Net_Rating

Chenjie Li

November 23, 2018

```
# load data
data <- read.csv('/home/chenjie/Desktop/Math564Project/Net_Rating/net_rating.csv')
data$color = "green"
data$color[data$win_ratio>=0.5]="blue"
data$color[data$win_ratio>=0.7317073]="red" #won more than 60 games
```

2014

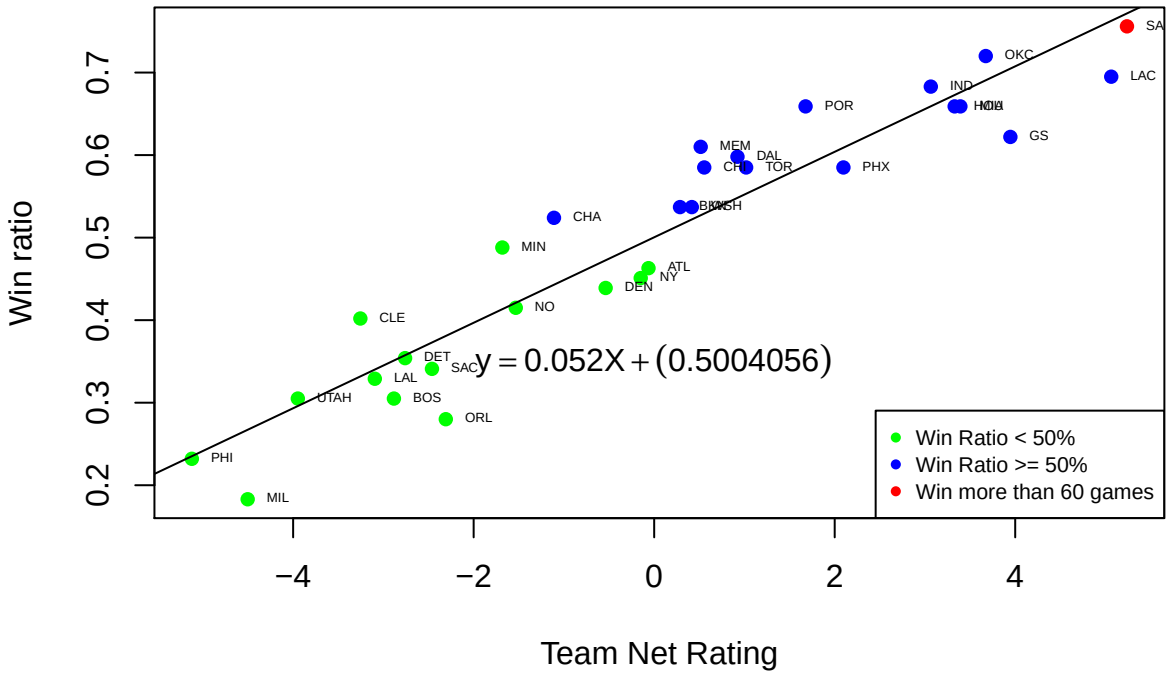
```
s14 <- data[data$season == 2014,]
mod14 <- lm(win_ratio ~ net_rating, data = s14)
summary(mod14)
```

```
##
## Call:
## lm(formula = win_ratio ~ net_rating, data = s14)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.100853 -0.033156 -0.004747  0.031210  0.082903
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 0.500406   0.009466   52.86  < 2e-16 ***
## net_rating  0.051784   0.003344   15.49 2.93e-15 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.05185 on 28 degrees of freedom
## Multiple R-squared:  0.8954, Adjusted R-squared:  0.8917
## F-statistic: 239.8 on 1 and 28 DF,  p-value: 2.934e-15
```

```
plot(s14$net_rating,s14$win_ratio,xlab = 'Team Net Rating', ylab = 'Win ratio', main = '2014 Win_Ratio a
```

```
## integer(0)
```

```
legend("bottomright",legend=c("Win Ratio < 50%", "Win Ratio >= 50%","Win more than 60 games"),
       col=c("green", "blue","red"), pch = c(16,16,16), cex = 0.7)
```



##

2015

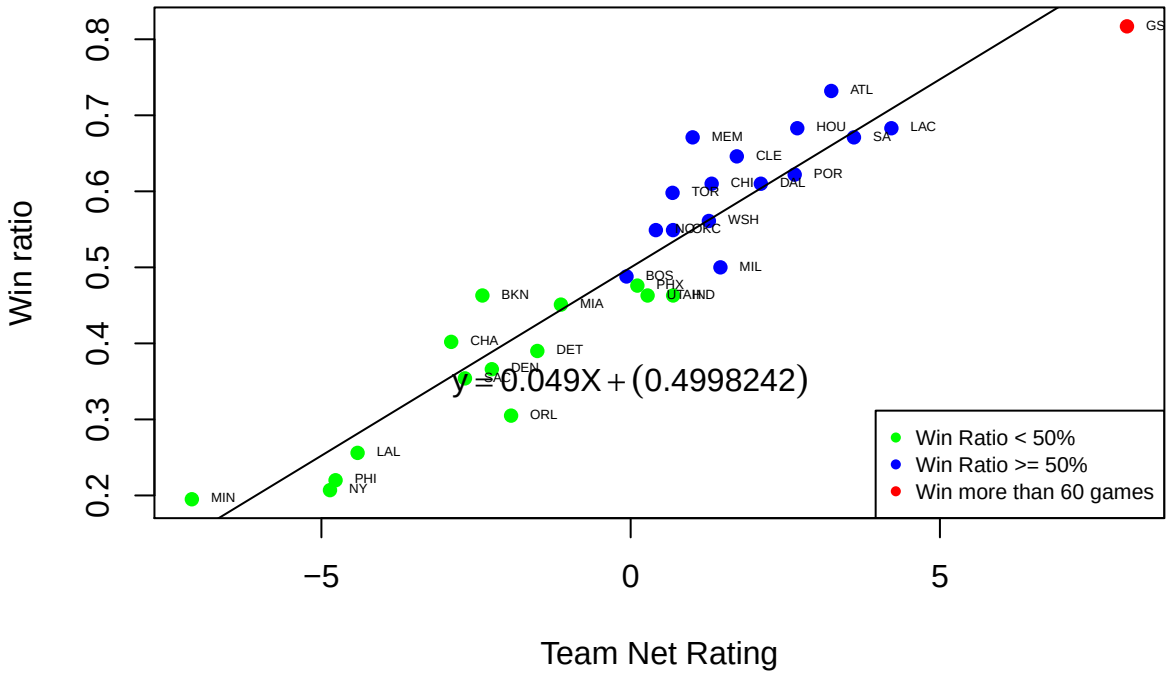
```
s15 <- data[data$season == 2015,]
mod15 <- lm(win_ratio ~ net_rating, data = s15)
summary(mod15)
```

```
##
## Call:
## lm(formula = win_ratio ~ net_rating, data = s15)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.099146 -0.033654 -0.008004  0.045677  0.121572
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 0.499824   0.009899   50.49  < 2e-16 ***
## net_rating  0.049495   0.003210   15.42 3.26e-15 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.05422 on 28 degrees of freedom
## Multiple R-squared:  0.8947, Adjusted R-squared:  0.8909
## F-statistic: 237.8 on 1 and 28 DF,  p-value: 3.262e-15
```

```
plot(s15$net_rating,s15$win_ratio,xlab = 'Team Net Rating', ylab = 'Win ratio', main = '2015 Win_Ratio a
```

```
## integer(0)
```

```
legend("bottomright",legend=c("Win Ratio < 50%", "Win Ratio >= 50%","Win more than 60 games"),
       col=c("green", "blue","red"), pch = c(16,16,16), cex = 0.7)
```



2016

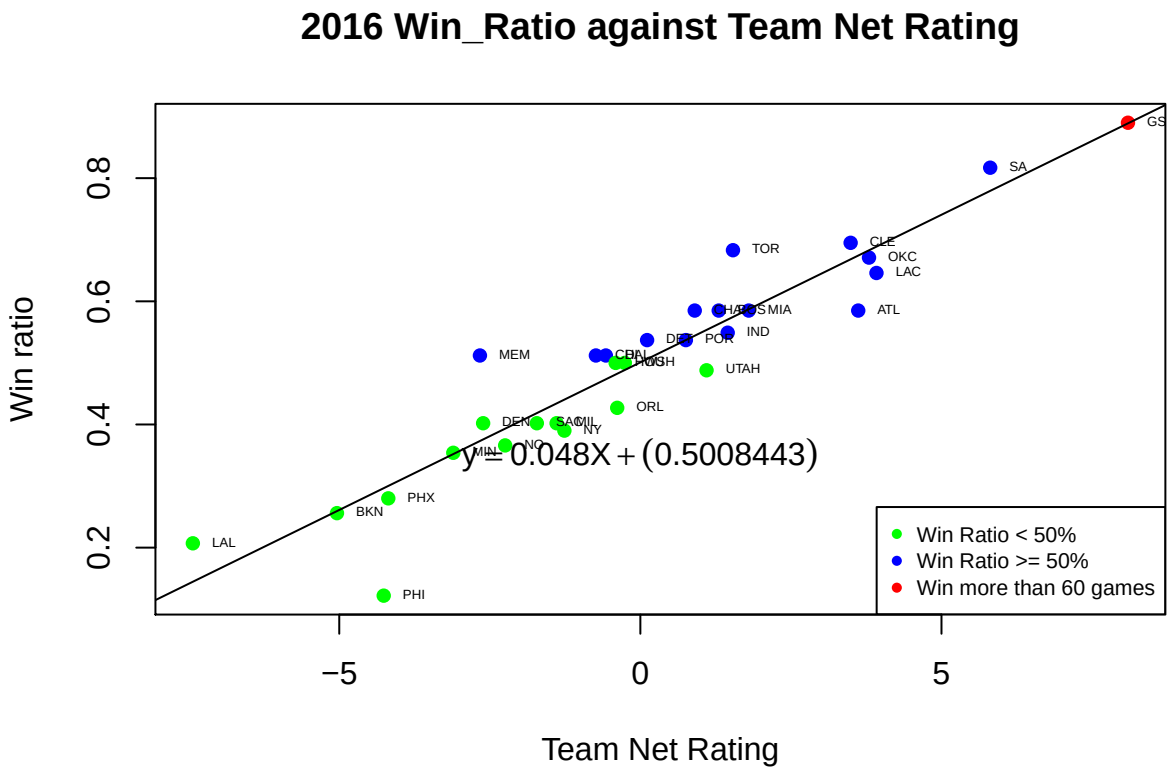
```
s16 <- data[data$season == 2016,]
mod16 <- lm(win_ratio ~ net_rating, data = s16)
summary(mod16)
```

```
##
## Call:
## lm(formula = win_ratio ~ net_rating, data = s16)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.174523 -0.025737  0.000448  0.029789  0.138892
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 0.500844   0.010815   46.31  < 2e-16 ***
## net_rating  0.047943   0.003317   14.46 1.64e-14 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.05924 on 28 degrees of freedom
## Multiple R-squared:  0.8818, Adjusted R-squared:  0.8776
## F-statistic:   209 on 1 and 28 DF,  p-value: 1.639e-14
```

```
plot(s16$net_rating,s16$win_ratio,xlab = 'Team Net Rating', ylab = 'Win ratio', main = '2016 Win_Ratio a
```

```
## integer(0)
```

```
legend("bottomright",legend=c("Win Ratio < 50%", "Win Ratio >= 50%","Win more than 60 games"),
       col=c("green", "blue","red"), pch = c(16,16,16), cex = 0.7)
```



2017

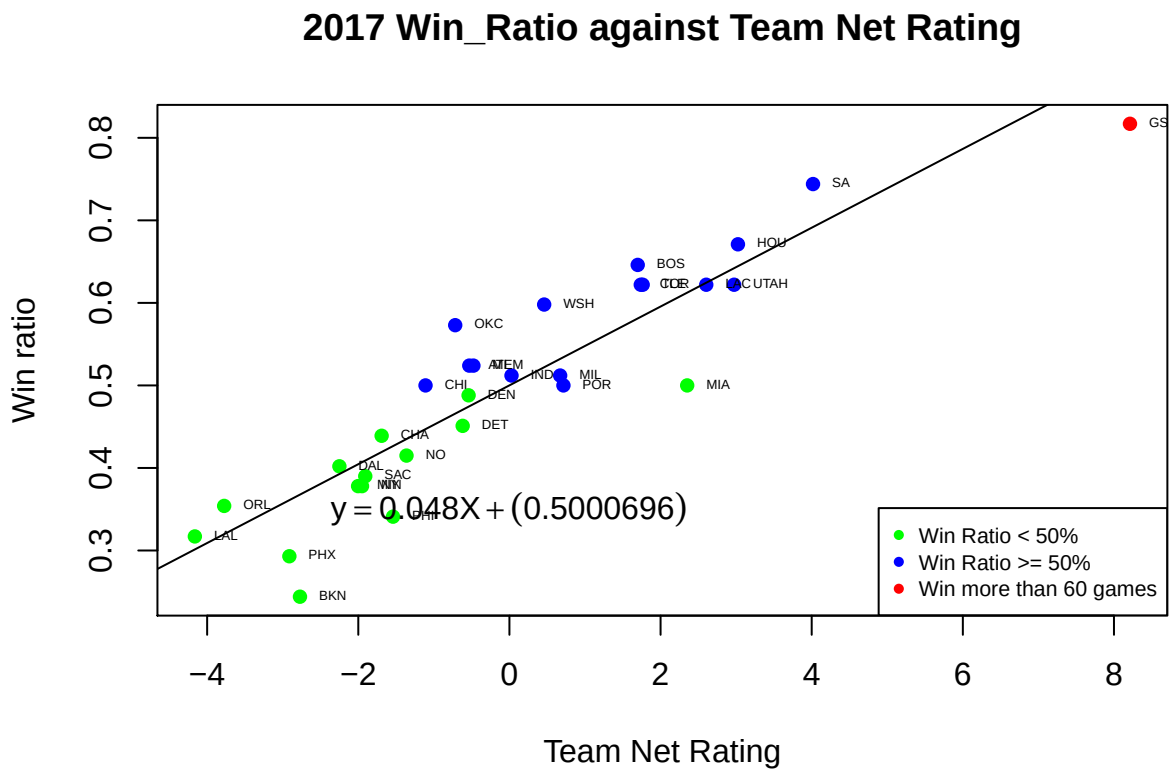
```
s17 <- data[data$season == 2017,]
mod17 <- lm(win_ratio ~ net_rating, data = s17)
summary(mod17)
```

```
##
## Call:
## lm(formula = win_ratio ~ net_rating, data = s17)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.123715 -0.024860  0.009999  0.038621  0.107193
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 0.500070   0.010213   48.96  < 2e-16 ***
## net_rating  0.047755   0.003972   12.02 1.43e-12 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.05594 on 28 degrees of freedom
## Multiple R-squared:  0.8377, Adjusted R-squared:  0.8319
## F-statistic: 144.5 on 1 and 28 DF,  p-value: 1.426e-12
```

```
plot(s17$net_rating,s17$win_ratio,xlab = 'Team Net Rating', ylab = 'Win ratio', main = '2017 Win_Ratio a
```

```
## integer(0)
```

```
legend("bottomright",legend=c("Win Ratio < 50%", "Win Ratio >= 50%","Win more than 60 games"),
       col=c("green", "blue","red"), pch = c(16,16,16), cex = 0.7)
```



4 years as a whole

```
four_years_total <- data
mod_total <- lm(win_ratio ~ net_rating, data = four_years_total)
summary(mod_total)
```

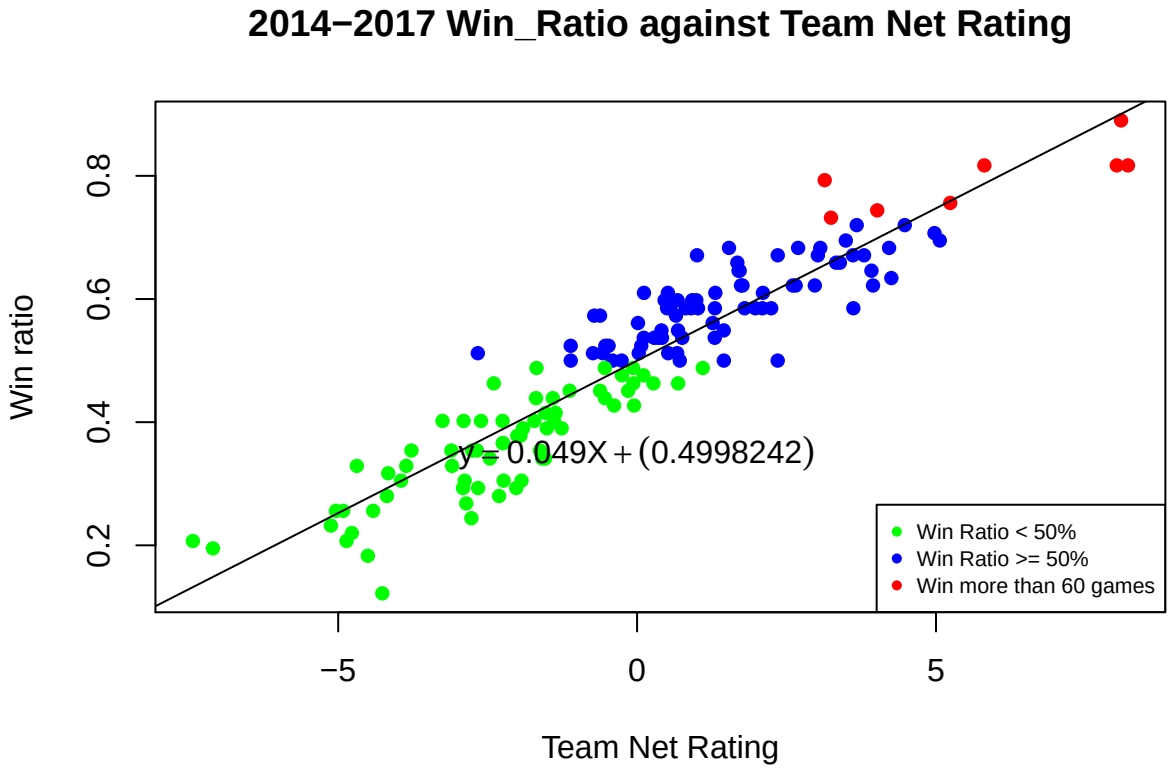
```
##
## Call:
## lm(formula = win_ratio ~ net_rating, data = four_years_total)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.165760 -0.032991 -0.004223  0.040341  0.144595
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 0.500245   0.004578  109.27   <2e-16 ***
## net_rating  0.049858   0.001600   31.16   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
```

```
## Residual standard error: 0.05607 on 148 degrees of freedom
## Multiple R-squared:  0.8677, Adjusted R-squared:  0.8668
## F-statistic: 970.7 on 1 and 148 DF,  p-value: < 2.2e-16
```

```
plot(four_years_total$net_rating,four_years_total$win_ratio,xlab = 'Team Net Rating', ylab = 'Win ratio
```

```
## integer(0)
```

```
legend("bottomright",legend=c("Win Ratio < 50%", "Win Ratio >= 50%","Win more than 60 games"),
       col=c("green", "blue","red"), pch = c(16,16,16), cex = 0.7)
```

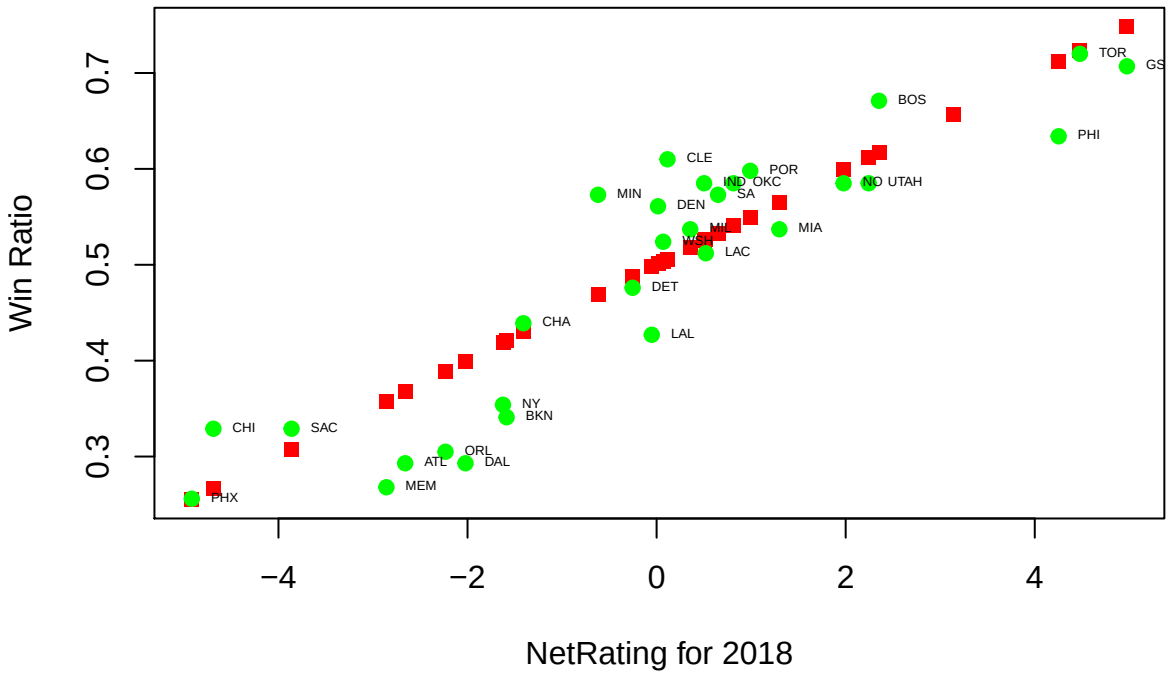


##

predict 2018 and compare with the actual results

```
x<-subset(data, season==2018, select=c(team,win_ratio,net_rating))
s18 <- data[data$season == 2018,]
pred <- predict(mod_total,s18,interval = "confidence")
data1 <-cbind(x,pred)
```

```
plot(data1$net_rating,data1$fit,pch=15,col="red",xlab = "NetRating for 2018",ylab = "Win Ratio")+points
```




```
## integer(0)
```

```
SSE <-sum((data1$fit-data1$win_ratio)^2)
SSE
```

```
## [1] 0.1188568
```

```
SSTO <- sum((data1$fit - mean(data1$win_ratio))^2)
SSTO
```

```
## [1] 0.4589247
```

```
R_square <- 1 - SSE/SSTO
R_square
```

```
## [1] 0.7410102
```

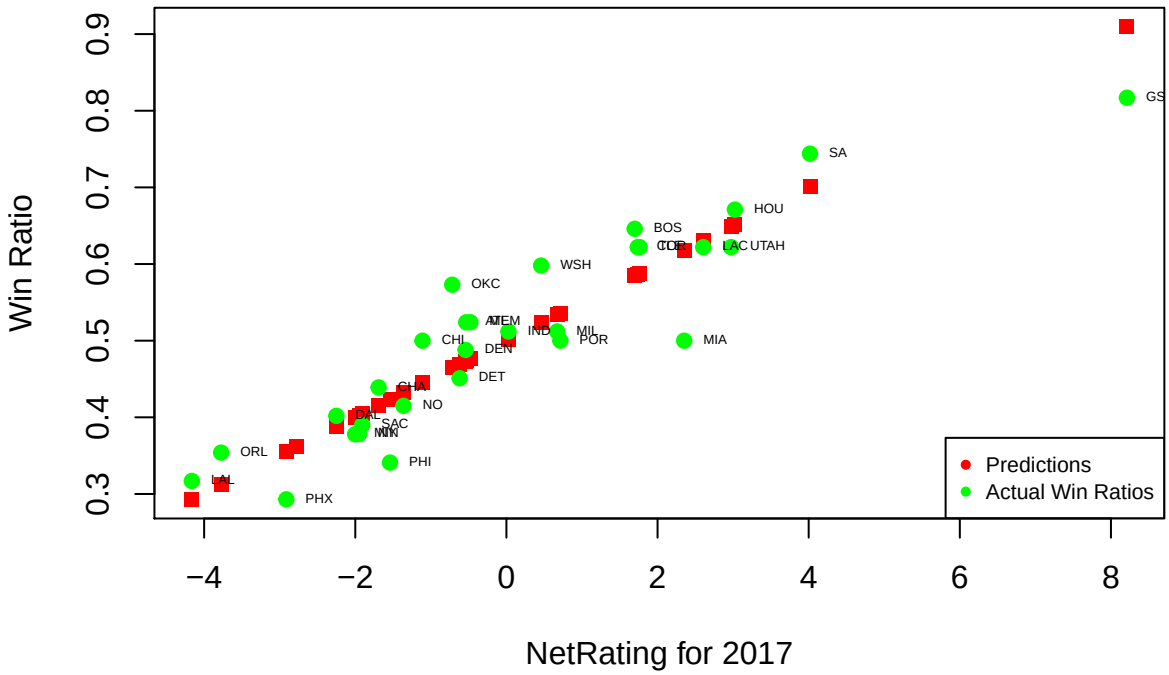
predict 2017 and compare with the actual results

```
x<-subset(data, season==2017, select=c(team,win_ratio,net_rating))
s17 <- data[data$season == 2017,]
pred <- predict(mod_total,s17,interval = "confidence")
data1 <-cbind(x,pred)
```

```
plot(data1$net_rating,data1$fit,pch=15,col="red",xlab = "NetRating for 2017",ylab = "Win Ratio")+points
```

```
## integer(0)
```

```
legend("bottomright",legend=c("Predictions","Actual Win Ratios"),
       col=c("red","green"), pch = c(16,16,16), cex = 0.7)
```




```
SSE <-sum((data1$fit-data1$win_ratio)^2)
SSE
```

```
## [1] 0.08850127
```

```
SSTO <- sum((data1$fit - mean(data1$win_ratio))^2)
SSTO
```

```
## [1] 0.4930312
```

```
R_square <- 1 - SSE/SSTO
R_square
```

```
## [1] 0.8204956
```

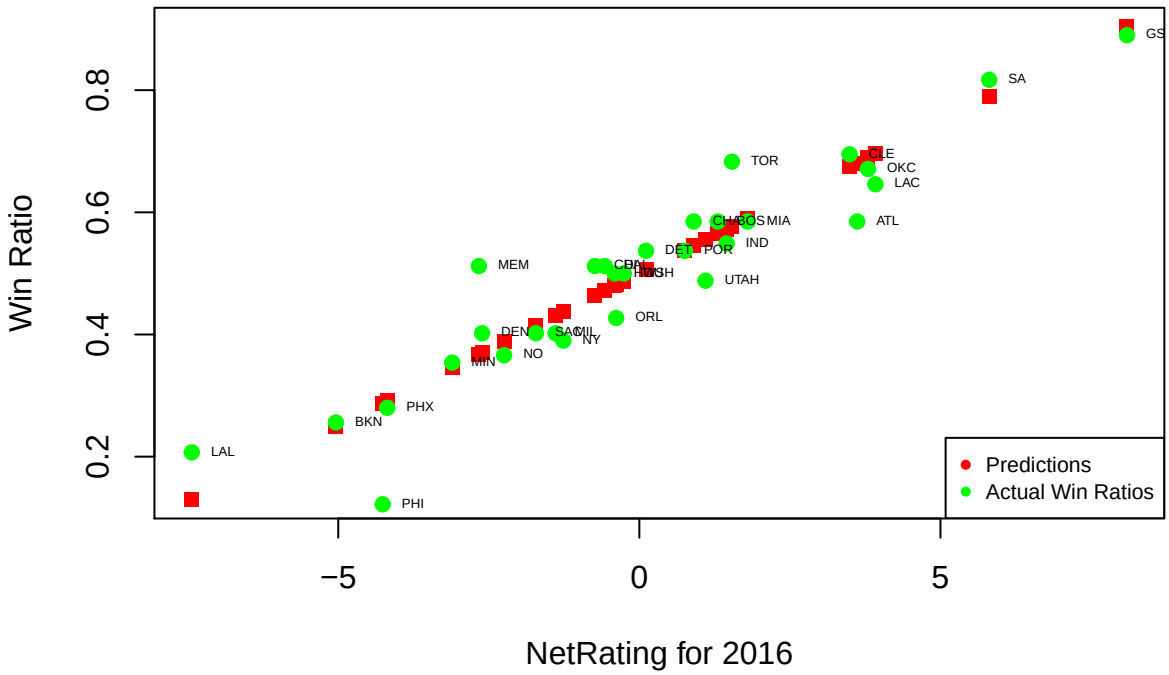
predict 2016 and compare with the actual results

```
x<-subset(data, season==2016, select=c(team,win_ratio,net_rating))
s16 <- data[data$season == 2016,]
pred <- predict(mod_total,s16,interval = "confidence")
data1 <-cbind(x,pred)
```

```
plot(data1$net_rating,data1$fit,pch=15,col="red",xlab = "NetRating for 2016",ylab = "Win Ratio")+points
```

```
## integer(0)
```

```
legend("bottomright",legend=c("Predictions","Actual Win Ratios"),
       col=c("red","green"), pch = c(16,16,16), cex = 0.7)
```




```
SSE <-sum((data1$fit-data1$win_ratio)^2)
SSE
```

```
## [1] 0.0994302
```

```
SSTO <- sum((data1$fit - mean(data1$win_ratio))^2)
SSTO
```

```
## [1] 0.792981
```

```
R_square <- 1 - SSE/SSTO
R_square
```

```
## [1] 0.8746121
```

```
mod14$coef
```

```
## (Intercept)  net_rating
##  0.50040561  0.05178442
```

```
mod15$coef
```

```
## (Intercept)  net_rating
##  0.49982420  0.04949518
```

```
mod16$coef
```

```
## (Intercept)  net_rating
##  0.50084433  0.04794289
```

```
mod17$coef
```

```
## (Intercept)  net_rating
##  0.50006959  0.04775468
```

```
summary(mod14)$r.squared
```

```
## [1] 0.8954489
```

```r
summary(mod15)$r.squared
```

```
## [1] 0.8946563
```

```r
summary(mod16)$r.squared
```

```
## [1] 0.8818368
```

```r
summary(mod17)$r.squared
```

```
## [1] 0.8377133
```

```r
sample_data <-subset(data,data$season !=2018)
sample_data
```

```
##     season       team win_ratio       toe   pace     optoe  lg_pace
## 1     2014 SA              0.756 0.5236427  95.88 0.4718391 94.82033
## 2     2014 OKC             0.720 0.5088652  96.34 0.4727062 94.82033
## 3     2014 LAC             0.695 0.5185407  96.68 0.4688749 94.82033
## 4     2014 IND             0.683 0.4816401  92.89 0.4503546 94.82033
## 5     2014 MIA             0.659 0.5270691  91.92 0.4920732 94.82033
## 6     2014 POR             0.659 0.5048991  95.92 0.4883191 94.82033
## 7     2014 HOU             0.659 0.5172414  97.30 0.4847973 94.82033
## 8     2014 GS              0.622 0.5042784  97.39 0.4658565 94.82033
## 9     2014 MEM             0.610 0.4885542  90.60 0.4831598 94.82033
## 10    2014 DAL             0.598 0.5134189  94.97 0.5042067 94.82033
## 11    2014 TOR             0.585 0.4924653  92.52 0.4820232 94.82033
## 12    2014 CHI             0.585 0.4669549  90.88 0.4611680 94.82033
## 13    2014 PHX             0.585 0.5070175  96.83 0.4864865 94.82033
## 14    2014 WSH             0.537 0.4919262  94.23 0.4877319 94.82033
## 15    2014 BKN             0.537 0.4932182  92.44 0.4902795 94.82033
## 16    2014 CHA             0.524 0.4747292  92.96 0.4860534 94.82033
## 17    2014 MIN             0.488 0.4846241  98.35 0.5008432 94.82033
## 18    2014 ATL             0.463 0.4967475  95.43 0.4973730 94.82033
## 19    2014 NY              0.451 0.4996993  91.25 0.5012484 94.82033
## 20    2014 DEN             0.439 0.4880000  98.96 0.4931507 94.82033
## 21    2014 NO              0.415 0.4877319  93.09 0.5033516 94.82033
## 22    2014 CLE             0.402 0.4682448  93.72 0.5011806 94.82033
## 23    2014 DET             0.354 0.4799542  95.58 0.5073314 94.82033
## 24    2014 SAC             0.341 0.4781306  95.05 0.5026898 94.82033
## 25    2014 LAL             0.329 0.4842342  99.72 0.5136642 94.82033
## 26    2014 UTAH            0.305 0.4729242  92.34 0.5134639 94.82033
## 27    2014 BOS             0.305 0.4694836  94.28 0.4984839 94.82033
## 28    2014 ORL             0.280 0.4744268  94.32 0.4976359 94.82033
## 29    2014 PHI             0.232 0.4607679  99.78 0.5094448 94.82033
## 30    2014 MIL             0.183 0.4648553  92.99 0.5107784 94.82033
## 31    2015 GS              0.817 0.5319751  99.29 0.4554122 94.73500
## 32    2015 ATL             0.732 0.5127900  94.66 0.4803288 94.73500
## 33    2015 HOU             0.683 0.4985406  97.38 0.4723331 94.73500
## 34    2015 LAC             0.683 0.5260355  95.41 0.4841791 94.73500
## 35    2015 MEM             0.671 0.4861613  92.85 0.4759358 94.73500
## 36    2015 SA              0.671 0.5082256  94.48 0.4720280 94.73500
## 37    2015 CLE             0.646 0.5092317  93.23 0.4917937 94.73500
## 38    2015 POR             0.622 0.5020150  95.08 0.4755887 94.73500
## 39    2015 CHI             0.610 0.4847579  93.81 0.4715354 94.73500
## 40    2015 DAL             0.610 0.5083477  96.41 0.4876649 94.73500
```

```
## 41    2015 TOR          0.598 0.5056716  93.50 0.4988081 94.73500
## 42    2015 WSH          0.561 0.4867569  94.33 0.4740566 94.73500
## 43    2015 NO           0.549 0.4964072  92.29 0.4922249 94.73500
## 44    2015 OKC          0.549 0.4877073  96.64 0.4809908 94.73500
## 45    2015 MIL          0.500 0.4845972  94.85 0.4701007 94.73500
## 46    2015 BOS          0.488 0.4820225  96.74 0.4826790 94.73500
## 47    2015 PHX          0.476 0.4886493  97.22 0.4875938 94.73500
## 48    2015 IND          0.463 0.4789318  93.70 0.4719905 94.73500
## 49    2015 UTAH         0.463 0.4866210  91.10 0.4837722 94.73500
## 50    2015 BKN          0.463 0.4808033  93.72 0.5050326 94.73500
## 51    2015 MIA          0.451 0.4849246  91.49 0.4965986 94.73500
## 52    2015 CHA          0.402 0.4551358  93.47 0.4845422 94.73500
## 53    2015 DET          0.390 0.4800000  93.70 0.4952550 94.73500
## 54    2015 DEN          0.366 0.4733862  96.89 0.4953326 94.73500
## 55    2015 SAC          0.354 0.4761905  96.19 0.5025758 94.73500
## 56    2015 ORL          0.305 0.4803288  94.52 0.4997036 94.73500
## 57    2015 LAL          0.256 0.4679005  95.08 0.5118906 94.73500
## 58    2015 PHI          0.220 0.4406977  96.48 0.4875445 94.73500
## 59    2015 NY           0.207 0.4564315  92.08 0.5064457 94.73500
## 60    2015 MIN          0.195 0.4584561  95.46 0.5288630 94.73500
## 61    2016 GS           0.890 0.5480447 100.27 0.4700714 96.56067
## 62    2016 SA           0.817 0.5205970  94.54 0.4612655 96.56067
## 63    2016 CLE          0.695 0.5173224  93.84 0.4813940 96.56067
## 64    2016 TOR          0.683 0.4984802  93.36 0.4825753 96.56067
## 65    2016 OKC          0.671 0.5183276  97.55 0.4807475 96.56067
## 66    2016 LAC          0.646 0.5115590  96.83 0.4724638 96.56067
## 67    2016 MIA          0.585 0.4984967  94.16 0.4800469 96.56067
## 68    2016 BOS          0.585 0.4868642  99.44 0.4742268 96.56067
## 69    2016 CHA          0.585 0.4932945  96.27 0.4842407 96.56067
## 70    2016 ATL          0.585 0.5002872  97.69 0.4645270 96.56067
## 71    2016 IND          0.549 0.4873418  97.38 0.4729653 96.56067
## 72    2016 DET          0.537 0.4915942  95.78 0.4904679 96.56067
## 73    2016 POR          0.537 0.5040230  97.09 0.4965076 96.56067
## 74    2016 DAL          0.512 0.4907193  95.38 0.4965238 96.56067
## 75    2016 CHI          0.512 0.4802483  96.63 0.4876265 96.56067
## 76    2016 MEM          0.512 0.4704841  93.97 0.4978619 96.56067
## 77    2016 HOU          0.500 0.4988413  98.44 0.5028670 96.56067
## 78    2016 WSH          0.500 0.4957555  99.16 0.4982699 96.56067
## 79    2016 UTAH         0.488 0.4941825  91.70 0.4826113 96.56067
## 80    2016 ORL          0.427 0.4903955  96.70 0.4942330 96.56067
## 81    2016 MIL          0.402 0.4835294  95.06 0.4976553 96.56067
## 82    2016 DEN          0.402 0.4784854  96.36 0.5046512 96.56067
## 83    2016 SAC          0.402 0.4932735 100.68 0.5097602 96.56067
## 84    2016 NY           0.390 0.4768056  94.15 0.4897481 96.56067
## 85    2016 NO           0.366 0.4885845  97.65 0.5107872 96.56067
## 86    2016 MIN          0.354 0.4832736  96.22 0.5144509 96.56067
## 87    2016 PHX          0.280 0.4701240  99.39 0.5107872 96.56067
## 88    2016 BKN          0.256 0.4750716  96.13 0.5256780 96.56067
## 89    2016 LAL          0.207 0.4506066  96.55 0.5249267 96.56067
## 90    2016 PHI          0.122 0.4634146  98.45 0.5052144 96.56067
## 91    2017 GS           0.817 0.5507572 100.38 0.4714208 96.98067
## 92    2017 SA           0.744 0.5192194  94.89 0.4781596 96.98067
## 93    2017 HOU          0.671 0.5349099 100.56 0.5057471 96.98067
## 94    2017 BOS          0.646 0.5120551  97.21 0.4951177 96.98067
```

```
## 95    2017 UTAH          0.622 0.5122549  92.11 0.4809524 96.98067
## 96    2017 TOR           0.622 0.5123384  95.42 0.4944150 96.98067
## 97    2017 CLE           0.622 0.5305889  96.75 0.5131653 96.98067
## 98    2017 LAC           0.622 0.5243688  96.78 0.4982699 96.98067
## 99    2017 WSH           0.598 0.5174746  97.97 0.5129236 96.98067
## 100   2017 OKC           0.573 0.4940644  98.21 0.5011494 96.98067
## 101   2017 MEM           0.524 0.4863905  92.87 0.4913687 96.98067
## 102   2017 ATL           0.524 0.4927620  97.79 0.4980304 96.98067
## 103   2017 IND           0.512 0.5017261  96.59 0.5014510 96.98067
## 104   2017 MIL           0.512 0.5115453  95.00 0.5046948 96.98067
## 105   2017 CHI           0.500 0.4856816  96.04 0.4968909 96.98067
## 106   2017 POR           0.500 0.5099829  97.40 0.5028670 96.98067
## 107   2017 MIA           0.500 0.5113438  95.53 0.4874636 96.98067
## 108   2017 DEN           0.488 0.5215928  99.20 0.5268757 96.98067
## 109   2017 DET           0.451 0.4954700  95.54 0.5017503 96.98067
## 110   2017 CHA           0.439 0.4962231  95.95 0.5133144 96.98067
## 111   2017 NO            0.415 0.4899329  98.59 0.5033296 96.98067
## 112   2017 DAL           0.402 0.4922986  93.12 0.5157310 96.98067
## 113   2017 SAC           0.390 0.4970692  95.45 0.5164512 96.98067
## 114   2017 MIN           0.378 0.5040888  95.49 0.5244261 96.98067
## 115   2017 NY            0.378 0.4916013  96.66 0.5111748 96.98067
## 116   2017 ORL           0.354 0.4797069  96.92 0.5174785 96.98067
## 117   2017 PHI           0.341 0.4863481  99.09 0.5014229 96.98067
## 118   2017 LAL           0.317 0.4880089  99.27 0.5286769 96.98067
## 119   2017 PHX           0.293 0.4844617 100.88 0.5124575 96.98067
## 120   2017 BKN           0.244 0.4848315 101.76 0.5112452 96.98067
##      net_rating color
## 1    5.23825844   red
## 2    3.67386072  blue
## 3    5.06398121  blue
## 4    3.06486323  blue
## 5    3.39254589  blue
## 6    1.67723321  blue
## 7    3.32925341  blue
## 8    3.94631463  blue
## 9    0.51542909  blue
## 10   0.92267131  blue
## 11   1.01888574  blue
## 12   0.55464585  blue
## 13   2.09662022  blue
## 14   0.41681723  blue
## 15   0.28650096  blue
## 16  -1.11019953  blue
## 17  -1.68227743 green
## 18  -0.06295651 green
## 19  -0.14907716 green
## 20  -0.53755536 green
## 21  -1.53346806 green
## 22  -3.25536332 green
## 23  -2.75965336 green
## 24  -2.46186471 green
## 25  -3.09507577 green
## 26  -3.94792571 green
## 27  -2.88351031 green
```

```
## 28  -2.30866597 green
## 29  -5.12229025 green
## 30  -4.50366940 green
## 31   8.02441295   red
## 32   3.24354755   red
## 33   2.69391400  blue
## 34   4.21546311  blue
## 35   1.00219612  blue
## 36   3.61002110  blue
## 37   1.71609885  blue
## 38   2.65224636  blue
## 39   1.30934487  blue
## 40   2.10484706  blue
## 41   0.67740613  blue
## 42   1.26460172  blue
## 43   0.40743648  blue
## 44   0.68515377  blue
## 45   1.45141026  blue
## 46  -0.06704065 green
## 47   0.10831809 green
## 48   0.68654509 green
## 49   0.27395242 green
## 50  -2.39696613 green
## 51  -1.12741411 green
## 52  -2.90137731 green
## 53  -1.50883769 green
## 54  -2.24456007 green
## 55  -2.67906113 green
## 56  -1.93308136 green
## 57  -4.41502862 green
## 58  -4.77097186 green
## 59  -4.86124635 green
## 60  -7.09456934 green
## 61   8.09686118   red
## 62   5.80898973   red
## 63   3.49160885  blue
## 64   1.53777371  blue
## 65   3.79651897  blue
## 66   3.92042592  blue
## 67   1.79910515  blue
## 68   1.30141982  blue
## 69   0.90265193  blue
## 70   3.61784002  blue
## 71   1.44984773  blue
## 72   0.11171597  blue
## 73   0.75566202  blue
## 74  -0.57335241  blue
## 75  -0.73835375  blue
## 76  -2.66433485  blue
## 77  -0.41040725  blue
## 78  -0.25820634  blue
## 79   1.09886704 green
## 80  -0.38430445 green
## 81  -1.39063889 green
```

```
## 82  -2.61114165 green
## 83  -1.71899652 green
## 84  -1.26193465 green
## 85  -2.24531732 green
## 86  -3.10672773 green
## 87  -4.18546336 green
## 88  -5.03806741 green
## 89  -7.43118906 green
## 90  -4.26175998 green
## 91   8.21172271   red
## 92   4.01746666   red
## 93   3.02391147  blue
## 94   1.69774130  blue
## 95   2.97304124  blue
## 96   1.76349425  blue
## 97   1.73822000  blue
## 98   2.60448626  blue
## 99   0.45974531  blue
## 100 -0.71747918  blue
## 101 -0.47671420  blue
## 102 -0.53123394  blue
## 103  0.02740559  blue
## 104  0.67105483  blue
## 105 -1.11006157  blue
## 106  0.71466824  blue
## 107  2.35230396  blue
## 108 -0.54037737 green
## 109 -0.61870079 green
## 110 -1.69096827 green
## 111 -1.36190585 green
## 112 -2.24996384 green
## 113 -1.90761549 green
## 114 -2.00247475 green
## 115 -1.95087217 green
## 116 -3.77480047 green
## 117 -1.54026336 green
## 118 -4.16279757 green
## 119 -2.91214591 green
## 120 -2.77154430 green
```

```r
# training and testing data using "new_net_rating"

set.seed(1)  # setting seed to reproduce results of random sampling

trainingRowIndex <- sample(1:nrow(sample_data), 0.833*nrow(sample_data))  # row incices for training da

trainingData <- sample_data[trainingRowIndex, ]  # model training data

testData  <- sample_data[-trainingRowIndex, ]   # test data

train_new_toe<- lm(win_ratio ~ net_rating, data=trainingData)  # build the model

predict_new_toe <- predict(train_new_toe, testData)  # predict
```

```r
summary(train_new_toe) # model summary
```

```
##
## Call:
## lm(formula = win_ratio ~ net_rating, data = trainingData)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.162641 -0.032062 -0.004896  0.038822  0.146060
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 0.501537   0.005519   90.87   <2e-16 ***
## net_rating  0.050893   0.002021   25.18   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.0549 on 97 degrees of freedom
## Multiple R-squared:  0.8673, Adjusted R-squared:  0.8659
## F-statistic:   634 on 1 and 97 DF,  p-value: < 2.2e-16
```

```r
# Calculate: akaike information criterion
AIC(train_new_toe)
```

```
## [1] -289.7162
```

```r
actuals_preds_new <- data.frame(cbind(actuals=testData$win_ratio, predicteds=predict_new_toe))
# make actuals_predicteds dataframe.

correlation_accuracy_new <- cor(actuals_preds_new)
correlation_accuracy_new
```

```
##              actuals predicteds
## actuals    1.0000000  0.9648473
## predicteds 0.9648473  1.0000000
```

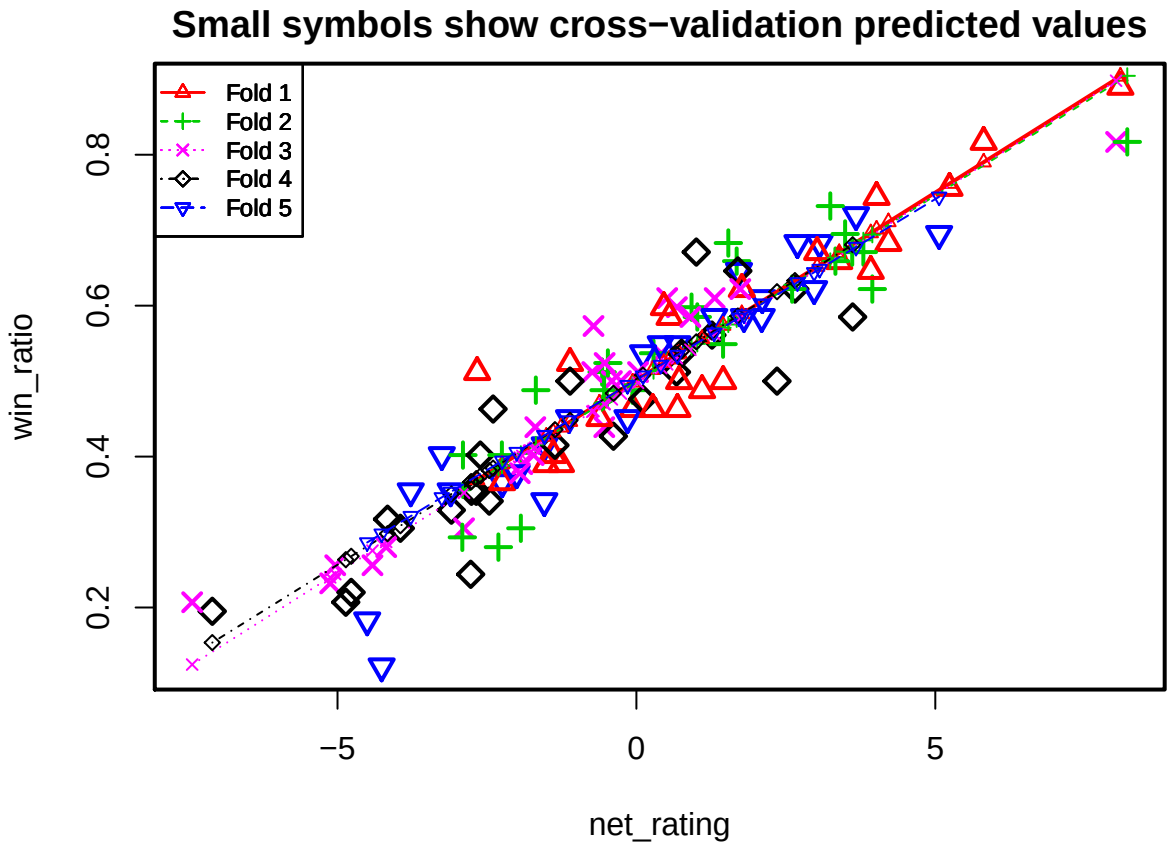
5 - Fold Cross Validation - new net rating

```r
library(DAAG)
```

```
## Loading required package: lattice
```

```r
sample_data <-subset(data,data$season !=2018)
cv.lm(sample_data, form.lm = formula(win_ratio ~ net_rating), m=5, dots = FALSE, seed=123, plotit=TRUE,
```

```
## Analysis of Variance Table
##
## Response: win_ratio
##             Df Sum Sq Mean Sq F value Pr(>F)
## net_rating   1  2.527   2.527     862 <2e-16 ***
## Residuals  118  0.346   0.003
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```




```
##
## fold 1
## Observations in test set: 24
##                    1      5    12      16      18      34      45      48
## net_rating    5.2383  3.393 0.555 -1.1102 -0.0630  4.2155  1.4514  0.6865
## cvpred        0.7617  0.670 0.529  0.4462  0.4983  0.7109  0.5735  0.5355
## win_ratio     0.7560  0.659 0.585  0.5240  0.4630  0.6830  0.5000  0.4630
## CV residual -0.0057 -0.011 0.056  0.0778 -0.0353 -0.0279 -0.0735 -0.0725
##                   49      53      61     62      66     76     79      81
## net_rating    0.274 -1.5088  8.0969 5.8090  3.9204 -2.664  1.099 -1.3906
## cvpred        0.515  0.4264  0.9037 0.7901  0.6962  0.369  0.556  0.4323
## win_ratio     0.463  0.3900  0.8900 0.8170  0.6460  0.512  0.488  0.4020
## CV residual -0.052 -0.0364 -0.0137 0.0269 -0.0502  0.143 -0.068 -0.0303
##                   84      85     92     93     96     99     106     109
## net_rating  -1.2619 -2.2453 4.017 3.0239 1.763 0.4597  0.7147 -0.6187
## cvpred        0.4387  0.3898 0.701 0.6517 0.589 0.5242  0.5369  0.4707
## win_ratio     0.3900  0.3660 0.744 0.6710 0.622 0.5980  0.5000  0.4510
## CV residual -0.0487 -0.0238 0.043 0.0193 0.033 0.0738 -0.0369 -0.0197
##
## Sum of squares = 0.07    Mean square = 0    n = 24
##
## fold 2
## Observations in test set: 24
##                    6        7       8      10      11      15      17      28
## net_rating   1.6772  3.32925  3.9463 0.9227 1.0189 0.2865 -1.682 -2.309
## cvpred       0.5819  0.66347  0.6939 0.5446 0.5494 0.5132  0.416  0.385
## win_ratio    0.6590  0.65900  0.6220 0.5980 0.5850 0.5370  0.488  0.280
```

```
## CV residual 0.0771 -0.00447 -0.0719 0.0534 0.0356 0.0238  0.072 -0.105
##                 32       36       46      52      56     63    64      65
## net_rating  3.2435  3.61002 -0.06704 -2.9014 -1.9331 3.4916 1.538  3.7965
## cvpred      0.6592  0.67733  0.49577  0.3558  0.4036 0.6715 0.575  0.6865
## win_ratio   0.7320  0.67100  0.48800  0.4020  0.3050 0.6950 0.683  0.6710
## CV residual 0.0728 -0.00633 -0.00777  0.0462 -0.0986 0.0235 0.108 -0.0155
##                 71      74      91       98     101     108    112
## net_rating   1.4498 -0.5734  8.2117  2.60449 -0.4767 -0.5404 -2.250
## cvpred       0.5707  0.4708  0.9045  0.62768  0.4755  0.4724  0.388
## win_ratio    0.5490  0.5120  0.8170  0.62200  0.5240  0.4880  0.402
## CV residual -0.0217  0.0412 -0.0875 -0.00568  0.0485  0.0156  0.014
##                119
## net_rating  -2.9121
## cvpred       0.3553
## win_ratio    0.2930
## CV residual -0.0623
##
## Sum of squares = 0.08    Mean square = 0    n = 24
##
## fold 3
## Observations in test set: 24
##                  9     14      20     27       29      31     39     41
## net_rating  0.5154 0.4168 -0.5376 -2.884 -5.12229  8.0244 1.3093 0.6774
## cvpred      0.5222 0.5173  0.4695  0.352  0.23998  0.8981 0.5619 0.5303
## win_ratio   0.6100 0.5370  0.4390  0.305  0.23200  0.8170 0.6100 0.5980
## CV residual 0.0878 0.0197 -0.0305 -0.047 -0.00798 -0.0811 0.0481 0.0677
##                 57     69      75      77      78       83       87
## net_rating  -4.4150 0.9027 -0.7384 -0.4104 -0.2582 -1.71900 -4.18546
## cvpred       0.2754 0.5416  0.4594  0.4758  0.4835  0.41034  0.28687
## win_ratio    0.2560 0.5850  0.5120  0.5000  0.5000  0.40200  0.28000
## CV residual -0.0194 0.0434  0.0526  0.0242  0.0165 -0.00834 -0.00687
##                 88      89     97    100     102    103     110     113
## net_rating  -5.0381 -7.4312 1.7382 -0.717 -0.5312 0.0274 -1.6910 -1.9076
## cvpred       0.2442  0.1244 0.5834  0.460  0.4698 0.4978  0.4117  0.4009
## win_ratio    0.2560  0.2070 0.6220  0.573  0.5240 0.5120  0.4390  0.3900
## CV residual  0.0118  0.0826 0.0386  0.113  0.0542 0.0142  0.0273 -0.0109
##                115
## net_rating  -1.9509
## cvpred       0.3987
## win_ratio    0.3780
## CV residual -0.0207
##
## Sum of squares = 0.06    Mean square = 0    n = 24
##
## fold 4
## Observations in test set: 24
##                 23      24      25       26    35      38       42     47
## net_rating  -2.7597 -2.4619 -3.0951 -3.94793 1.002  2.6522  1.26460  0.108
## cvpred       0.3669  0.3815  0.3504  0.30839 0.552  0.6332  0.56495  0.508
## win_ratio    0.3540  0.3410  0.3290  0.30500 0.671  0.6220  0.56100  0.476
## CV residual -0.0129 -0.0405 -0.0214 -0.00339 0.119 -0.0112 -0.00395 -0.032
##                 50      55      58      59      60      70      73
## net_rating  -2.3970 -2.6791 -4.7710 -4.8612 -7.0946  3.6178  0.7557
## cvpred       0.3847  0.3708  0.2679  0.2634  0.1535  0.6808  0.5399
```

```
## win_ratio    0.4630  0.3540  0.2200  0.2070  0.1950  0.5850  0.5370
## CV residual  0.0783 -0.0168 -0.0479 -0.0564  0.0415 -0.0958 -0.0029
##                  80      82     94     104     105    107     111     118
## net_rating  -0.3843 -2.6111 1.6977  0.6711 -1.1101  2.352 -1.3619 -4.1628
## cvpred       0.4838  0.3742 0.5863  0.5357  0.4481  0.618  0.4357  0.2978
## win_ratio    0.4270  0.4020 0.6460  0.5120  0.5000  0.500  0.4150  0.3170
## CV residual -0.0568  0.0278 0.0597 -0.0237  0.0519 -0.118 -0.0207  0.0192
##                 120
## net_rating  -2.772
## cvpred       0.366
## win_ratio    0.244
## CV residual -0.122
##
## Sum of squares = 0.08    Mean square = 0    n = 24
##
## fold 5
## Observations in test set: 24
##                   2       3      4      13      19      21     22     30
## net_rating  3.6739  5.0640 3.0649  2.0966 -0.1491 -1.5335 -3.255 -4.504
## cvpred      0.6777  0.7443 0.6486  0.6022  0.4947  0.4284  0.346  0.286
## win_ratio   0.7200  0.6950 0.6830  0.5850  0.4510  0.4150  0.402  0.183
## CV residual 0.0423 -0.0493 0.0344 -0.0172 -0.0437 -0.0134  0.056 -0.103
##                  33    37      40     43     44       51      54       67
## net_rating  2.6939 1.716 2.10485 0.4074 0.6852 -1.12741 -2.2446  1.79911
## cvpred      0.6308 0.584 0.60261 0.5213 0.5346  0.44787  0.3944  0.58798
## win_ratio   0.6830 0.646 0.61000 0.5490 0.5490  0.45100  0.3660  0.58500
## CV residual 0.0522 0.062 0.00739 0.0277 0.0144  0.00313 -0.0284 -0.00298
##                  68     72        86     90      95     114     116     117
## net_rating  1.3014 0.1117 -3.106728 -4.262  2.9730 -2.002 -3.7748 -1.5403
## cvpred      0.5641 0.5072  0.353107  0.298  0.6442  0.406  0.3211  0.4281
## win_ratio   0.5850 0.5370  0.354000  0.122  0.6220  0.378  0.3540  0.3410
## CV residual 0.0209 0.0298  0.000893 -0.176 -0.0222 -0.028  0.0329 -0.0871
##
## Sum of squares = 0.07    Mean square = 0    n = 24
##
## Overall (Sum over all 24 folds)
##      ms
## 0.00297
```

predict 2018 and compare with the actual results

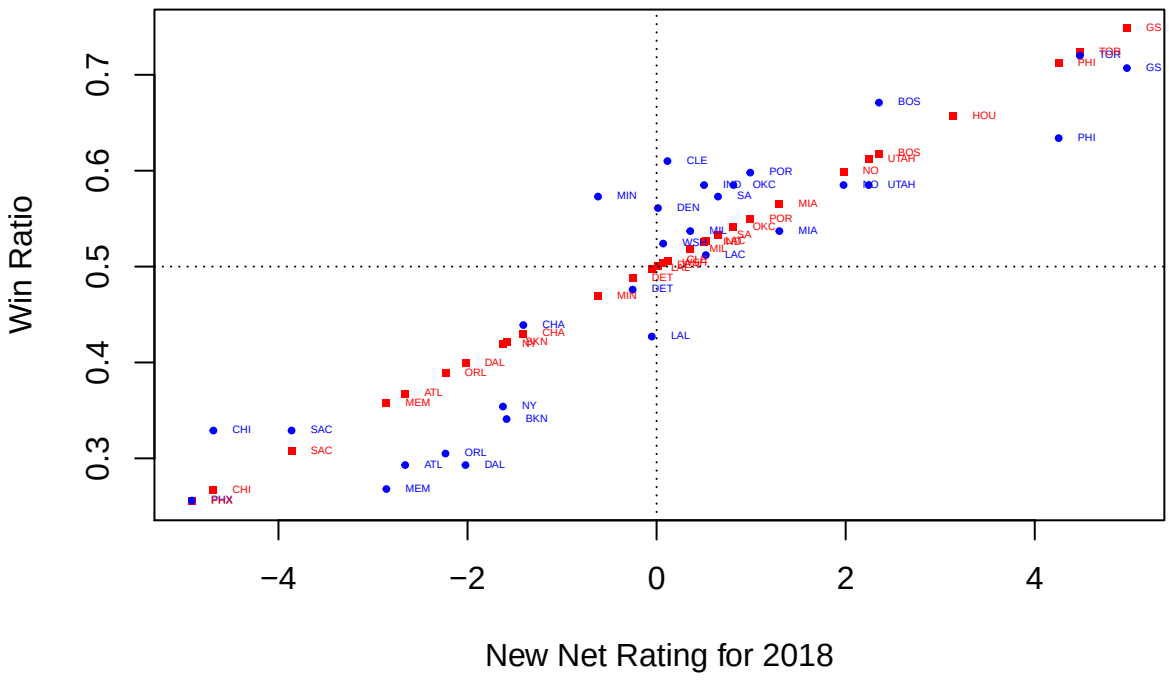
using new net rating

```
x<-subset(data, season==2018, select=c(team,win_ratio,net_rating))
s18 <- data[data$season == 2018,]
pred <- predict(mod_total,s18,interval = "confidence")
data1 <-cbind(x,pred)

ranking <- subset(data1,select=c(team,fit))
ordered_data <- ranking[order(-ranking$fit),]
ordered_data
```

```
##           team   fit
## 123 GS       0.748
## 122 TOR      0.724
## 125 PHI      0.712
## 121 HOU      0.657
## 124 BOS      0.618
## 131 UTAH     0.612
## 130 NO       0.599
## 135 MIA      0.565
## 127 POR      0.550
## 129 OKC      0.541
## 132 SA       0.533
## 138 LAC      0.526
## 128 IND      0.525
## 136 MIL      0.518
## 126 CLE      0.506
## 137 WSH      0.504
## 134 DEN      0.501
## 141 LAL      0.498
## 139 DET      0.488
## 133 MIN      0.469
## 140 CHA      0.430
## 143 BKN      0.421
## 142 NY       0.419
## 147 DAL      0.399
## 146 ORL      0.389
## 148 ATL      0.368
## 149 MEM      0.358
## 144 SAC      0.308
## 145 CHI      0.266
## 150 PHX      0.255
```

```
plot(data1$net_rating,data1$fit,pch=15,col="red",xlab = "New Net Rating for 2018",ylab = "Win Ratio",ce
```



```
## integer(0)
```

```
SSE <-sum((data1$fit-data1$win_ratio)^2)
SSE
```

```
## [1] 0.119
```

```
SSTO <- sum((data1$win_ratio - mean(data1$win_ratio))^2)
SSTO
```

```
## [1] 0.645
```

```
R_square <- 1 - SSE/SSTO
R_square
```

```
## [1] 0.816
```