data621_hw1_mia_wei

Wei Zhou, Mia Chen

2/29/2020

R Markdown

```
## -- Attaching packages ---------------------------------------------------------------------------

## v ggplot2 3.1.0     v purrr   0.3.2
## v tibble  2.1.1     v dplyr   0.7.8
## v tidyr   0.8.2     v stringr 1.3.1
## v readr   1.3.0     v forcats 0.3.0

## Warning: package 'tibble' was built under R version 3.5.2

## Warning: package 'purrr' was built under R version 3.5.2

## -- Conflicts ------------------------------------------------------------------------------------
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
# Read in the data
data <- read.csv("https://raw.githubusercontent.com/miachen410/DATA621/master/moneyball-training-data.c
glimpse(data)
```

```
## Observations: 2,276
## Variables: 17
## $ INDEX            <int> 1, 2, 3, 4, 5, 6, 7, 8, 11, 12, 13, 15, 16, 1...
## $ TARGET_WINS      <int> 39, 70, 86, 70, 82, 75, 80, 85, 86, 76, 78, 6...
## $ TEAM_BATTING_H   <int> 1445, 1339, 1377, 1387, 1297, 1279, 1244, 127...
## $ TEAM_BATTING_2B  <int> 194, 219, 232, 209, 186, 200, 179, 171, 197, ...
## $ TEAM_BATTING_3B  <int> 39, 22, 35, 38, 27, 36, 54, 37, 40, 18, 27, 3...
## $ TEAM_BATTING_HR  <int> 13, 190, 137, 96, 102, 92, 122, 115, 114, 96,...
## $ TEAM_BATTING_BB  <int> 143, 685, 602, 451, 472, 443, 525, 456, 447, ...
## $ TEAM_BATTING_SO  <int> 842, 1075, 917, 922, 920, 973, 1062, 1027, 92...
## $ TEAM_BASERUN_SB  <int> NA, 37, 46, 43, 49, 107, 80, 40, 69, 72, 60, ...
## $ TEAM_BASERUN_CS  <int> NA, 28, 27, 30, 39, 59, 54, 36, 27, 34, 39, 7...
## $ TEAM_BATTING_HBP <int> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, N...
## $ TEAM_PITCHING_H  <int> 9364, 1347, 1377, 1396, 1297, 1279, 1244, 128...
## $ TEAM_PITCHING_HR <int> 84, 191, 137, 97, 102, 92, 122, 116, 114, 96,...
## $ TEAM_PITCHING_BB <int> 927, 689, 602, 454, 472, 443, 525, 459, 447, ...
## $ TEAM_PITCHING_SO <int> 5456, 1082, 917, 928, 920, 973, 1062, 1033, 9...
## $ TEAM_FIELDING_E  <int> 1011, 193, 175, 164, 138, 123, 136, 112, 127,...
## $ TEAM_FIELDING_DP <int> NA, 155, 153, 156, 168, 149, 186, 136, 169, 1...
```

```
# Split data into training set and testing set by 7:3 ratio
set.seed(123)
train_ind = sample(seq_len(nrow(data)), size = nrow(data)*.7)
train = data[train_ind, ]
test = data[-train_ind, ]
```

```
glimpse(train)
```

```
## Observations: 1,593
## Variables: 17
```

```
## $ INDEX            <int> 723, 2013, 1035, 2246, 2381, 113, 1335, 2263,...
## $ TARGET_WINS      <int> 86, 64, 101, 75, 51, 67, 32, 93, 53, 102, 62,...
## $ TEAM_BATTING_H   <int> 1630, 1366, 1481, 1427, 1370, 1438, 1466, 155...
## $ TEAM_BATTING_2B  <int> 266, 273, 215, 299, 141, 232, 200, 212, 144, ...
## $ TEAM_BATTING_3B  <int> 107, 33, 53, 26, 44, 62, 88, 87, 62, 28, 41, ...
## $ TEAM_BATTING_HR  <int> 33, 111, 180, 128, 12, 36, 26, 57, 13, 166, 1...
## $ TEAM_BATTING_BB  <int> 470, 569, 673, 431, 65, 387, 376, 565, 418, 6...
## $ TEAM_BATTING_SO  <int> NA, 929, 867, 901, 551, 452, 612, 584, NA, 11...
## $ TEAM_BASERUN_SB  <int> 168, 152, 98, 71, NA, 48, 238, 366, 239, 155,...
## $ TEAM_BASERUN_CS  <int> NA, 84, 36, 35, NA, NA, NA, NA, NA, 51, 76, 6...
## $ TEAM_BATTING_HBP <int> NA, NA, NA, 49, NA, NA, NA, NA, NA, NA, NA, N...
## $ TEAM_PITCHING_H  <int> 1900, 1366, 1481, 1427, 3171, 1513, 1721, 185...
## $ TEAM_PITCHING_HR <int> 38, 111, 180, 128, 28, 38, 31, 68, 14, 166, 1...
## $ TEAM_PITCHING_BB <int> 548, 569, 673, 431, 150, 407, 441, 673, 448, ...
## $ TEAM_PITCHING_SO <int> NA, 929, 867, 901, 1275, 475, 718, 696, NA, 1...
## $ TEAM_FIELDING_E  <int> 364, 142, 146, 90, 921, 176, 686, 562, 332, 1...
## $ TEAM_FIELDING_DP <int> 98, 134, 147, 146, 86, 148, NA, NA, 74, 144, ...
```

1. DATA EXPLORATION (25 Points)

Describe the size and the variables in the moneyball training data set. Consider that too much detail will cause a manager to lose interest while too little detail will make the manager consider that you aren't doing your job. Some suggestions are given below. Please do NOT treat this as a check list of things to do to complete the assignment. You should have your own thoughts on what to tell the boss. These are just ideas.

```r
# Cleaning the column names by removing TEAMS_
names(train) <- gsub("TEAM_", "", names(train))
names(test) <- gsub("TEAM_", "", names(test))
summary(train)
```

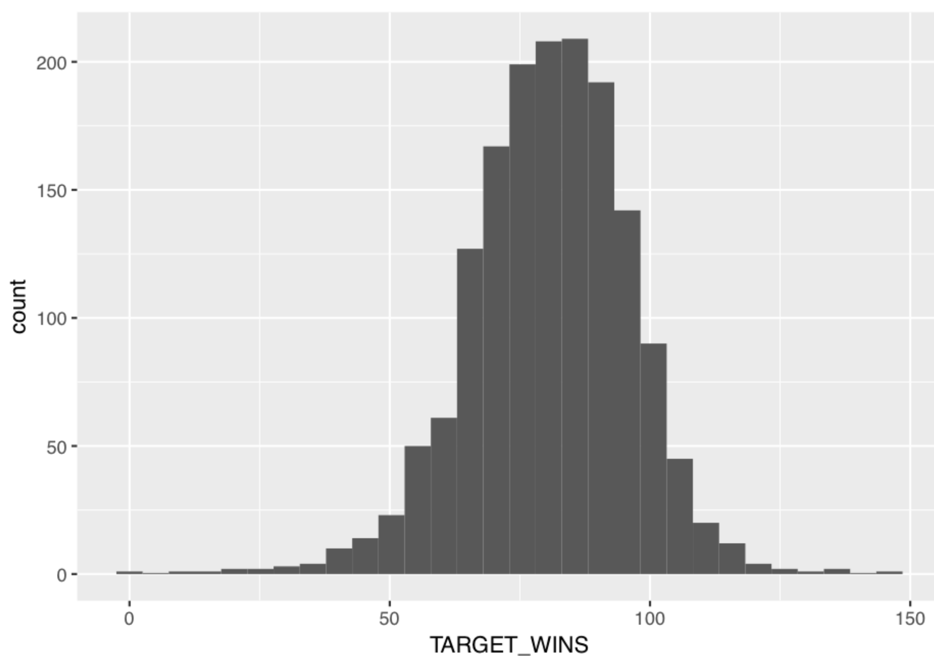
```
##      INDEX       TARGET_WINS       BATTING_H      BATTING_2B
##  Min.   :   1   Min.   :  0.00   Min.   : 891   Min.   : 69
##  1st Qu.: 625   1st Qu.: 71.00   1st Qu.:1381   1st Qu.:209
##  Median :1273   Median : 82.00   Median :1457   Median :239
##  Mean   :1274   Mean   : 80.89   Mean   :1470   Mean   :242
##  3rd Qu.:1958   3rd Qu.: 92.00   3rd Qu.:1539   3rd Qu.:274
##  Max.   :2535   Max.   :146.00   Max.   :2554   Max.   :403
##
##    BATTING_3B       BATTING_HR      BATTING_BB      BATTING_SO
##  Min.   :  0.00   Min.   :  0.0   Min.   :  0.0   Min.   :   0.0
##  1st Qu.: 34.00   1st Qu.: 44.0   1st Qu.:453.0   1st Qu.: 555.0
##  Median : 46.00   Median :105.0   Median :517.0   Median : 767.5
##  Mean   : 54.22   Mean   :101.7   Mean   :503.5   Mean   : 742.0
##  3rd Qu.: 70.00   3rd Qu.:147.0   3rd Qu.:582.0   3rd Qu.: 940.0
##  Max.   :223.00   Max.   :264.0   Max.   :878.0   Max.   :1399.0
##                                                   NA's   :65
##    BASERUN_SB      BASERUN_CS      BATTING_HBP      PITCHING_H
##  Min.   :  0.0   Min.   :  0.00   Min.   :29.00   Min.   : 1137
##  1st Qu.: 66.0   1st Qu.: 38.00   1st Qu.:51.00   1st Qu.: 1418
##  Median :101.0   Median : 49.00   Median :59.00   Median : 1515
##  Mean   :123.5   Mean   : 52.41   Mean   :59.83   Mean   : 1793
##  3rd Qu.:153.0   3rd Qu.: 62.00   3rd Qu.:67.00   3rd Qu.: 1677
##  Max.   :697.0   Max.   :201.00   Max.   :95.00   Max.   :24057
##  NA's   :88      NA's   :500      NA's   :1455
```

```
##   PITCHING_HR   PITCHING_BB    PITCHING_SO        FIELDING_E
##  Min.   :  0   Min.   :   0   Min.   :    0.0   Min.   :  65.0
##  1st Qu.: 55   1st Qu.: 476   1st Qu.:  624.8   1st Qu.: 126.0
##  Median :109   Median : 539   Median :  821.5   Median : 156.0
##  Mean   :108   Mean   : 556   Mean   :  830.5   Mean   : 244.2
##  3rd Qu.:152   3rd Qu.: 610   3rd Qu.:  974.0   3rd Qu.: 244.0
##  Max.   :343   Max.   :3645   Max.   :19278.0   Max.   :1898.0
##                               NA's   :65
##   FIELDING_DP
##  Min.   : 52.0
##  1st Qu.:133.0
##  Median :149.0
##  Mean   :147.1
##  3rd Qu.:164.0
##  Max.   :225.0
##  NA's   :199
```

a. Mean / Standard Deviation / Median

```r
ggplot(train, aes(x = TARGET_WINS)) +
  geom_histogram()
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



b. Bar Chart or Box Plot of the data

```r
library(reshape)
```

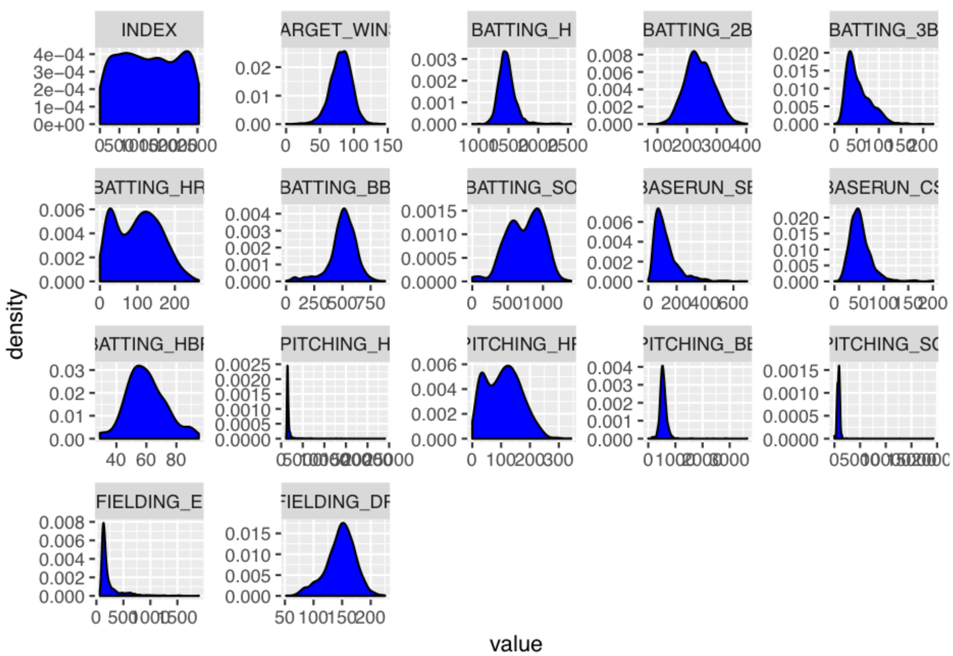
```
##
## Attaching package: 'reshape'

## The following object is masked from 'package:dplyr':
##
##     rename

## The following objects are masked from 'package:tidyr':
##
##     expand, smiths
```

```r
library(ggplot2)
par(mfrow = c(3, 3))
datasub = melt(train)
```

```
## Using  as id variables
```

```r
ggplot(datasub, aes(x= value)) +
    geom_density(fill='blue') +
    facet_wrap(~variable, scales = 'free')
```

```
## Warning: Removed 2372 rows containing non-finite values (stat_density).
```



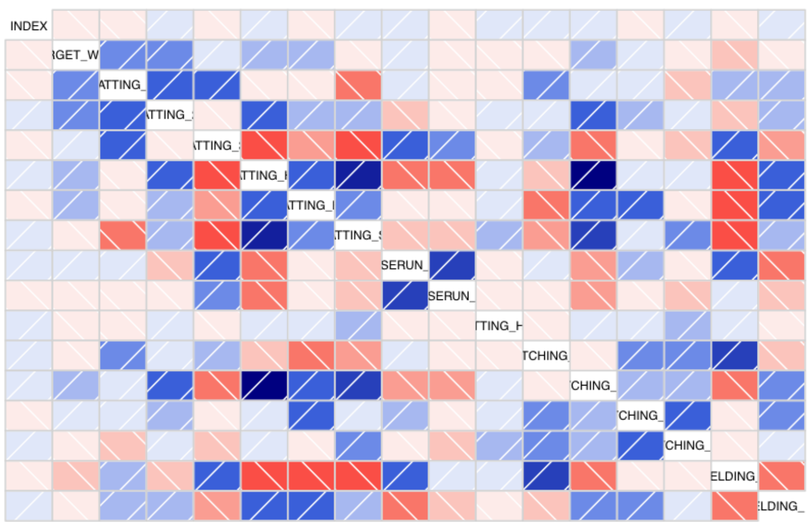
c. Is the data correlated to the target variable (or to other variables?)

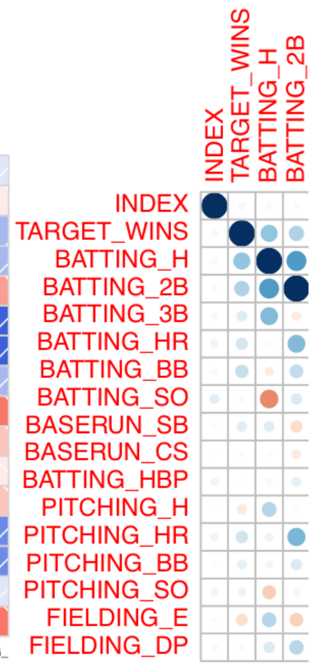
Findings: 1. TEAM_BATTING_H exhibits the highest correlation to the response variable, 2. TEAM_FIELDING_E exhibits the lowest correlation 3. Both TEAM_PITCHING_HR and TEAM_PITCHING_BB exhibit positive correlations to the response variable 4. The correlation plot shows that TARGET_WINS is positively correlated with BATTING_H, BATTING_2B, BATTING_HR, BATTING_BB, PITCHING_H, PITCHING_HR, PITCHING_BB and negatively correlated with FIELDING_E. Thus we are going to construct our linear model by selecting from these attributes.

```
library(corrplot)
```

```
## corrplot 0.84 loaded
```

```
library(corrgram)
corrplot(corrgram(train), method="circle")
```

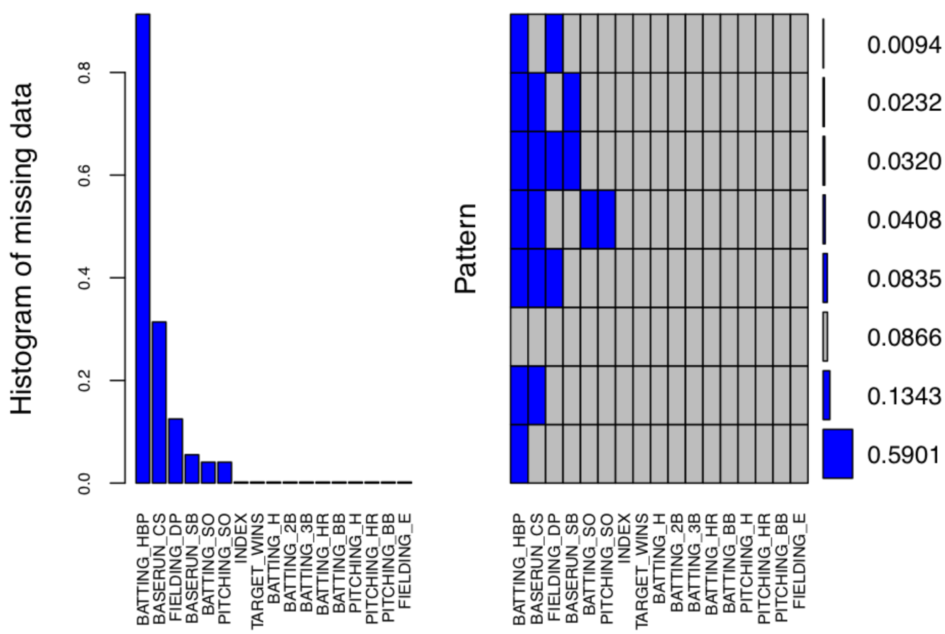




d. Are any of the variables missing and need to be imputed "fixed"?

```
## Warning: package 'VIM' was built under R version 3.5.2
## Loading required package: colorspace
## Loading required package: grid
## Loading required package: data.table
## Warning: package 'data.table' was built under R version 3.5.2
##
```

```
## Attaching package: 'data.table'

## The following object is masked from 'package:reshape':
##
##     melt

## The following objects are masked from 'package:dplyr':
##
##     between, first, last

## The following object is masked from 'package:purrr':
##
##     transpose

## VIM is ready to use.
##  Since version 4.0.0 the GUI is in its own package VIMGUI.
##
##           Please use the package to use the new (and old) GUI.

## Suggestions and bug-reports can be submitted at: https://github.com/alexkowa/VIM/issues

##
## Attaching package: 'VIM'

## The following object is masked from 'package:datasets':
##
##     sleep
```




```
##
##  Variables sorted by number of missings:
##     Variable      Count
##  BATTING_HBP 0.91337100
```

```
##    BASERUN_CS 0.31387320
##   FIELDING_DP 0.12492153
##    BASERUN_SB 0.05524168
##    BATTING_SO 0.04080352
##   PITCHING_SO 0.04080352
##         INDEX 0.00000000
##   TARGET_WINS 0.00000000
##     BATTING_H 0.00000000
##    BATTING_2B 0.00000000
##    BATTING_3B 0.00000000
##    BATTING_HR 0.00000000
##    BATTING_BB 0.00000000
##    PITCHING_H 0.00000000
##   PITCHING_HR 0.00000000
##   PITCHING_BB 0.00000000
##    FIELDING_E 0.00000000
```

2. DATA PREPARATION (25 Points)

Describe how you have transformed the data by changing the original variables or creating new variables. If you did transform the data or create new variables, discuss why you did this. Here are some possible transformations. a. Fix missing values (maybe with a Mean or Median value) b. Create flags to suggest if a variable was missing c. Transform data by putting it into buckets d. Mathematical transforms such as log or square root (or use Box-Cox) e. Combine variables (such as ratios or adding or multiplying) to create new variables

Missing imputation

Considering some columns has outliers, we'll fill in the missing values using their respective median values.

```
train_clean = train %>% mutate(
  PITCHING_SO = ifelse(is.na(train$PITCHING_SO), median(train$PITCHING_SO, na.rm = TRUE),train$PITCHING_
  BATTING_SO = ifelse(is.na(train$BATTING_SO), median(train$BATTING_SO, na.rm = TRUE), train$BATTING_SO)
  BASERUN_SB = ifelse(is.na(train$BASERUN_SB), median(train$BASERUN_SB, na.rm = TRUE), train$BASERUN_SB)
  BASERUN_CS = ifelse(is.na(train$BASERUN_CS), median(train$BASERUN_CS, na.rm = TRUE), train$BASERUN_CS)
  FIELDING_DP = ifelse(is.na(train$FIELDING_DP), median(train$FIELDING_DP, na.rm = TRUE), train$FIELDIN
```

Feature engineering

We'll add a new variable BATTING_HBP_YN that is 1 when the TEAM_BATTING_HBP exists and 0 when it does not.

```
train_clean = train_clean %>% mutate(BATTING_HBP_YN = ifelse(is.na(BATTING_HBP), 0, 1),
                                     BATTING_1B = BATTING_H - BATTING_2B - BATTING_3B - BATTING_HR)
```

Creat ratios: TARGET_WINS_Ratio = TARGET_WINS / 162 (i.e. the percentage of wins) TEAM_H_Ratio = (TEAM_BATTING_1B + TEAM_BATTING_2B + TEAM_BATTING_3B + TEAM_BATTING_HR) / TEAM_PITCHING_H (i.e. the ratio of hits earned to hits allowed) TEAM_BASERUN_Ratio = TEAM_BASERUN_SB / TEAM_BASERUN_CS (i.e. the ratio of successful steals to unsuccessful ones) TEAM_HR_SO_Ratio = TEAM_BATTING_HR / TEAM_BATTING_SO (i.e. the ratio of home runs to strikeouts)

```r
train_clean = train_clean %>%
  mutate(H_Ratio = (BATTING_1B + BATTING_2B + BATTING_3B + BATTING_HR) / PITCHING_H,
         BASERUN_Ratio = BASERUN_SB / BASERUN_CS,
         HR_SO_Ratio = BATTING_HR / ifelse(BATTING_SO == 0, median(BATTING_SO), BATTING_SO))
```

3. BUILD MODELS (25 Points) Using the training data set, build at least three different multiple linear regression models, using different variables (or the same variables with different transformations). Since we have not yet covered automated variable selection methods, you should select the variables manually (unless you previously learned Forward or Stepwise selection, etc.). Since you manually selected a variable for inclusion into the model or exclusion into the model, indicate why this was done. Discuss the coefficients in the models, do they make sense? For example, if a team hits a lot of Home Runs, it would be reasonably expected that such a team would win more games. However, if the coefficient is negative (suggesting that the team would lose more games), then that needs to be discussed. Are you keeping the model even though it is counter intuitive? Why? The boss needs to know.

Model 1: Simple linear regression using all features in training dataset

```r
library(caret)
```

```
## Warning: package 'caret' was built under R version 3.5.2

## Loading required package: lattice

##
## Attaching package: 'lattice'

## The following object is masked from 'package:corrgram':
##
##     panel.fill

##
## Attaching package: 'caret'

## The following object is masked from 'package:purrr':
##
##     lift
```

```r
train_model1 = train_clean
train_model1 = train_model1 %>% select(-INDEX,-BATTING_HBP)
model1 = train(TARGET_WINS ~ ., data = train_model1, method = 'lm', na.action=na.exclude)
```

```
## Warning in predict.lm(modelFit, newdata): prediction from a rank-deficient
## fit may be misleading

## Warning in predict.lm(modelFit, newdata): prediction from a rank-deficient
## fit may be misleading

## Warning in predict.lm(modelFit, newdata): prediction from a rank-deficient
## fit may be misleading

## Warning in predict.lm(modelFit, newdata): prediction from a rank-deficient
## fit may be misleading

## Warning in predict.lm(modelFit, newdata): prediction from a rank-deficient
## fit may be misleading

## Warning in predict.lm(modelFit, newdata): prediction from a rank-deficient
## fit may be misleading
```

```
## Warning in predict.lm(modelFit, newdata): prediction from a rank-deficient
## fit may be misleading

## Warning in predict.lm(modelFit, newdata): prediction from a rank-deficient
## fit may be misleading

## Warning in predict.lm(modelFit, newdata): prediction from a rank-deficient
## fit may be misleading

## Warning in predict.lm(modelFit, newdata): prediction from a rank-deficient
## fit may be misleading

## Warning in predict.lm(modelFit, newdata): prediction from a rank-deficient
## fit may be misleading

## Warning in predict.lm(modelFit, newdata): prediction from a rank-deficient
## fit may be misleading

## Warning in predict.lm(modelFit, newdata): prediction from a rank-deficient
## fit may be misleading

## Warning in predict.lm(modelFit, newdata): prediction from a rank-deficient
## fit may be misleading

## Warning in predict.lm(modelFit, newdata): prediction from a rank-deficient
## fit may be misleading

## Warning in predict.lm(modelFit, newdata): prediction from a rank-deficient
## fit may be misleading

## Warning in predict.lm(modelFit, newdata): prediction from a rank-deficient
## fit may be misleading

## Warning in predict.lm(modelFit, newdata): prediction from a rank-deficient
## fit may be misleading

## Warning in predict.lm(modelFit, newdata): prediction from a rank-deficient
## fit may be misleading

## Warning in predict.lm(modelFit, newdata): prediction from a rank-deficient
## fit may be misleading

## Warning in predict.lm(modelFit, newdata): prediction from a rank-deficient
## fit may be misleading

## Warning in predict.lm(modelFit, newdata): prediction from a rank-deficient
## fit may be misleading

## Warning in predict.lm(modelFit, newdata): prediction from a rank-deficient
## fit may be misleading

## Warning in predict.lm(modelFit, newdata): prediction from a rank-deficient
## fit may be misleading
```

```
## Warning in predict.lm(modelFit, newdata): prediction from a rank-deficient
## fit may be misleading
```

```r
summary(model1)
```

```
##
## Call:
## lm(formula = .outcome ~ ., data = dat)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -47.827  -8.555  -0.007   8.228  60.356
##
## Coefficients: (1 not defined because of singularities)
##                  Estimate Std. Error t value Pr(>|t|)
## (Intercept)     6.766e+01  9.423e+00   7.180 1.07e-12 ***
## BATTING_H       5.068e-02  4.628e-03  10.951  < 2e-16 ***
## BATTING_2B     -1.638e-02  1.140e-02  -1.437  0.15098
## BATTING_3B      8.561e-02  2.023e-02   4.232 2.45e-05 ***
## BATTING_HR      1.991e-01  3.665e-02   5.432 6.46e-08 ***
## BATTING_BB      1.251e-02  6.953e-03   1.800  0.07207 .
## BATTING_SO     -6.546e-04  3.813e-03  -0.172  0.86371
## BASERUN_SB      6.712e-02  2.813e-02   2.386  0.01716 *
## BASERUN_CS     -8.736e-02  4.789e-02  -1.824  0.06832 .
## PITCHING_H     -1.790e-03  5.719e-04  -3.130  0.00178 **
## PITCHING_HR    -1.454e-01  3.685e-02  -3.946 8.31e-05 ***
## PITCHING_BB     3.047e-04  4.720e-03   0.065  0.94854
## PITCHING_SO     1.396e-03  1.010e-03   1.383  0.16689
## FIELDING_E     -3.795e-02  4.103e-03  -9.249  < 2e-16 ***
## FIELDING_DP    -1.169e-01  1.531e-02  -7.636 3.86e-14 ***
## BATTING_HBP_YN -3.695e+00  1.423e+00  -2.597  0.00949 **
## BATTING_1B             NA         NA      NA       NA
## H_Ratio        -4.918e+01  7.469e+00  -6.584 6.22e-11 ***
## BASERUN_Ratio  -1.862e+00  1.410e+00  -1.321  0.18684
## HR_SO_Ratio     1.654e+01  1.435e+01   1.153  0.24911
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 12.79 on 1573 degrees of freedom
## Multiple R-squared:  0.3375, Adjusted R-squared:  0.3299
## F-statistic: 44.51 on 18 and 1573 DF,  p-value: < 2.2e-16
```

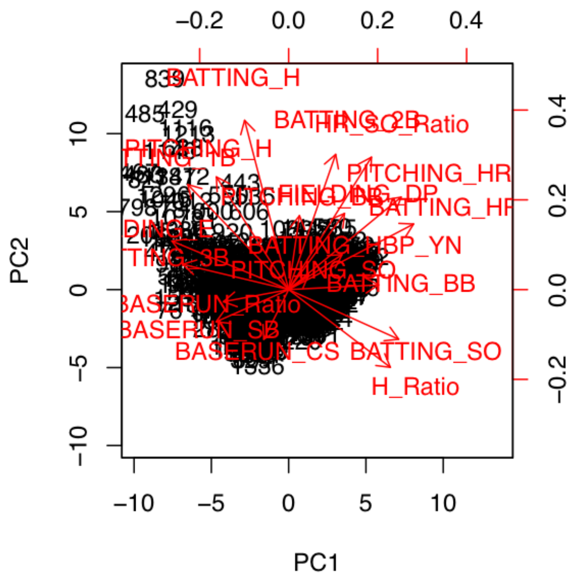
Model2 Principal Component Analysis

Given there is strong multicolinearity among variable, it is better to conduct principal component analysis on dataset in order to eliminate the colinearity.

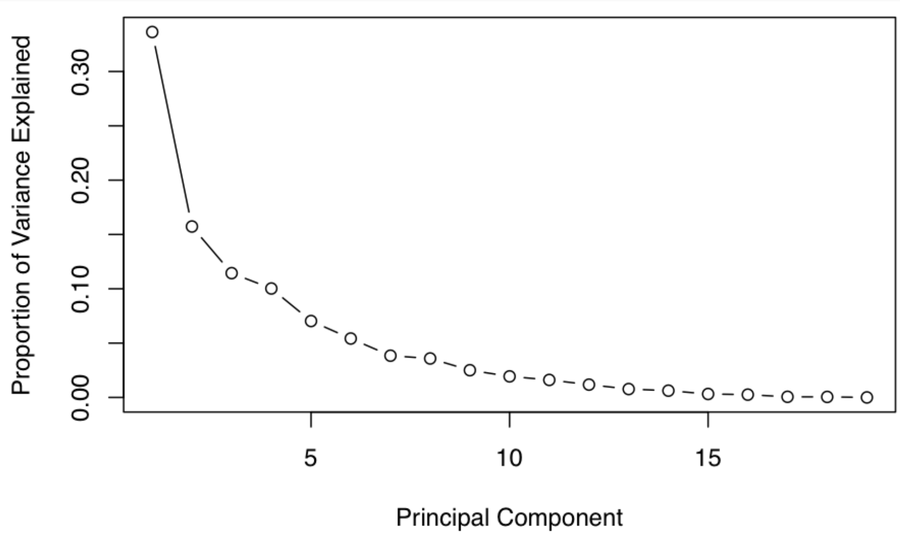
```
## Warning: package 'factoextra' was built under R version 3.5.2
```

```
## Welcome! Want to learn more? See two factoextra-related books at https://goo.gl/ve3WBa
```

```r
biplot(prin_comp, scale = 0)
```

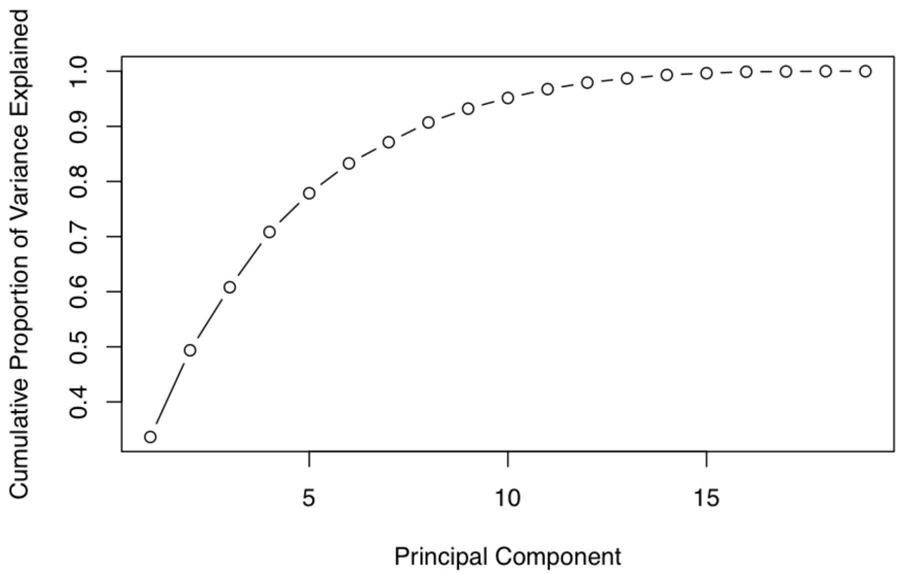



```r
std_dev <- prin_comp$sdev
pr_var <- std_dev^2
prop_varex <- pr_var/sum(pr_var)
 plot(prop_varex, xlab = "Principal Component",
             ylab = "Proportion of Variance Explained",
             type = "b")
```




```r
plot(cumsum(prop_varex), xlab = "Principal Component",
              ylab = "Cumulative Proportion of Variance Explained",
```

```r
               type = "b")
```



This plot shows that 15 components results in variance close to ~ 98%. Therefore, in this case, we'll select number of components as 15 [PC1 to PC15] and proceed to the modeling stage. This completes the steps to implement PCA on train data. For modeling, we'll use these 15 components as predictor variables and follow the normal procedures.

```r
model2_pca.data <- data.frame(TARGET_WINS = train_model2$TARGET_WINS, prin_comp$x)
model2_pca.data = model2_pca.data[1:16]
model2 = train(TARGET_WINS ~ ., data = model2_pca.data , method = 'lm', na.action=na.exclude)
summary(model2)
```

```
##
## Call:
## lm(formula = .outcome ~ ., data = dat)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -47.746  -8.883   0.064   8.324  56.687
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 80.941583   0.323745 250.016  < 2e-16 ***
## PC1          0.405345   0.128103   3.164 0.001585 **
## PC2          2.645992   0.187317  14.126  < 2e-16 ***
## PC3          2.605222   0.219663  11.860  < 2e-16 ***
## PC4         -2.823789   0.234651 -12.034  < 2e-16 ***
## PC5         -0.008195   0.280211  -0.029 0.976672
## PC6         -1.156916   0.319129  -3.625 0.000298 ***
## PC7         -1.895521   0.379162  -4.999 6.40e-07 ***
## PC8          4.093845   0.392434  10.432  < 2e-16 ***
```

```
## PC9          1.975951   0.469429   4.209 2.71e-05 ***
## PC10        -0.920239   0.533869  -1.724 0.084955 .
## PC11         2.310743   0.585530   3.946 8.28e-05 ***
## PC12        -2.248123   0.684972  -3.282 0.001053 **
## PC13         2.311541   0.847198   2.728 0.006434 **
## PC14        -5.342327   0.941939  -5.672 1.68e-08 ***
## PC15         6.312363   1.321921   4.775 1.96e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 12.92 on 1576 degrees of freedom
## Multiple R-squared:  0.3229, Adjusted R-squared:  0.3164
## F-statistic:  50.1 on 15 and 1576 DF,  p-value: < 2.2e-16
```

```
model3 <- lm(TARGET_WINS ~ BATTING_H+BATTING_2B+BATTING_3B+BATTING_HR+BATTING_BB+BATTING_HBP-BATTING_SO
summary(model3)
```

```
##
## Call:
## lm(formula = TARGET_WINS ~ BATTING_H + BATTING_2B + BATTING_3B +
##     BATTING_HR + BATTING_BB + BATTING_HBP - BATTING_SO + BASERUN_SB -
##     BASERUN_CS - FIELDING_E + FIELDING_DP - PITCHING_BB - PITCHING_H -
##     PITCHING_HR + PITCHING_SO, data = train_clean)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -21.1641  -6.2489   0.1316   5.1160  23.5190
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 56.30420   21.96474   2.563   0.0115 *
## BATTING_H    0.02089    0.01542   1.354   0.1780
## BATTING_2B   0.02833    0.03510   0.807   0.4211
## BATTING_3B  -0.12551    0.09357  -1.341   0.1822
## BATTING_HR   0.08063    0.03012   2.677   0.0084 **
## BATTING_BB   0.05762    0.01143   5.042 1.54e-06 ***
## BATTING_HBP  0.03145    0.05952   0.528   0.5982
## BASERUN_SB   0.01941    0.02747   0.707   0.4811
## FIELDING_DP -0.09366    0.04747  -1.973   0.0507 .
## PITCHING_SO -0.04326    0.00834  -5.187 8.12e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 8.681 on 128 degrees of freedom
##   (1455 observations deleted due to missingness)
## Multiple R-squared:  0.4951, Adjusted R-squared:  0.4596
## F-statistic: 13.95 on 9 and 128 DF,  p-value: 1.858e-15
```

Compare the RMSE(Root Mean Squared Error) among the 3 models - model 3 appears to have the lowest RMSE

```
fit1 <- fitted.values(model1)
error1 <- fit1 - test$TARGET_WINS
```

```
## Warning in fit1 - test$TARGET_WINS: longer object length is not a multiple
## of shorter object length
```

```r
rmse1 <- sqrt(mean(error1^2))
rmse1
```

```
## [1] 18.04854
```

```r
fit2 <- fitted.values(model2)
error2 <- fit2 - test$TARGET_WINS
```

```
## Warning in fit2 - test$TARGET_WINS: longer object length is not a multiple
## of shorter object length
```

```r
rmse2 <- sqrt(mean(error2^2))
rmse2
```

```
## [1] 17.93738
```

```r
fit3 <- fitted.values(model3)
error3 <- fit3 - test$TARGET_WINS
```

```
## Warning in fit3 - test$TARGET_WINS: longer object length is not a multiple
## of shorter object length
```

```r
rmse3 <- sqrt(mean(error3^2))
rmse3
```

```
## [1] 17.78142
```

Model selection rationale

As discussed above, we selected Model2, which was based on principal component analysis, followed by removal of any highly collinear variables. Although Model2 did not have the lowest RMSE, it was the most stable (little collinearity between variables).

Inference and regression diagnostics

For our inferences to be valid, we need to perform some regression diagnostics and validate some assumptions:

Independence of errors: Based on the residual plot below, the residuals appear random over the index values Outliers and leverage: Based on the leverage plots below, there do not appear to be any data points exerting undue leverage on the regression Normality: Based on the qq-plot below, the residuals are fairly normally distributed, although there are some outliers in the tails Constant variance: Based on the spread-level plot below, variance appears relatively constant, although again with a few outliers