-13 ***
Multiple R-squared:  0.9073
```

This is, again, a very strong model with an R-squared value of 0.91, and both variables are significant. The key message here is that simple models, using only a couple independent variables, can be constructed to answer some of the most important questions in baseball.

Using our regression models, we would like to predict before the season starts how many games the 2002 Oakland A's will win. To do this, we first have to predict how many runs the team will score and how many runs they will allow. These models use team statistics. However, when we are predicting for the 2002 Oakland A's before the season has occurred, the team is probably different than it was the year before. So we don't know the team statistics. But we can estimate these statistics using past player performance.

Using the 2001 regular season statistics for these players, we can estimate that team on-base percentage will be about 0.339 and team slugging percentage will be about 0.430.
We built the following linear regression equation in R to predict runs scored. If we plug in 0.339 for on-base percentage and 0.430 for slugging percentage, we predict that the 2002 Oakland A's will score about **805** runs.

RS = -804.63 + 2737.77*OBP +1584*SLG

Similarly, we can make a prediction for runs allowed. Using the 2001 regular season statistics for these players, we can estimate that team opponent on-base percentage will be about 0.307 and team opponent slugging percentage will be about 0.373. Our regression equation to predict runs allowed was as follows.
RA = -837.38 +2913.6*OOBP + 1514.29*OSLG

By plugging in 0.307 for opponents on-base percentage and 0.373 for opponents slugging percentage, we predict that the 2002 Oakland A's will allow **622** runs.

We can now make a prediction for how many games they will win.

Our regression equation to predict wins is as follows.

$$W = 80.88 + 0.105*(RS-RA)$$

We predicted 805 runs scored and 622 runs allowed. We can plug in the difference between runs scored and runs allowed to predict that the A's will win 100 games in 2002. Paul DePodesta used a similar approach to make predictions. It turns out that our predictions and Paul's predictions closely match actual performance.

	Our Prediction	Paul's Prediction	Actual
Runs Scored	805	800 – 820	800
Runs Allowed	622	650 – 670	653
Wins	100	93 – 97	103

These predictions show us that by using publicly available data and simple analytics, we can predict very close to what actually happened before the season even started.

Initially, we stated that the goal of a baseball team is to make the playoffs and we built predictive models to achieve this goal. But why isn't the goal of a baseball team to win the playoffs or win the World Series? Billy Beane and Paul Depodesta see their job as making sure the team makes it to the playoffs,and after that, all bets are off. The A's made it to the playoffs four years in a row-- 2000, 2001, 2002, and 2003-- but they didn't win the World Series.
Why not?

The playoffs suffer from the sample size problem. There are not enough games to make any statistical claims.We can compute the correlation between whether the team wins the World Series-- a binary variable--and the number of regular season wins, since we would expect teams with more wins to be more likely to win the World Series.

```
#Finding correlation between a team wins and winning the world series
#making a subset of all the values in b
#Selecting team with playoff ranks as 1 as assiging WSwin(worldseries win = 1) to them
recent <- subset(b, Year >= 1994 & Year <= 2011)
recent[recent$RankPlayoffs == 1 & is.na(recent$RankPlayoffs) == FALSE, ]
recent$WSwin <- 0
recent[recent$RankPlayoffs == 1 & is.na(recent$RankPlayoffs) == FALSE, "WSwin"] <- 1
# Finding a correlation between world series wins and the wins in regular league.
```

```
cor(recent$WSwin, recent$W)
```
```
0.225398
```

This correlation is 0.02, which is very low. So it turns out that winning regular season games gets you to the playoffs, but in the playoffs, there are too few games for luck to even out.

```
# Now we'll try using logistic regression to predict whether or not a team will win the World Series.
# We will need only those team making to the playoffs so we make a subset.

baseball = subset(b , b$Playoffs==1)

#Lets chaeck teams making to the playoffs overall
playoff = table(baseball$Year)
playoff
names(playoff)

#Adding one column baseball dataset to show number of teams qualifying for playoff for that year.
baseball$NumCompetitors = playoff[as.character(baseball$Year)]

#Adding a column in baseball dataset for showing team winning the Worldseries.
baseball$worldseries = as.numeric(baseball$RankPlayoffs==1)
```

Now let's use logistic regression to check whether we can predict WorldSeries for a win
When we're not sure which of our variables are useful in predicting a outcome, it's often helpful to build bivariate models, which are models that predict the outcome using a single independent variable.

```
#LOGISTIC REGRESSION
#But we are unsure which variables we should use to run the prediction. For this we will use the bivariate model,
#testing with every independent variable to check which one should be included.

model1 = glm(worldseries~Year, data = baseball,family = binomial)
summary(model1)
model2 = glm(worldseries~RS, data = baseball,family = binomial)
summary(model2)
model3 = glm(worldseries~RA, data = baseball,family = binomial)
summary(model3)
model4 = glm(worldseries~W, data = baseball,family = binomial)
summary(model4)
model5 = glm(worldseries~OBP, data = baseball,family = binomial)
summary(model5)
model6 = glm(worldseries~SLG, data = baseball,family = binomial)
summary(model6)
```

```
model7 = glm(worldseries~BA, data = baseball,family = binomial)
summary(model7)
model8 = glm(worldseries~RankSeason, data = baseball,family = binomial)
summary(model8)
model9 = glm(worldseries~OOBP, data = baseball,family = binomial)
summary(model9)
model10 = glm(worldseries~SLG, data = baseball,family = binomial)
summary(model10)
model11 = glm(worldseries~NumCompetitors, data = baseball,family = binomial)
summary(model11)
model12 = glm(worldseries~League, data = baseball,family = binomial)
summary(model12)


Call:
glm(formula = worldseries ~ Year, family = binomial, data = baseball)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-1.0297  -0.6797  -0.5435  -0.4648   2.1504

Coefficients:
             Estimate Std. Error z value Pr(>|z|)
(Intercept) 72.23602   22.64409    3.19  0.00142 **
Year        -0.03700    0.01138   -3.25  0.00115 **
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 239.12  on 243  degrees of freedom
Residual deviance: 228.35  on 242  degrees of freedom
AIC: 232.35

Number of Fisher Scoring iterations: 4

> model2 = glm(worldseries~RS, data = baseball,family = binomial)
> summary(model2)

Call:
glm(formula = worldseries ~ RS, family = binomial, data = baseball)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-0.8254  -0.6819  -0.6363  -0.5561   2.0308

Coefficients:
             Estimate Std. Error z value Pr(>|z|)
(Intercept)  0.661226   1.636494   0.404    0.686
RS          -0.002681   0.002098  -1.278    0.201

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 239.12  on 243  degrees of freedom
Residual deviance: 237.45  on 242  degrees of freedom
AIC: 241.45

Number of Fisher Scoring iterations: 4
```

```
> model3 = glm(worldseries~RA, data = baseball,family = binomial)
> summary(model3)

Call:
glm(formula = worldseries ~ RA, family = binomial, data = baseball)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-0.9749  -0.6883  -0.6118  -0.4746   2.1577

Coefficients:
             Estimate Std. Error z value Pr(>|z|)
(Intercept)  1.888174   1.483831   1.272   0.2032
RA          -0.005053   0.002273  -2.223   0.0262 *
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 239.12  on 243  degrees of freedom
Residual deviance: 233.88  on 242  degrees of freedom
AIC: 237.88

Number of Fisher Scoring iterations: 4

> model4 = glm(worldseries~W, data = baseball,family = binomial)
> summary(model4)

Call:
glm(formula = worldseries ~ W, family = binomial, data = baseball)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-1.0623  -0.6777  -0.6117  -0.5367   2.1254

Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept) -6.85568    2.87620  -2.384   0.0171 *
W            0.05671    0.02988   1.898   0.0577 .
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 239.12  on 243  degrees of freedom
Residual deviance: 235.51  on 242  degrees of freedom
AIC: 239.51

Number of Fisher Scoring iterations: 4

> model5 = glm(worldseries~OBP, data = baseball,family = binomial)
> summary(model5)

Call:
glm(formula = worldseries ~ OBP, family = binomial, data = baseball)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-0.8071  -0.6749  -0.6365  -0.5797   1.9753

Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept)    2.741      3.989   0.687    0.492
```

```
OBP          -12.402     11.865  -1.045    0.296

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 239.12  on 243  degrees of freedom
Residual deviance: 238.02  on 242  degrees of freedom
AIC: 242.02

Number of Fisher Scoring iterations: 4

> model6 = glm(worldseries~SLG, data = baseball,family = binomial)
> summary(model6)

Call:
glm(formula = worldseries ~ SLG, family = binomial, data = baseball)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-0.9498  -0.6953  -0.6088  -0.5197   2.1136

Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept)    3.200      2.358   1.357   0.1748
SLG          -11.130      5.689  -1.956   0.0504 .
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 239.12  on 243  degrees of freedom
Residual deviance: 235.23  on 242  degrees of freedom
AIC: 239.23

Number of Fisher Scoring iterations: 4

> model7 = glm(worldseries~BA, data = baseball,family = binomial)
> summary(model7)

Call:
glm(formula = worldseries ~ BA, family = binomial, data = baseball)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-0.6797  -0.6592  -0.6513  -0.6389   1.8431

Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept)  -0.6392     3.8988  -0.164    0.870
BA           -2.9765    14.6123  -0.204    0.839

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 239.12  on 243  degrees of freedom
Residual deviance: 239.08  on 242  degrees of freedom
AIC: 243.08

Number of Fisher Scoring iterations: 4

> model8 = glm(worldseries~RankSeason, data = baseball,family = binomial)
> summary(model8)

Call:
glm(formula = worldseries ~ RankSeason, family = binomial, data = baseball)
```

```
Deviance Residuals:
    Min       1Q   Median       3Q      Max
-0.7805  -0.7131  -0.5918  -0.4882   2.1781

Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept)  -0.8256     0.3268  -2.527   0.0115 *
RankSeason   -0.2069     0.1027  -2.016   0.0438 *
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 239.12  on 243  degrees of freedom
Residual deviance: 234.75  on 242  degrees of freedom
AIC: 238.75

Number of Fisher Scoring iterations: 4

> model9 = glm(worldseries~OOBP, data = baseball,family = binomial)
> summary(model9)

Call:
glm(formula = worldseries ~ OOBP, family = binomial, data = baseball)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-0.5318  -0.5176  -0.5106  -0.5023   2.0697

Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept)  -0.9306     8.3728  -0.111    0.912
OOBP         -3.2233    26.0587  -0.124    0.902

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 84.926  on 113  degrees of freedom
Residual deviance: 84.910  on 112  degrees of freedom
  (130 observations deleted due to missingness)
AIC: 88.91

Number of Fisher Scoring iterations: 4

> model10 = glm(worldseries~SLG, data = baseball,family = binomial)
> summary(model10)

Call:
glm(formula = worldseries ~ SLG, family = binomial, data = baseball)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-0.9498  -0.6953  -0.6088  -0.5197   2.1136

Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept)    3.200      2.358   1.357   0.1748
SLG          -11.130      5.689  -1.956   0.0504 .
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 239.12  on 243  degrees of freedom
Residual deviance: 235.23  on 242  degrees of freedom
```

```
AIC: 239.23

Number of Fisher Scoring iterations: 4

> model11 = glm(worldseries~NumCompetitors, data = baseball,family = binomial
)
> summary(model11)

Call:
glm(formula = worldseries ~ NumCompetitors, family = binomial,
    data = baseball)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-0.9871  -0.8017  -0.5089  -0.5089   2.2643

Coefficients:
              Estimate Std. Error z value Pr(>|z|)
(Intercept)    0.03868    0.43750   0.088 0.929559
NumCompetitors -0.25220    0.07422  -3.398 0.000678 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 239.12  on 243  degrees of freedom
Residual deviance: 226.96  on 242  degrees of freedom
AIC: 230.96

Number of Fisher Scoring iterations: 4

> model12 = glm(worldseries~League, data = baseball,family = binomial)
> summary(model12)

Call:
glm(formula = worldseries ~ League, family = binomial, data = baseball)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-0.6772  -0.6772  -0.6306  -0.6306   1.8509

Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept)  -1.3558     0.2243  -6.045  1.5e-09 ***
LeagueNL     -0.1583     0.3252  -0.487    0.626
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 239.12  on 243  degrees of freedom
Residual deviance: 238.88  on 242  degrees of freedom
AIC: 242.88

Number of Fisher Scoring iterations: 4
```

After running the bivariate model for every independent variable we can see that only 4 variables turn out to be significant.

Year , RA, Rankseason, NumCompetitors(the variable that we created)

Now let's try runnig a model with those significant variables found with bivariate method.
Lets find how many turn out to be significant in multivariate model.

```
model13 = glm(worldseries~ Year + RA + NumCompetitors + RankSeason, data = baseball,family = binomial)
summary(model13)
```

```
Coefficients:
                  Estimate Std. Error z value Pr(>|z|)
(Intercept)    12.5874376 53.6474210   0.235    0.814
Year           -0.0061425  0.0274665  -0.224    0.823
RA             -0.0008238  0.0027391  -0.301    0.764
NumCompetitors -0.1794264  0.1815933  -0.988    0.323
RankSeason     -0.0685046  0.1203459  -0.569    0.569
```

We can see that none of them are significant in multivariate model.
Often, variables that were significant in bivariate models are no longer significant in multivariate analysis due to correlation between the variables.

#Lets check for correlation if there is any between these variables.
cor(baseball[c("Year", "RA", "RankSeason", "NumCompetitors")])

While every pair was at least moderately correlated, the only strongly correlated pair was Year/NumCompetitors, with correlation coefficient 0.914.

Now lets build a two variable to check whether they come up with any significance.
The two-variable models can be built with the following commands:

```
Model14 = glm(WorldSeries ~ Year + RA, data=baseball, family=binomial)
Model15 = glm(WorldSeries ~ Year + RankSeason, data=baseball, family=binomial)
Model16 = glm(WorldSeries ~ Year + NumCompetitors, data=baseball, family=binomial)
Model17 = glm(WorldSeries ~ RA + RankSeason, data=baseball, family=binomial)
Model18 = glm(WorldSeries ~ RA + NumCompetitors, data=baseball, family=binomial)
Model19 = glm(WorldSeries ~ RankSeason + NumCompetitors, data=baseball, family=binomial)
```

```
summary(Model14)
summary(Model15)
summary(Model16)
summary(Model17)
summary(Model18)
summary(Model19)
```

```
Call:
glm(formula = worldseries ~ Year + RA, family = binomial, data = baseball)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-1.0402  -0.6878  -0.5298  -0.4785   2.1370

Coefficients:
             Estimate Std. Error z value Pr(>|z|)
(Intercept) 63.610741  25.654830   2.479   0.0132 *
Year        -0.032084   0.013323  -2.408   0.0160 *
RA          -0.001766   0.002585  -0.683   0.4945
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 239.12  on 243  degrees of freedom
Residual deviance: 227.88  on 241  degrees of freedom
AIC: 233.88

Number of Fisher Scoring iterations: 4

> summary(Model15)

Call:
glm(formula = worldseries ~ Year + RankSeason, family = binomial,
    data = baseball)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-1.0560  -0.6957  -0.5379  -0.4528   2.2673

Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept) 63.64855   24.37063   2.612  0.00901 **
Year        -0.03254    0.01231  -2.643  0.00822 **
RankSeason  -0.10064    0.11352  -0.887  0.37534
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 239.12  on 243  degrees of freedom
Residual deviance: 227.55  on 241  degrees of freedom
AIC: 233.55

Number of Fisher Scoring iterations: 4

> summary(Model16)

Call:
glm(formula = worldseries ~ Year + NumCompetitors, family = binomial,
    data = baseball)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-1.0050  -0.7823  -0.5115  -0.4970   2.2552

Coefficients:
               Estimate Std. Error z value Pr(>|z|)
(Intercept)    13.350467  53.481896   0.250    0.803
Year           -0.006802   0.027328  -0.249    0.803
```

```
NumCompetitors -0.212610   0.175520  -1.211    0.226

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 239.12  on 243  degrees of freedom
Residual deviance: 226.90  on 241  degrees of freedom
AIC: 232.9

Number of Fisher Scoring iterations: 4

> summary(Model17)

Call:
glm(formula = worldseries ~ RA + RankSeason, family = binomial,
    data = baseball)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-0.9374  -0.6933  -0.5936  -0.4564   2.1979

Coefficients:
             Estimate Std. Error z value Pr(>|z|)
(Intercept)  1.487461   1.506143   0.988    0.323
RA          -0.003815   0.002441  -1.563    0.118
RankSeason  -0.140824   0.110908  -1.270    0.204

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 239.12  on 243  degrees of freedom
Residual deviance: 232.22  on 241  degrees of freedom
AIC: 238.22

Number of Fisher Scoring iterations: 4

> summary(Model18)

Call:
glm(formula = worldseries ~ RA + NumCompetitors, family = binomial,
    data = baseball)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-1.0433  -0.7826  -0.5133  -0.4701   2.2208

Coefficients:
                Estimate Std. Error z value Pr(>|z|)
(Intercept)     0.716895   1.528736   0.469  0.63911
RA             -0.001233   0.002661  -0.463  0.64313
NumCompetitors -0.229385   0.088399  -2.595  0.00946 **
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 239.12  on 243  degrees of freedom
Residual deviance: 226.74  on 241  degrees of freedom
AIC: 232.74

Number of Fisher Scoring iterations: 4

> summary(Model19)

Call:
glm(formula = worldseries ~ RankSeason + NumCompetitors, family = binomial,
```

```
    data = baseball)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-1.0090  -0.7592  -0.5204  -0.4501   2.2562

Coefficients:
             Estimate Std. Error z value Pr(>|z|)
(Intercept)   0.12277    0.45737   0.268  0.78837
RankSeason   -0.07697    0.11711  -0.657  0.51102
NumCompetitors -0.22784    0.08201  -2.778  0.00546 **
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 239.12  on 243  degrees of freedom
Residual deviance: 226.52  on 241  degrees of freedom
AIC: 232.52

Number of Fisher Scoring iterations: 4
```

None of the models with two independent variables had both variables significant, so none seem promising as compared to a simple bivariate model.

This seems to confirm the claim made by Billy Beane in Moneyball that all that matters in the Playoffs is luck, since NumCompetitors has nothing to do with the quality of the teams!