Multiple Linear Regression & Calculation of R^2 - Moneyball

Markus Stellwag

11 12 2019

Table of Contents

1 Context and Goal of the Analysis

In this analysis we consider the data set Moneyball. Main objective is to create predictive models and check, if the model fit for different predictions by comparing them to the data given. Besides, we want to calculate R squared and find models with high accuracy.

2 Data Analysis

```
library(dplyr)

library(ggplot2)
library(lattice)
library(mice)
```

Firstly, we need to read the file and see what kind of variables and information we have. Fields are separated with a comma and decimals with a dot. Now we have a look to the data frame to see the characteristics of our dataset.Once checked if the information of the files are ok, we have a look at the head and the tails of the file to see if the information are consistent. After these initial steps we are ready to start analyzing the data and create our predictive models.

```
baseball <- read.csv("baseball.csv", header = TRUE, sep = ",", dec = ".")

str(baseball)

## 'data.frame':    1232 obs. of  15 variables:
##  $ Team       : Factor w/ 39 levels "ANA","ARI","ATL",..: 2 3 4 5 7 8 9 10 11 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 0.335 ...
##  $ 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 0.268 ...
##  $ 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 0.402 ...

head(baseball)

##   Team League Year  RS  RA  W   OBP   SLG    BA Playoffs RankSeason
## 1  ARI     NL 2012 734 688 81 0.328 0.418 0.259        0         NA
## 2  ATL     NL 2012 700 600 94 0.320 0.389 0.247        1          4
## 3  BAL     AL 2012 712 705 93 0.311 0.417 0.247        1          5
## 4  BOS     AL 2012 734 806 69 0.315 0.415 0.260        0         NA
## 5  CHC     NL 2012 613 759 61 0.302 0.378 0.240        0         NA
## 6  CHW     AL 2012 748 676 85 0.318 0.422 0.255        0         NA
##   RankPlayoffs   G  OOBP  OSLG
## 1           NA 162 0.317 0.415
## 2            5 162 0.306 0.378
## 3            4 162 0.315 0.403
## 4           NA 162 0.331 0.428
## 5           NA 162 0.335 0.424
## 6           NA 162 0.319 0.405
```

```
tail(baseball)
```

```
##      Team League Year  RS  RA   W   OBP   SLG    BA Playoffs RankSeason
## 1227  NYY     AL 1962 817 680  96 0.337 0.426 0.267        1          2
## 1228  PHI     NL 1962 705 759  81 0.330 0.390 0.260        0         NA
## 1229  PIT     NL 1962 706 626  93 0.321 0.394 0.268        0         NA
## 1230  SFG     NL 1962 878 690 103 0.341 0.441 0.278        1          1
## 1231  STL     NL 1962 774 664  84 0.335 0.394 0.271        0         NA
## 1232  WSA     AL 1962 599 716  60 0.308 0.373 0.250        0         NA
##      RankPlayoffs   G OOBP OSLG
## 1227            1 162   NA   NA
## 1228           NA 161   NA   NA
## 1229           NA 161   NA   NA
## 1230            2 165   NA   NA
## 1231           NA 163   NA   NA
## 1232           NA 162   NA   NA
```

In the first part of this analysis we filter for the team Oakland.

```
oakland = filter(baseball, Team == "OAK")
```

Followed by that we create a plot to visualize the recordings of the Oakland A’s.

```
oak.wins.plot <- ggplot(data = oakland, aes(x=Year, y=W)) +
                 geom_line() +
                 labs(title="Oakland A's - % Wins per Year", x='Years', y='Wins')
print(oak.wins.plot)
```

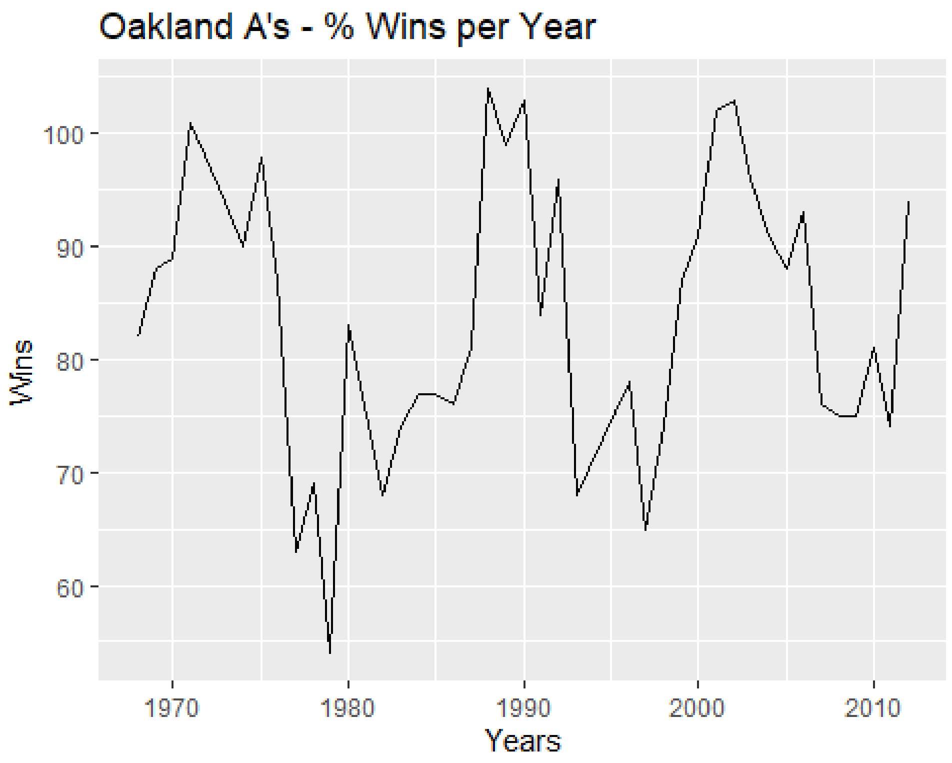

Figure 1: Oakland A's - % Wins per Year

The variable "% Wins/Year" labels the Y-axis, the variable "Years" the X-axis. Starting from the year 1968 until 2012 the graph reflects percentages of wins/ year at one glance. In the next step we want to analyze the period 1997-2001 as there the line graph skyrockets.

```
oak.1997 = filter(oakland, Year>=1997)

oak.range = filter(oak.1997, Year<=2001)

oak.97.02 = xyplot(W ~ Year, data = oak.range, xlab = "Years", ylab = "% Wins/Year",
     main = "Oakland A's - % Wins per Year from 1997 - 2001", type = "l", col = "darkblue")

oak.97.02
```

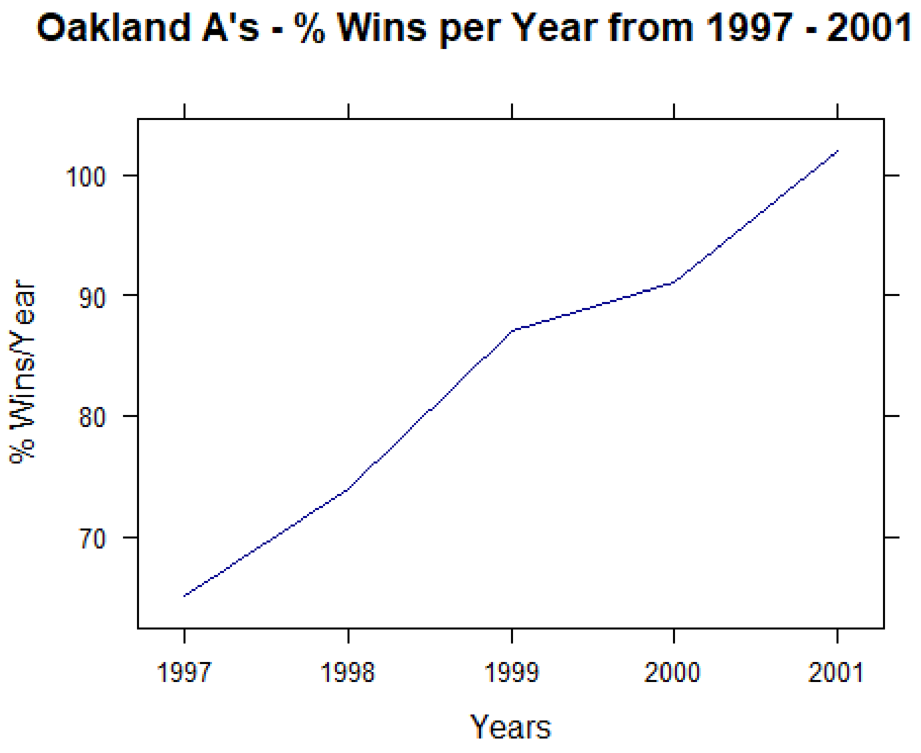

Figure 2: Oakland A's - % Wins per Year from 1997-2001

If we now consider either graphs, we can determine in the period between 1997 - 2001 a steep ascent of the graph starting at approximately 65% to more than 100%. This slope is an indication of a very successful phase of the Oakland A's compared to the rest of their history. We will keep this development of the graph in mind for our further analysis.

3 Creating subsets of data one for train (before 2002) and the other for test.

Here we create two different subsets, one training subset which comprises all data before 2002, and one subset implying all information beginning from 2002.

```
training = subset(baseball, Year < 2002, select = Team:OSLG)

test = subset(baseball, Year >= 2002, select = Team:OSLG)
```

As result we have now two separated datasets for our further approach.

4 Using runs scored and allowed to create a model for wins prediction

In this part our objective is to compile a model for prediction referring to wins. For that we could determine that our dependent variable is in this case “wins”, and the independent are “runs scored” (RS) and “runs allowed” (RA).

Accordingly, we create linear models considering RS and RA separately, and eventually together. The iterations enable us to see the level of significance of the variables which can be seen by the stars "*" on the right side.

```
w.pred1 = lm(W ~ RS, data = training)

w.pred2 = lm(W ~ RA, data = training)

w.pred3 = lm(W ~ RS + RA, data = training)
```

Now we use the summary function to call the linear models.

```
summary(w.pred1)

##
## Call:
## lm(formula = W ~ RS, data = training)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -35.499  -7.074   0.193   6.931  24.153
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 37.24223    2.49210   14.94   <2e-16 ***
## RS           0.06200    0.00351   17.66   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 9.832 on 900 degrees of freedom
## Multiple R-squared:  0.2574, Adjusted R-squared:  0.2566
## F-statistic:   312 on 1 and 900 DF,  p-value: < 2.2e-16

summary(w.pred2)

##
## Call:
## lm(formula = W ~ RA, data = training)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -27.9060  -6.6692   0.1744   6.7569  30.3763
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) 124.335359   2.479183   50.15   <2e-16 ***
## RA          -0.061741   0.003492  -17.68   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 9.829 on 900 degrees of freedom
## Multiple R-squared:  0.2578, Adjusted R-squared:  0.257
## F-statistic: 312.7 on 1 and 900 DF,  p-value: < 2.2e-16
```

```
summary(w.pred3)
```

```
##
## Call:
## lm(formula = W ~ RS + RA, data = training)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -14.2682  -2.6316   0.1189   2.9461  11.6768
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 80.531366   1.181688   68.15   <2e-16 ***
## RS           0.106016   0.001547   68.55   <2e-16 ***
## RA          -0.105519   0.001539  -68.57   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.941 on 899 degrees of freedom
## Multiple R-squared:  0.8808, Adjusted R-squared:  0.8806
## F-statistic:  3322 on 2 and 899 DF,  p-value: < 2.2e-16
```

In order to find a suitable model for our wins prediction, we have to compare the R^2 of the different models. Basically we can say, the higher the R-squared, the better the model fits your data. But we have to be careful not to overfit the models with variables, since the R^2 value increases with additional variables. Therefore, we checked the models by calling them via the summary function to see the level of significance.

w.pred1 returns a value of 0.2574 for R^2 which is according to our prior explanation very low percentage. However, we detect a high level of significance ("***") for this variable.

Considering w.pred2 we obtain the value 0.2578. As well this call depicts a high level of significance of the variable which is good for our predictive model.

If we take the combination of both independent variables into account, we receive the best result with win.pred3, having a R^2 value of 0.8808. Here we can see an improvement by adding the both variables to the model. Hence, we decide to process with model win.pred3, since we couldn't detect any multicolliniarity.

5 Using the model on test data and compare the R² with the one obtained on the training data

In the following section we create our predictive model for wins by processing with the chosen model as explained in task 3.

```
prediction.w = predict(w.pred3, newdata = test)

head(prediction.w, 10)

##        1        2        3        4        5        6        7        8 
## 85.75027 91.43139 81.62408 73.29901 65.43043 88.50073 89.41111 62.08067 
##        9       10 
## 66.97980 86.80148

tail(prediction.w, 10)

##      321      322      323      324      325      326      327      328 
## 96.33500 79.40719 71.45894 64.71616 93.07087 98.54245 95.58990 55.01387 
##      329      330 
## 76.83535 79.35289
```

By calling prediction.w we receive a set of numbers referring to wins prediction. As next step we need to compare those with actual data, in this case the our "test", column Wins.

```
head(test$W, 10)

##  [1] 81 94 93 69 61 85 97 68 64 88

tail(test$W, 10)

##  [1] 103  80  72  66  93  95  97  55  72  78
```

The numbers are partly very close. We evaluate this predictive model as accurate. For further investigations we want to calculate R^2 in order to see the difference between the R^2 of our linear model and the predictive model. Following has been used to receive a comparing value.

```
SSE.1 <- sum((test$W - prediction.w) ^ 2)

SST.1 <- sum((test$W - mean(test$W)) ^ 2)

1 - SSE.1/SST.1

## [1] 0.8757492
```

As resulting value we receive 0.8757492 as R^2 of our predictive model. If we compare now this value with R^2 of the w.pred3 (0.8808), we see a margin of approximately 0.01%. Hence, we can evaluate this model as accurate

Conclusion

According our results we have achieved almost a match with our predictive model as we could determine by comparing the R^2 in the prior part. This leads us to the conclusion that the created predictive model fits well due to its accuracy. An argument for that could be the large dataset which provided us enough information for the implementation.

6 Creating a model for predicting runs scored

In this part we created linear models considering OBP, SLG and BA separately and eventually as multiple variable regression model.

```
rs.model1 = lm(RS ~ OBP, data = training)

summary(rs.model1)


rs.model2 = lm(RS ~ SLG, data = training)

summary(rs.model2)


rs.model3 = lm(RS ~ BA, data = training)

summary(rs.model3)


rs.model4 = lm(RS ~ OBP + SLG, data = training)

summary(rs.model4)


rs.model5 = lm(RS ~ OBP + BA, data = training)

summary(rs.model5)

rs.model6 = lm(RS ~ OBP + SLG + BA, data = training)

summary(rs.model6)
```

In order to find a suitable model for our runs scored prediction, we have to compare again the multiple R^2 of the different models. Again we have to be careful not to overfit the models with variables, since the R^2 value increases with additional variables.

The models rs.model1 - rs.model3 show high significance of the variables. Now we want to proceed with multiple variables. If we take the combination of the independent variables into account, we receive the best result with rs.model4, having a R^2 value of 0.9296. Compared to rs.model6, we could detect a lower level of significance related to "BA", showed by two stars "**". Hence, we decide to process with model rs.pred3, since we couldn't detect any multicolliniarity, like in rs.model6.

In the following part we create our predictive model for runs scored by processing with the chosen model as explained on section prior.

```
rs.predict = predict(rs.model4, newdata = test)

head(rs.predict, 10)
tail(rs.predict, 10)
```

By calling rs.predict we receive a set of numbers referring to runs scored prediction. As next step we need to compare those with actual data, in this case the our "test", column runs scored ("RS").

```
head(test$RS, 10)
tail(test$RS, 10)
```

The numbers are partly very close. We evaluate this predictive model as accurate. For further investigations we want to calculate R^2 in order to see the difference between the R^2 of our linear model and the predictive model. Following has been used to receive a comparing value.

```
SSE2 <- sum((test$RS - rs.predict) ^ 2)

SSE2

SST2 <- sum((test$RS - mean(test$RS)) ^ 2)

SST2

1 - SSE2/SST2
```

As resulting value we receive 0.8821854 for R^2 of our predictive model. If we compare now this value to R^2 of the rs.predict, we see a margin of approximately 0.05%.

Conclusion

According our results we have repeatedly achieved an almost match with our predictive model as we could determine by comparing the R^2 in the prior part. This leads us to the conclusion that the created predictive model fits well due to its accuracy. We stick with the argument that the large dataset provided us enough information for the implementation of the predictive model.

7 Creating a prediction model of runs allowed.

In this part we create linear models considering OBP, SLG separately and eventually as multiple variable regression model.

```
ra.model1 = lm(RA ~ OOBP, data = training)

summary(ra.model1)


ra.model2 = lm(RA ~ OSLG, data = training)

summary(ra.model2)


ra.model3 = lm(RA ~ OOBP + OSLG, data = training)

summary(ra.model3)
```

In order to find a suitable model for our runs allowed prediction, we have to compare again the multiple R^2 of the different models.

The models ra.model1 - ra.model2 show high significance of the variables. Now we want to proceed with multiple variables. If we take the combination of the independent variables into account, we receive the best result with ra.model3, having a R^2 value of 0.9073. Hence, we decide to process with model ra.model3, since we couldn't detect any multicolliniarity.

In the following part we create our predictive model for runs allowed by processing with the chosen model as explained on section prior.

```
ra.predict = predict(ra.model3, newdata = test)

head(ra.predict, 10)
tail(ra.predict, 10)
```

By calling ra.predict we receive a set of numbers referring to runs allowed prediction. As next step we need to compare those with actual data, in this case the our "test", column runs scored ("RA").

```
head(test$RA, 10)
tail(test$RA, 10)
```

The numbers are partly close. For further investigations we want to calculate R^2 in order to see the difference between the R^2 of our linear model and the predictive model. Following has been used to receive a comparing value.

```
SSE3 <- sum((test$RA - ra.predict) ^ 2)

SSE3

SST3 <- sum((test$RA - mean(test$RA)) ^ 2)

SST3

1 - SSE3/SST3
```

As resulting value we receive 0.882031 for R^2 of our predictive model. If we compare now this value to R^2 of the rs.predict, we see a margin of approximately 0.02%.

Conclusion

According our results we have repeatedly achieved an almost match with our predictive model as we could determine by comparing the R^2 in the prior part. This leads us to the conclusion that the created predictive model fits well due to its accuracy.

Final Conclusion

In sum all results for the predictive models were pretty accurate, since we have a large dataset that provided us enough information. In one case we could determine one less significant variable after creating a multiple variable model. Otherwise we could not detect any specific anomalies in the plots which could have led us to insignificance in advance.

9 References:

Hadley Wickham, Jim Hester and Romain Francois (2017). readr: Read Rectangular Text Data. R package version 1.1.1. https://CRAN.R-project.org/package=readr

Hadley Wickham, Romain Francois, Lionel Henry and Kirill Müller (2017). dplyr: A Grammar of Data Manipulation. R package version 0.7.4. https://CRAN.R-project.org/package=dplyr

Alexander Kowarik, Matthias Templ (2016). Imputation with the R Package VIM. Journal of Statistical Software, 74(7), 1-16. doi:10.18637/jss.v074.i07

https://www.r-bloggers.com/using-linear-regression-to-predict-energy-output-of-a-power-plant/

https://www.kaggle.com/wduckett/moneyball-mlb-stats-19622012/kernels