OBP\_1.R

sports2i

2020-10-28

##################################################################  
####### ###########  
####### OBP ~ Barrel% + Whiff% + First Strike% ###########  
####### ###########  
##################################################################  
  
# Setting up the working directory  
setwd("C:/Users/sports2i/Desktop/WD")  
  
# Loading Library  
library(tidyverse)

## -- Attaching packages --------------------------------------- tidyverse 1.3.0 --

## √ ggplot2 3.3.2 √ purrr 0.3.4  
## √ tibble 3.0.4 √ dplyr 1.0.2  
## √ tidyr 1.1.2 √ stringr 1.4.0  
## √ readr 1.4.0 √ forcats 0.5.0

## -- Conflicts ------------------------------------------ tidyverse\_conflicts() --  
## x dplyr::filter() masks stats::filter()  
## x dplyr::lag() masks stats::lag()

library(car)

## Loading required package: carData

##   
## Attaching package: 'car'

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

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

library(GGally)

## Registered S3 method overwritten by 'GGally':  
## method from   
## +.gg ggplot2

library(MASS)

##   
## Attaching package: 'MASS'

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

library(lmtest)

## Loading required package: zoo

##   
## Attaching package: 'zoo'

## The following objects are masked from 'package:base':  
##   
## as.Date, as.Date.numeric

library(multcomp)

## Loading required package: mvtnorm

## Loading required package: survival

## Loading required package: TH.data

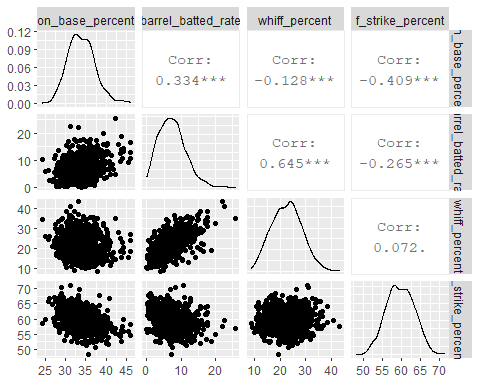
##   
## Attaching package: 'TH.data'

## The following object is masked from 'package:MASS':  
##   
## geyser

# Reading in data  
mlb\_orig <- read.csv(file = "MLB\_OBP3.csv", header = TRUE)  
mlb\_orig$year <- as.character(mlb\_orig$year)  
  
# turn OBP into %  
mlb\_orig$on\_base\_percent <- mlb\_orig$on\_base\_percent \* 100  
  
# Order by OBP  
mlb\_orig <- arrange(mlb\_orig, desc(on\_base\_percent))  
  
# Data check  
mlb <- mlb\_orig[,-c(1,2,3,6,9)]  
  
head(mlb)

## on\_base\_percent barrel\_batted\_rate whiff\_percent f\_strike\_percent  
## 1 46.0 12.9 27.0 54.7  
## 2 45.9 10.7 24.0 54.5  
## 3 45.9 16.5 18.6 58.1  
## 4 45.4 9.1 15.4 55.9  
## 5 44.2 13.3 18.2 59.8  
## 6 44.1 14.6 20.7 55.8

# Matrix scatter plots  
ggpairs(mlb)



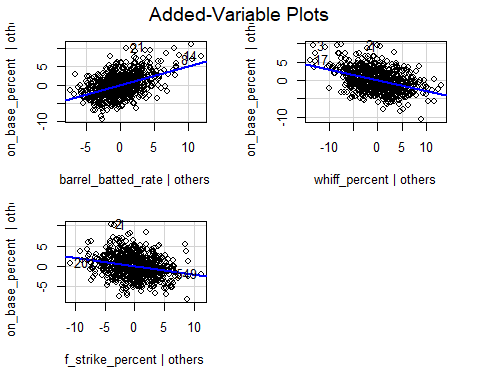
# Fitting model  
mlb\_mlr <- lm(on\_base\_percent ~ ., data = mlb)  
summary(mlb\_mlr)

##   
## Call:  
## lm(formula = on\_base\_percent ~ ., data = mlb)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -7.9984 -1.7669 -0.0807 1.7093 9.5521   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 48.77953 1.86192 26.198 < 2e-16 \*\*\*  
## barrel\_batted\_rate 0.51951 0.03768 13.788 < 2e-16 \*\*\*  
## whiff\_percent -0.28952 0.02438 -11.874 < 2e-16 \*\*\*  
## f\_strike\_percent -0.20261 0.03154 -6.424 2.45e-10 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 2.725 on 704 degrees of freedom  
## Multiple R-squared: 0.3519, Adjusted R-squared: 0.3492   
## F-statistic: 127.4 on 3 and 704 DF, p-value: < 2.2e-16

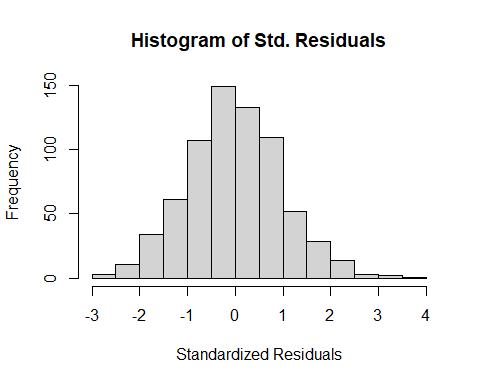
# Confidence Interval  
confint(mlb\_mlr)

## 2.5 % 97.5 %  
## (Intercept) 45.1239446 52.4351160  
## barrel\_batted\_rate 0.4455322 0.5934827  
## whiff\_percent -0.3373876 -0.2416443  
## f\_strike\_percent -0.2645293 -0.1406816

# Test on reduced model  
#reduced.lm <- lm(on\_base\_percent ~ barrel\_batted\_rate   
# + whiff\_percent, data = mlb)  
#anova(mlb\_mlr, reduced.lm)  
  
# Added variable plots  
avPlots(mlb\_mlr)



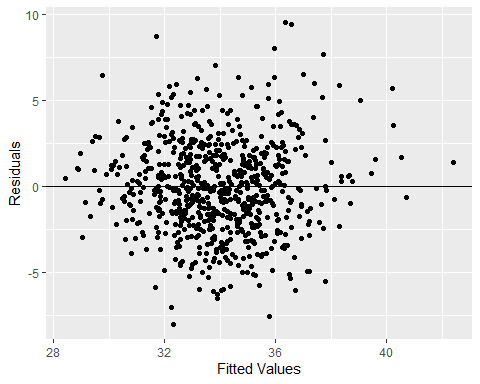
# Std.Residual histogram (normality)  
std <- stdres(mlb\_mlr)  
hist(std, main = "Histogram of Std. Residuals", xlab = "Standardized Residuals")



ks.test(std,"pnorm")

##   
## One-sample Kolmogorov-Smirnov test  
##   
## data: std  
## D = 0.019111, p-value = 0.9582  
## alternative hypothesis: two-sided

# Residual plots  
qplot(x = mlb\_mlr$fitted.values, y = mlb\_mlr$residuals,   
 data = mlb\_mlr, geom = "point", xlab = "Fitted Values",   
 ylab = "Residuals") + geom\_hline(aes(yintercept=0))



bptest(mlb\_mlr)

##   
## studentized Breusch-Pagan test  
##   
## data: mlb\_mlr  
## BP = 7.016, df = 3, p-value = 0.07139

# Try prediction - Joey Votto (2015)  
predict.lm (mlb\_mlr,newdata=data.frame(barrel\_batted\_rate = 10.7, f\_strike\_percent = 54.5, whiff\_percent = 24), interval="prediction", level = 0.95)

## fit lwr upr  
## 1 36.34788 30.98583 41.70993

# Real Stats: 0.459  
  
# Try prediction - Shin-Soo Choo(2018)  
predict.lm (mlb\_mlr,newdata=data.frame(barrel\_batted\_rate = 11.1, f\_strike\_percent = 58.2, whiff\_percent = 27), interval="prediction", level = 0.95)

## fit lwr upr  
## 1 34.9375 29.58054 40.29445

# Real Stats: 0.376  
  
# Total prediction  
preds <- predict.lm(mlb\_mlr, newdata = mlb, interval = "prediction", level = .95)  
  
diff <- mlb$on\_base\_percent - preds[,1]  
diff

## 1 2 3 4 5   
## 9.4182717721 9.5521190936 5.7049691152 7.6771410899 5.8960157370   
## 6 7 8 9 10   
## 5.0340243635 8.0513217017 1.3972705140 3.5693371159 6.5036432618   
## 11 12 13 14 15   
## 5.9879796594 5.1996404657 6.3507317018 1.6812242225 5.9477489135   
## 16 17 18 19 20   
## 4.0440101477 1.5912087565 4.9654251282 6.3264760846 5.1518305350   
## 21 22 23 24 25   
## 7.0579409211 5.7806056705 4.3052025676 2.7111912799 8.7089379569   
## 26 27 28 29 30   
## 3.5524708450 4.2166535566 3.6116041844 3.8033072074 0.7576266317   
## 31 32 33 34 35   
## 3.3277238988 3.6353192447 5.3102790419 -0.6087553014 3.1235126078   
## 36 37 38 39 40   
## 4.6837589856 4.6625428674 2.8610591807 3.5224186899 3.9938661705   
## 41 42 43 44 45   
## 6.3043751613 3.4814751651 1.3881830709 0.6268280969 5.3243903572   
## 46 47 48 49 50   
## 5.6809061629 0.5943181673 2.2736175908 0.3310834397 0.6247517383   
## 51 52 53 54 55   
## 1.8254237490 2.1848582918 2.5723708095 4.4413957274 2.4258940663   
## 56 57 58 59 60   
## 0.3130141520 2.2913993322 3.8934359156 4.4055627366 4.2416908404   
## 61 62 63 64 65   
## 2.5341212765 2.8965212713 5.9649426721 2.5659513030 2.0541561500   
## 66 67 68 69 70   
## 5.4743593348 2.2873006071 2.3706586985 3.5008525410 4.6434609823   
## 71 72 73 74 75   
## 2.2812021549 5.8147218447 1.7962777134 1.4858572879 1.6007983367   
## 76 77 78 79 80   
## 1.4528873498 3.5970907934 3.0547669937 0.0141249332 4.4407866680   
## 81 82 83 84 85   
## -0.9811107228 5.3724497540 1.5802941583 3.2734008042 2.0089961787   
## 86 87 88 89 90   
## -0.2337430930 4.7287143563 2.6625041251 1.9429035709 2.7049104087   
## 91 92 93 94 95   
## 2.2084635193 2.6247923348 4.4371191261 3.8043925868 5.1618181823   
## 96 97 98 99 100   
## 1.1683155880 1.9256420679 1.3062533568 -0.0838671460 -0.7147654985   
## 101 102 103 104 105   
## 2.0865157114 1.7169310114 4.7232848714 3.6303643654 1.1780758720   
## 106 107 108 109 110   
## 1.2119731093 0.8462765427 2.6303255532 3.0842388256 2.3394088971   
## 111 112 113 114 115   
## 4.1862365128 4.2837666223 5.3413110356 2.5091526284 0.7557422618   
## 116 117 118 119 120   
## 3.0740451647 1.7067479760 4.7915732685 1.6764797593 1.8277220075   
## 121 122 123 124 125   
## 4.8757294485 3.6902813414 1.3669148405 1.5166203500 3.6097023890   
## 126 127 128 129 130   
## 2.7580626231 4.1086990117 -0.2092222932 3.6457845638 3.4840324459   
## 131 132 133 134 135   
## 2.4352932947 -0.1160243958 3.2794587844 1.0414124508 -1.0549512438   
## 136 137 138 139 140   
## 1.8016426622 1.3061411828 1.0387271956 2.2970913416 1.0271614652   
## 141 142 143 144 145   
## 2.9344710580 1.6546399822 1.4422531900 2.4726898884 0.8110188671   
## 146 147 148 149 150   
## 0.6615655100 1.4507473201 0.0888382043 1.5446072362 0.2689223813   
## 151 152 153 154 155   
## 2.8216644826 0.2874236381 1.3445050630 1.0385569067 1.2485674659   
## 156 157 158 159 160   
## 2.4557331051 1.3405395238 0.1387110344 4.3810964373 1.4496366929   
## 161 162 163 164 165   
## 0.1727755925 0.6785048677 0.7459404936 0.5295465240 0.2161926780   
## 166 167 168 169 170   
## 0.4136010856 1.0873092650 6.4522694159 1.0946559867 2.1289986964   
## 171 172 173 174 175   
## 1.2162154830 4.2133182078 2.4713049220 1.2900486545 4.5798224698   
## 176 177 178 179 180   
## -2.3125645537 0.6597503675 2.4137245666 -1.2361551841 3.9001207468   
## 181 182 183 184 185   
## 2.1305797249 1.8724774972 1.8647350562 -0.2539281137 0.0009212837   
## 186 187 188 189 190   
## 3.2196355358 2.4517746633 3.8914742122 2.5883211491 -1.0268018823   
## 191 192 193 194 195   
## 1.7067219561 1.5169425652 -0.0609306596 2.6966231136 -0.7775938063   
## 196 197 198 199 200   
## 2.7659842488 0.5774892322 -1.1275805196 -0.6440405016 1.7466436657   
## 201 202 203 204 205   
## -1.4377925408 0.8130528713 1.8454471105 0.9641013988 2.6821079436   
## 206 207 208 209 210   
## 3.9256593812 0.5655677580 0.3746465785 2.7342003891 1.5724478910   
## 211 212 213 214 215   
## 2.6394569270 0.9191846000 1.9952833858 2.5612606081 -0.9426603805   
## 216 217 218 219 220   
## 1.8313802389 2.3565604833 2.2344675416 0.7805450516 3.4497512640   
## 221 222 223 224 225   
## 0.0701852450 0.3323704232 0.7611119720 -2.0939996398 0.1917509209   
## 226 227 228 229 230   
## 0.6242794370 0.6948429849 0.8900112295 -0.1914507907 3.1343499740   
## 231 232 233 234 235   
## 2.7755030059 3.7414537442 3.0411664447 1.3213427315 2.4253318565   
## 236 237 238 239 240   
## -0.2021351625 1.6587561186 -2.4131610382 -2.0032105498 1.4645257871   
## 241 242 243 244 245   
## 2.2800019853 -0.5099637576 2.6375956177 2.6772206526 3.6887678717   
## 246 247 248 249 250   
## 3.6623856786 0.6958281611 1.7305948790 0.2441138672 0.5785685300   
## 251 252 253 254 255   
## 2.5708588809 1.2811337871 -1.3733851733 2.8008697844 -1.7426549870   
## 256 257 258 259 260   
## 1.5846229887 0.1044021236 -0.7122604750 -0.5716660437 2.2501370017   
## 261 262 263 264 265   
## 1.1320054647 1.7567040620 0.2319911754 -0.8005949074 -2.1811341687   
## 266 267 268 269 270   
## 1.1571588449 1.7557671560 -0.0802296250 1.4783260543 0.1186861478   
## 271 272 273 274 275   
## -0.9740075097 2.8941298612 1.0896183105 1.0672881326 0.5197256910   
## 276 277 278 279 280   
## 2.6452929040 -0.5135581845 1.8634868619 0.9202153109 -0.8409457860   
## 281 282 283 284 285   
## 1.0568935643 2.9949172469 -0.8846788425 -1.1106988913 0.0085917758   
## 286 287 288 289 290   
## 2.5722468854 -1.7949719397 1.9269032783 2.2631719033 -0.6060290049   
## 291 292 293 294 295   
## 0.2813575135 1.7057576236 0.1121895510 -0.9980861207 1.5700573598   
## 296 297 298 299 300   
## 1.8870140591 0.5728766605 -0.9226027861 -1.6187615164 -0.1929319239   
## 301 302 303 304 305   
## 1.2597641785 -0.6121674444 -0.6841855575 -0.8628087571 -2.8228565688   
## 306 307 308 309 310   
## -0.3122838369 -0.7186784144 1.9938270134 1.0841711739 0.1732855080   
## 311 312 313 314 315   
## 3.4690995819 -0.8162338277 1.0849823909 1.4227053484 -0.1432653947   
## 316 317 318 319 320   
## 1.6190383545 1.0647962932 0.2033724949 1.4329512665 -2.3542298653   
## 321 322 323 324 325   
## -1.8880695359 -1.0489071028 3.7845843797 -0.0812613571 1.1143606493   
## 326 327 328 329 330   
## -1.9667488310 0.9582287688 -0.5382406993 -1.3729461015 2.5707570104   
## 331 332 333 334 335   
## -0.3142425580 -1.1246044679 -1.8085705975 -0.4022666393 -0.9574330264   
## 336 337 338 339 340   
## -1.5135768550 -3.0121276693 1.2206719570 2.4718764687 -0.6647613878   
## 341 342 343 344 345   
## -0.7630991928 -0.4198577121 0.0089110175 -0.2367029483 2.3033833377   
## 346 347 348 349 350   
## -0.8590765510 2.5197356690 2.0429304962 0.5915235153 0.8654468280   
## 351 352 353 354 355   
## -1.2444444403 1.3171843159 -1.2431436087 0.5568017000 0.2689329022   
## 356 357 358 359 360   
## -1.0852505661 0.5579483392 -0.4129059406 1.1164115309 -1.1475983036   
## 361 362 363 364 365   
## 0.6426400746 -0.3602688877 -0.7535195905 -1.2399943580 0.0590832632   
## 366 367 368 369 370   
## -2.7631074564 0.3359567898 2.8717364828 -1.5650134561 -0.0998269819   
## 371 372 373 374 375   
## -1.3225303032 -1.7899722455 2.1375506170 -2.5550406181 0.1602616120   
## 376 377 378 379 380   
## -1.5915613255 -2.6570699450 -1.8947132277 -0.2460118117 -2.6659266783   
## 381 382 383 384 385   
## -0.8316772394 1.2800840802 -1.5440369993 -0.0285486152 -0.6258048561   
## 386 387 388 389 390   
## -0.7574542010 -0.2625645863 -0.1119400670 2.7656860694 0.7304682477   
## 391 392 393 394 395   
## -0.0802029171 0.6792710700 1.3207065560 -2.7458393096 -0.9319352441   
## 396 397 398 399 400   
## -2.2172096299 0.2290618105 0.3314222572 -0.8334869884 0.4990857387   
## 401 402 403 404 405   
## 1.3010666273 0.0474285473 -0.1147918568 -1.7547520737 1.8403623356   
## 406 407 408 409 410   
## -2.3294355912 -2.1325563267 -2.2690739871 -3.4141384096 0.6222871203   
## 411 412 413 414 415   
## 0.3065306413 -1.6252724230 -3.4165544037 -2.3483902180 1.6978730082   
## 416 417 418 419 420   
## -0.6257998569 -0.0345113076 -1.5598344145 -3.1562790377 -2.6392324300   
## 421 422 423 424 425   
## -1.4474681700 -1.8769885888 -2.3617304715 1.4575320583 -2.8304100644   
## 426 427 428 429 430   
## -1.3359064148 1.5587752392 -0.1162229901 0.8197723080 -3.5377473163   
## 431 432 433 434 435   
## -0.4671716805 -1.8968686378 0.0533102271 -3.6175059932 -0.5206610404   
## 436 437 438 439 440   
## 0.5824621903 -0.0751535449 0.6510262438 1.0709929175 -0.2647757700   
## 441 442 443 444 445   
## -2.0897348672 -4.0844566765 -0.9828366119 -2.8778757211 -0.7660373112   
## 446 447 448 449 450   
## 0.5625282449 -0.4979079572 -1.7627544805 -2.1292291363 1.0747839546   
## 451 452 453 454 455   
## -1.8638018308 2.8656456814 -1.2423766115 -0.8836406423 -1.2125161577   
## 456 457 458 459 460   
## -1.5474687129 -0.7797229092 -0.6702614925 -0.7134365806 -0.8610975250   
## 461 462 463 464 465   
## -1.6873920670 0.6292374789 -1.4617341559 -0.6251272192 2.9010431026   
## 466 467 468 469 470   
## -2.3308785185 -2.4655479927 -5.5121168875 0.5211903182 0.4122561675   
## 471 472 473 474 475   
## -4.9046912823 -2.7014668273 -3.5620762026 -0.6690969636 1.8180573827   
## 476 477 478 479 480   
## -1.2724697727 -0.0760099165 -0.1982440116 -2.4591466926 -1.5565347205   
## 481 482 483 484 485   
## -4.9304429401 -2.6036397276 -1.5258358689 -0.0745366808 0.8960945047   
## 486 487 488 489 490   
## -0.5746526791 -0.2270373482 -1.2432786017 -0.4212676797 -1.1568542143   
## 491 492 493 494 495   
## -0.7262690971 -0.0041556917 -2.8125092672 -0.2259006311 -0.0613842904   
## 496 497 498 499 500   
## -0.9169954596 -1.1797674316 -3.1696986647 2.6124882274 -2.6346799432   
## 501 502 503 504 505   
## -0.7758526666 -1.7417057670 -1.6037936174 -3.1244840949 -3.1903778987   
## 506 507 508 509 510   
## -2.3485584875 -1.7454922754 -1.6270588241 -1.0225391431 -3.3256168250   
## 511 512 513 514 515   
## 0.4789914912 -3.1881881998 -3.8811557375 1.1937824769 -1.9605136207   
## 516 517 518 519 520   
## 0.7101848081 0.0434879164 1.4861936854 -1.1492383207 -0.7996126488   
## 521 522 523 524 525   
## -3.2077066591 -1.1117360375 -1.5201044010 -2.6539102382 -3.9153233270   
## 526 527 528 529 530   
## -1.6843235188 -2.4931243997 -0.1101175140 -0.2596562509 -1.9890404369   
## 531 532 533 534 535   
## -0.2531026356 -1.7977080620 -1.0449376694 -0.8787986621 -0.2470655832   
## 536 537 538 539 540   
## 1.1442313978 -3.5165548819 -0.2408441840 -5.1085707514 1.2518932203   
## 541 542 543 544 545   
## -2.0468548223 -3.9439931737 -2.2614478687 -4.0800897167 -1.3531312930   
## 546 547 548 549 550   
## -0.9634029143 -2.4271217843 -1.2452768460 0.2983858285 -0.5412414701   
## 551 552 553 554 555   
## -0.7173278437 -3.0773857263 -1.4064645833 1.2387234936 0.2170608066   
## 556 557 558 559 560   
## -2.4652756934 -0.0268883953 -1.5324526072 -1.3166318442 -1.7859774309   
## 561 562 563 564 565   
## -0.4356762319 0.3623935621 -5.3230092782 -1.4385676200 -3.1694572942   
## 566 567 568 569 570   
## -2.4970611975 1.0056627770 -1.6265103085 -0.8494874349 -1.5658690557   
## 571 572 573 574 575   
## -4.0624680253 -1.4221181096 -2.2178076837 -4.8794569875 0.7220044893   
## 576 577 578 579 580   
## -1.5303080681 -0.9256493436 1.9216966334 -1.9405203155 -3.3233205084   
## 581 582 583 584 585   
## -1.8288174069 -2.8990816986 -2.4733658988 -3.2290424483 -0.3174232244   
## 586 587 588 589 590   
## 0.1409935677 -1.7793403077 -2.5791961022 -2.2408071508 -1.4289002127   
## 591 592 593 594 595   
## -6.0147235642 -3.0154933198 -2.4307898692 -1.5825600995 -4.1500785752   
## 596 597 598 599 600   
## -3.0751008095 -0.0986759896 -1.9937340912 0.7188031354 -3.4679184520   
## 601 602 603 604 605   
## -3.6418487134 -2.2747521942 -1.2758234612 -3.9261508974 -4.6167974276   
## 606 607 608 609 610   
## -1.3161470696 -1.1966251779 -3.5042565218 -1.3561785385 -4.1369798550   
## 611 612 613 614 615   
## -2.8296304171 -3.2921024306 0.9385459340 -0.3287293897 -0.4789934157   
## 616 617 618 619 620   
## -1.5285162875 -3.7654600454 -0.8333238250 -2.2616779970 -3.3351041076   
## 621 622 623 624 625   
## -2.1800896608 -3.6659003763 -2.9481714350 -2.4985227902 -5.1772321467   
## 626 627 628 629 630   
## -1.7155330950 -4.1024467728 -4.5582941593 -2.8697656983 -5.1080967639   
## 631 632 633 634 635   
## -2.0547460762 -3.8501924002 -4.2738748860 -4.9736306172 -0.5601063565   
## 636 637 638 639 640   
## -1.5603733463 1.0015913803 -2.9223555772 -0.5995486064 1.0508029726   
## 641 642 643 644 645   
## -4.4778014450 -2.6818681644 -4.9977694477 -0.6807394236 -3.5357808938   
## 646 647 648 649 650   
## -1.0212401974 -1.0800386627 -0.8486785733 -5.7313250937 -4.1816990365   
## 651 652 653 654 655   
## -4.3753292133 -3.9012351475 -1.3877396981 -2.8387026830 -3.0627430107   
## 656 657 658 659 660   
## -4.3582785411 -3.8675853011 -0.2341579398 -2.3935498232 -2.7154809865   
## 661 662 663 664 665   
## -4.2785695474 -3.2555142485 -3.4451068673 -2.6656783872 -0.7431710581   
## 666 667 668 669 670   
## -2.0878296065 -1.2196854203 0.4647072901 -2.1130747671 -3.5308366630   
## 671 672 673 674 675   
## -4.7125724795 -1.9249178102 -2.9437111279 -1.9367314799 -0.9700970842   
## 676 677 678 679 680   
## -4.4036827024 -5.7762404160 -2.6790737174 -4.4927661725 -4.9619017933   
## 681 682 683 684 685   
## -2.8381512832 -4.3524460804 -5.2681085025 -0.9263862094 -4.2032016822   
## 686 687 688 689 690   
## -7.5642920636 -3.6300645102 -4.7574156990 -4.4546638185 -6.0009188599   
## 691 692 693 694 695   
## -2.1667056746 -6.0437436095 -4.5794505259 -3.0353604226 -3.6476140289   
## 696 697 698 699 700   
## -6.0613261667 -5.1972203193 -6.2913536257 -1.7281815063 -3.0837211209   
## 701 702 703 704 705   
## -6.5049065860 -5.9498941650 -4.8777297570 -3.8919435697 -2.9398404354   
## 706 707 708   
## -5.8654916151 -7.0330947103 -7.9984294494

# Top missed  
hlier <- mlb\_orig[which(diff>8),]  
hlier

## last\_name first\_name year on\_base\_percent barrel\_batted\_rate  
## 1 Harper Bryce 2015 46.0 12.9  
## 2 Votto Joey 2015 45.9 10.7  
## 7 Cabrera Miguel 2015 44.0 10.6  
## 25 Nimmo Brandon 2018 40.4 6.5  
## hard\_hit\_percent whiff\_percent f\_strike\_percent X  
## 1 47.5 27.0 54.7 NA  
## 2 42.0 24.0 54.5 NA  
## 7 53.0 23.1 57.5 NA  
## 25 42.1 26.6 63.0 NA

# Low missed  
llier <- mlb\_orig[which(diff<(-7.5)),]  
llier

## last\_name first\_name year on\_base\_percent barrel\_batted\_rate  
## 686 Odor Rougned 2019 28.2 13.6  
## 708 Davis Chris 2018 24.3 10.3  
## hard\_hit\_percent whiff\_percent f\_strike\_percent X  
## 686 45.6 29.4 57.1 NA  
## 708 40.1 34.4 58.6 NA

# Cross Validation  
n.cv <- 1000  
bias <- rep(NA,n.cv)  
rpmse <- rep(NA,n.cv)  
cvg <- rep(NA,n.cv)  
wid <- rep(NA,n.cv)  
n.test <- round(nrow(mlb)/10)  
  
for(i in 1:n.cv){  
 #split into test and training set  
 test.obs <- sample(1:nrow(mlb),n.test)  
 test.set <- mlb[test.obs,]  
 train.set <- mlb[-test.obs,]  
   
 # fit a lm using training data only  
 train.lm <- lm(on\_base\_percent ~ ., data=train.set)  
   
 # Prediction and prediction intervals  
 test.pred <- predict.lm(train.lm,newdata = test.set,interval="prediction",level = 0.95)   
   
 # calculate results  
 bias[i] <- mean(test.pred[,1] - test.set$on\_base\_percent)  
 rpmse[i] <- sqrt(mean((test.pred[,1] - test.set$on\_base\_percent)^2))  
 cvg[i] <- mean(test.pred[,2] < test.set$on\_base\_percent & test.pred[,3] > test.set$on\_base\_percent)  
 wid[i] <- mean(test.pred[,3]-test.pred[,2])   
}  
mean(bias)

## [1] -0.003081227

mean(rpmse)

## [1] 2.724675

mean(cvg)

## [1] 0.9473662

mean(wid)

## [1] 10.73377