Predicting Pitcher DL

Roger Chow

April 16, 2017

# Load libraries

setwd("F:/Capstone\_Workspace/predictDL/");  
library('RODBC');  
library('DBI');  
library('plyr');  
library('dplyr');

##   
## Attaching package: 'dplyr'

## The following objects are masked from 'package:plyr':  
##   
## arrange, count, desc, failwith, id, mutate, rename, summarise,  
## summarize

## The following objects are masked from 'package:stats':  
##   
## filter, lag

## The following objects are masked from 'package:base':  
##   
## intersect, setdiff, setequal, union

library('stringi');  
library('sqldf');

## Loading required package: gsubfn

## Loading required package: proto

## Loading required package: RSQLite

library('corrplot');  
library('reshape2');  
library('lattice');  
library('ggplot2');  
library('caret');  
library('logistf');  
library('klaR');

## Loading required package: MASS

##   
## Attaching package: 'MASS'

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

library('pROC');

## Type 'citation("pROC")' for a citation.

##   
## Attaching package: 'pROC'

## The following objects are masked from 'package:stats':  
##   
## cov, smooth, var

library('pls');

##   
## Attaching package: 'pls'

## The following object is masked from 'package:corrplot':  
##   
## corrplot

## The following object is masked from 'package:stats':  
##   
## loadings

library('ROSE');

## Loaded ROSE 0.0-3

library('randomForest')

## randomForest 4.6-12

## Type rfNews() to see new features/changes/bug fixes.

##   
## Attaching package: 'randomForest'

## The following object is masked from 'package:ggplot2':  
##   
## margin

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

resetPar <- function() {  
 dev.new()  
 op <- par(no.readonly = TRUE)  
 dev.off()  
 op  
}  
  
par(resetPar());

# Prepare data

## Load from Cache

Load the data from cache if the cache exists. This is will save time when re-running the analysis since the data will not change. It takes approximately 30 minutes to load the data from scratch.

need\_load <- TRUE;  
if (file.exists("pitches\_dl\_dataset.csv")){  
 pitches\_dl\_dataset <- read.csv("pitches\_dl\_dataset.csv");  
 pitches\_dl\_predict <- read.csv("pitches\_dl\_predict.csv");  
   
   
 drops <- c("X");  
 pitches\_dl\_dataset <- pitches\_dl\_dataset[ , !(names(pitches\_dl\_dataset) %in% drops)];  
 pitches\_dl\_predict <- pitches\_dl\_predict[ , !(names(pitches\_dl\_predict) %in% drops)];  
  
 need\_load <- FALSE;  
}

## Load pitchFX data from Database

If there is data cached, load the data from the SQL Server database.

Impute missing values using the Median.

The median is used to measure the central tendency for each continuous variable for each pitcher. The count of the pitch types is also calculate for each pitcher in each year.

if (need\_load) {  
 years <- c(2010, 2011, 2012, 2013, 2014, 2015, 2016);  
   
 dbhandle <- odbcDriverConnect('driver={SQL Server};server=localhost;database=PitchFx;trusted\_connection=true');  
   
 impute <- function(x, fun) {  
 missing <- is.na(x)  
 replace(x, missing, fun(x[!missing]))  
 }  
   
   
 centroid\_fun <- median;  
 impute\_fun <- median;  
   
   
 for (i in 2010:2016)  
 {  
 query <-  
 paste("SELECT ", i, " as season, m.rsID, p.id, p.atbatid, a.pitcher,   
 p.x, p.y, p.start\_speed, p.end\_speed, p.sz\_top, p.sz\_bot,  
 p.px, p.pz, p.x0, p.y0, p.z0, p.vx0, p.vy0, p.vz0, p.ax, p.ay, p.az,  
 p.break\_y, p.break\_length, p.spin\_dir, p.spin\_rate,   
 1 as type\_ALL,  
 CASE WHEN ISNULL(p.pitch\_type,'UN') = 'AB' THEN 1 ELSE 0 END AS type\_AB,  
 CASE WHEN ISNULL(p.pitch\_type,'UN') = 'AS' THEN 1 ELSE 0 END AS type\_AS,  
 CASE WHEN ISNULL(p.pitch\_type,'UN') = 'CH' THEN 1 ELSE 0 END AS type\_CH,  
 CASE WHEN ISNULL(p.pitch\_type,'UN') = 'CU' THEN 1 ELSE 0 END AS type\_CU,  
 CASE WHEN ISNULL(p.pitch\_type,'UN') = 'EP' THEN 1 ELSE 0 END AS type\_EP,  
 CASE WHEN ISNULL(p.pitch\_type,'UN') = 'FA' THEN 1 ELSE 0 END AS type\_FA,  
 CASE WHEN ISNULL(p.pitch\_type,'UN') = 'FC' THEN 1 ELSE 0 END AS type\_FC,  
 CASE WHEN ISNULL(p.pitch\_type,'UN') = 'FF' THEN 1 ELSE 0 END AS type\_FF,  
 CASE WHEN ISNULL(p.pitch\_type,'UN') = 'FO' THEN 1 ELSE 0 END AS type\_FO,  
 CASE WHEN ISNULL(p.pitch\_type,'UN') = 'FS' THEN 1 ELSE 0 END AS type\_FS,  
 CASE WHEN ISNULL(p.pitch\_type,'UN') = 'FT' THEN 1 ELSE 0 END AS type\_FT,  
 CASE WHEN ISNULL(p.pitch\_type,'UN') = 'IN' THEN 1 ELSE 0 END AS type\_IN,  
 CASE WHEN ISNULL(p.pitch\_type,'UN') = 'KC' THEN 1 ELSE 0 END AS type\_KC,  
 CASE WHEN ISNULL(p.pitch\_type,'UN') = 'KN' THEN 1 ELSE 0 END AS type\_KN,  
 CASE WHEN ISNULL(p.pitch\_type,'UN') = 'PO' THEN 1 ELSE 0 END AS type\_PO,  
 CASE WHEN ISNULL(p.pitch\_type,'UN') = 'SC' THEN 1 ELSE 0 END AS type\_SC,  
 CASE WHEN ISNULL(p.pitch\_type,'UN') = 'SI' THEN 1 ELSE 0 END AS type\_SI,  
 CASE WHEN ISNULL(p.pitch\_type,'UN') = 'SL' THEN 1 ELSE 0 END AS type\_SL,  
 CASE WHEN ISNULL(p.pitch\_type,'UN') not in ('AS', 'CH', 'CU', 'EP', 'FA', 'FC', 'FF', 'FO', 'FS', 'FT', 'IN', 'KC', 'KN', 'PO', 'SC', 'SI') THEN 1 ELSE 0 END as type\_UN  
 FROM [PitchFx",i,"].[dbo].[Pitches] p  
 INNER JOIN [PitchFx",i,"].[dbo].[AtBats] a on a.ID = p.AtBatID  
 INNER JOIN [Mapping].[dbo].[RSID\_MLBID\_MAP] m on a.pitcher = m.mlbid  
 INNER JOIN [Lahman].[dbo].[Master] ms on ms.retroID = m.rsID", sep="");  
   
 pitches\_raw <-sqlQuery(dbhandle, query);  
   
   
 pitches\_impute\_centroid <- ddply(pitches\_raw, ~rsID, transform,  
 x = impute(x, impute\_fun),  
 y = impute(y, impute\_fun),  
 start\_speed = impute(start\_speed, impute\_fun),  
 end\_speed = impute(end\_speed, impute\_fun),  
 sz\_top = impute(sz\_top, impute\_fun),  
 sz\_bot = impute(sz\_bot, impute\_fun),  
 px = impute(px, impute\_fun),  
 pz = impute(pz, impute\_fun),  
 x0 = impute(x0, impute\_fun),  
 y0 = impute(y0, impute\_fun),  
 z0 = impute(z0, impute\_fun),   
 vx0 = impute(vx0, impute\_fun),  
 vy0 = impute(vy0, impute\_fun),  
 vz0 = impute(vz0, impute\_fun),  
 ax = impute(ax, impute\_fun),  
 ay = impute(ay, impute\_fun),  
 az = impute(az, impute\_fun),  
 break\_y = impute(break\_y, impute\_fun),  
 break\_length = impute(break\_length, impute\_fun),  
 spin\_dir = impute(spin\_dir, impute\_fun),  
 spin\_rate = impute(spin\_rate, impute\_fun)  
 );  
   
 pitches\_aggregate <- ddply(pitches\_impute\_centroid, ~rsID, summarise,  
 season = max(season),  
 x = centroid\_fun(x),  
 y = centroid\_fun(y),  
 start\_speed = centroid\_fun(start\_speed),  
 end\_speed = centroid\_fun(end\_speed),  
 sz\_top = centroid\_fun(sz\_top),  
 sz\_bot = centroid\_fun(sz\_bot),  
 px = centroid\_fun(px),  
 pz = centroid\_fun(pz),  
 x0 = centroid\_fun(x0),  
 y0 = centroid\_fun(y0),  
 z0 = centroid\_fun(z0),   
 vx0 = centroid\_fun(vx0),  
 vy0 = centroid\_fun(vy0),  
 vz0 = centroid\_fun(vz0),  
 ax = centroid\_fun(ax),  
 ay = centroid\_fun(ay),  
 az = centroid\_fun(az),  
 break\_y = centroid\_fun(break\_y),  
 break\_length = centroid\_fun(break\_length),  
 spin\_dir = centroid\_fun(spin\_dir),  
 spin\_rate = centroid\_fun(spin\_rate),  
 num\_pitches = sum(type\_ALL),  
 num\_AB = sum(type\_AB),  
 num\_AS = sum(type\_AS),  
 num\_CH = sum(type\_CH),  
 num\_CU = sum(type\_CU),  
 num\_EP = sum(type\_EP),  
 num\_FA = sum(type\_FA),  
 num\_FC = sum(type\_FC),  
 num\_FF = sum(type\_FF),  
 num\_FO = sum(type\_FO),  
 num\_FS = sum(type\_FS),  
 num\_FT = sum(type\_FT),  
 num\_IN = sum(type\_IN),  
 num\_KC = sum(type\_KC),  
 num\_KN = sum(type\_KN),  
 num\_PO = sum(type\_PO),  
 num\_SC = sum(type\_SC),  
 num\_SI = sum(type\_SI),  
 num\_SL = sum(type\_SL),  
 num\_UN = sum(type\_UN)  
 );  
   
 assign(paste("pitches",i,sep=""), pitches\_aggregate);  
   
 };  
   
 pitches <- rbind(pitches2010, pitches2011, pitches2012, pitches2013, pitches2014, pitches2015, pitches2016);  
 pitches <- pitches[complete.cases(pitches),];  
   
   
 close(dbhandle);  
}

## Load the disabled this

If there is no data cached, load the disable list data from the SQL Server database.

This data contains the list of pitchers on the disable list in each year. Define a new column for season\_1 to denote the previous season. This is required since the pitching data from the previous season will be used to determine if the player will be on the disabled list in the current season.

if (need\_load) {  
 dbhandle <- odbcDriverConnect('driver={SQL Server};server=localhost;database=PitchFx;trusted\_connection=true');  
 query <- "   
 SELECT rsid, 2011 as season\_dl, sum(days) as DLDays  
 FROM [DisabledList].[dbo].[DL2011]  
 WHERE Position in ('LHP','RHP','RP','SP','P')  
 GROUP BY rsid  
 UNION  
 SELECT rsid, 2012 as season\_dl, sum(days) as DLDays  
 FROM [DisabledList].[dbo].[DL2012]  
 WHERE Pos in ('LHP','RHP','RP','SP','P')  
 GROUP BY rsid  
 UNION  
 SELECT rsid, 2013 as season\_dl, sum(days) as DLDays  
 FROM [DisabledList].[dbo].[DL2013]  
 WHERE Position in ('LHP','RHP','RP','SP','P')  
 GROUP BY rsid  
 UNION  
 SELECT rsid, 2014 as season\_dl, sum(days) as DLDays  
 FROM [DisabledList].[dbo].[DL2014]  
 WHERE Position in ('LHP','RHP','RP','SP','P')  
 GROUP BY rsid  
 UNION  
 SELECT rsid, 2015 as season\_dl, sum(days) as DLDays  
 FROM [DisabledList].[dbo].[DL2015]  
 WHERE Position in ('LHP','RHP','RP','SP','P')  
 GROUP BY rsid  
 UNION  
 SELECT rsid, 2016 as season\_dl, sum(days) as DLDays  
 FROM [DisabledList].[dbo].[DL2016]  
 WHERE Position in ('LHP','RHP','RP','SP','P')  
 GROUP BY rsid  
 ";  
   
 dl <- sqlQuery(dbhandle, query);  
 dl <- dl[complete.cases(dl),];  
 dl$season\_1 <- dl$season-1;  
 close(dbhandle);  
 }

## Join the pitch and disabled list data

If there is no data cached, join the pitching data and disabled list data by season.

Define the response variable OnDL to be TRUE if the pitcher is was on the disabled list or FALSE otherwise.

The pitching data from seasons 2010 to 2015 will be used to build and test the model since the disabled list is only available up to the 2016 season.

The pitching data from 2016 will be used to predict which players are most likey be on the disable list in 2017.

if (need\_load) {  
 #use previous season to predict DL in current season  
 pitches\_dl <- merge(x=pitches, y=dl, by.x=c("rsID", "season"), by.y=c("rsid", "season\_1"), all.x = TRUE, all.y=FALSE)  
   
 pitches\_dl[pitches\_dl==""] <- NA; #replace blanks with NA  
   
 pitches\_dl$DLDays[is.na(pitches\_dl$DLDays)] <- 0; #no DL pitchers are on DL for 0 days  
 pitches\_dl$OnDL <- (ifelse(pitches\_dl$DLDays>0, 1, 0));  
   
 drops <- c("season\_dl", "DLDays");  
 pitches\_dl <- pitches\_dl[ , !(names(pitches\_dl) %in% drops)];  
   
 pitches\_dl <- pitches\_dl[complete.cases(pitches\_dl),];  
   
 pitches\_dl\_dataset <- pitches\_dl[pitches\_dl$season < 2016,]; #for modeling  
 pitches\_dl\_predict <- pitches\_dl[pitches\_dl$season == 2016,]; #for 2017 prediction  
   
 #pitches\_dl\_dataset$OnDL <- as.factor(ifelse(pitches\_dl\_dataset$DLDays>0, 'YES', 'NO'));  
   
   
 drops <- c("season");  
 pitches\_dl\_dataset <- pitches\_dl\_dataset[ , !(names(pitches\_dl\_dataset) %in% drops)];  
 pitches\_dl\_predict <- pitches\_dl\_predict[ , !(names(pitches\_dl\_predict) %in% drops)];  
  
   
 #write to csv to save time  
 write.csv(pitches\_dl\_dataset, "pitches\_dl\_dataset.csv");  
 write.csv(pitches\_dl\_predict, "pitches\_dl\_predict.csv");  
   
}

## Transform count variables

Apply Anscombe transformation to count variables

trf\_func <- function(x) {  
 return ( 2\*sqrt(x+3/8));  
 #return ( log(x+1));  
}  
  
pitches\_dl\_dataset$trf\_num\_pitches <- trf\_func(pitches\_dl\_dataset$num\_pitches);  
pitches\_dl\_predict$trf\_num\_pitches <- trf\_func(pitches\_dl\_predict$num\_pitches);  
#   
# for (t in c("AB", "AS", "CH", "CU", "EP", "FA", "FC", "FF", "FO", "FS", "FT", "IN","KC", "KN", "PO", "SC", "SI", "SL", "UN"))  
# {  
# expression <- paste("pitches\_dl\_dataset$num\_", t, sep="");  
# eval\_expression <- eval(parse(text=expression)) /pitches\_dl\_dataset$num\_pitches;  
# assign\_var = paste("pitches\_dl\_dataset$trf\_num\_", t, sep="");  
# assign(assign\_var, eval\_expression)  
#   
# expression <- paste("pitches\_dl\_dataset$num\_", t, sep="");  
# eval\_expression <- eval(parse(text=expression))/pitches\_dl\_predict$num\_pitches;  
# assign\_var = paste("pitches\_dl\_predict$trf\_num\_", t, sep="");  
# assign(assign\_var, eval\_expression)  
#   
# }  
  
  
pitches\_dl\_dataset$trf\_num\_AB <- trf\_func(pitches\_dl\_dataset$num\_AB)  
pitches\_dl\_dataset$trf\_num\_AS <- trf\_func(pitches\_dl\_dataset$num\_AS)  
pitches\_dl\_dataset$trf\_num\_CH <- trf\_func(pitches\_dl\_dataset$num\_CH)  
pitches\_dl\_dataset$trf\_num\_CU <- trf\_func(pitches\_dl\_dataset$num\_CU)  
pitches\_dl\_dataset$trf\_num\_EP <- trf\_func(pitches\_dl\_dataset$num\_EP)  
pitches\_dl\_dataset$trf\_num\_FA <- trf\_func(pitches\_dl\_dataset$num\_FA)  
pitches\_dl\_dataset$trf\_num\_FC <- trf\_func(pitches\_dl\_dataset$num\_FC)  
pitches\_dl\_dataset$trf\_num\_FF <- trf\_func(pitches\_dl\_dataset$num\_FF)  
pitches\_dl\_dataset$trf\_num\_FO <- trf\_func(pitches\_dl\_dataset$num\_FO)  
pitches\_dl\_dataset$trf\_num\_FS <- trf\_func(pitches\_dl\_dataset$num\_FS)  
pitches\_dl\_dataset$trf\_num\_FT <- trf\_func(pitches\_dl\_dataset$num\_FT)  
pitches\_dl\_dataset$trf\_num\_IN <- trf\_func(pitches\_dl\_dataset$num\_IN)  
pitches\_dl\_dataset$trf\_num\_KC <- trf\_func(pitches\_dl\_dataset$num\_KC)  
pitches\_dl\_dataset$trf\_num\_KN <- trf\_func(pitches\_dl\_dataset$num\_KN)  
pitches\_dl\_dataset$trf\_num\_PO <- trf\_func(pitches\_dl\_dataset$num\_PO)  
pitches\_dl\_dataset$trf\_num\_SC <- trf\_func(pitches\_dl\_dataset$num\_SC)  
pitches\_dl\_dataset$trf\_num\_SI <- trf\_func(pitches\_dl\_dataset$num\_SI)  
pitches\_dl\_dataset$trf\_num\_SL <- trf\_func(pitches\_dl\_dataset$num\_SL)  
pitches\_dl\_dataset$trf\_num\_UN <- trf\_func(pitches\_dl\_dataset$num\_UN)  
  
pitches\_dl\_predict$trf\_num\_AB <- trf\_func(pitches\_dl\_predict$num\_AB)  
pitches\_dl\_predict$trf\_num\_AS <- trf\_func(pitches\_dl\_predict$num\_AS)  
pitches\_dl\_predict$trf\_num\_CH <- trf\_func(pitches\_dl\_predict$num\_CH)  
pitches\_dl\_predict$trf\_num\_CU <- trf\_func(pitches\_dl\_predict$num\_CU)  
pitches\_dl\_predict$trf\_num\_EP <- trf\_func(pitches\_dl\_predict$num\_EP)  
pitches\_dl\_predict$trf\_num\_FA <- trf\_func(pitches\_dl\_predict$num\_FA)  
pitches\_dl\_predict$trf\_num\_FC <- trf\_func(pitches\_dl\_predict$num\_FC)  
pitches\_dl\_predict$trf\_num\_FF <- trf\_func(pitches\_dl\_predict$num\_FF)  
pitches\_dl\_predict$trf\_num\_FO <- trf\_func(pitches\_dl\_predict$num\_FO)  
pitches\_dl\_predict$trf\_num\_FS <- trf\_func(pitches\_dl\_predict$num\_FS)  
pitches\_dl\_predict$trf\_num\_FT <- trf\_func(pitches\_dl\_predict$num\_FT)  
pitches\_dl\_predict$trf\_num\_IN <- trf\_func(pitches\_dl\_predict$num\_IN)  
pitches\_dl\_predict$trf\_num\_KC <- trf\_func(pitches\_dl\_predict$num\_KC)  
pitches\_dl\_predict$trf\_num\_KN <- trf\_func(pitches\_dl\_predict$num\_KN)  
pitches\_dl\_predict$trf\_num\_PO <- trf\_func(pitches\_dl\_predict$num\_PO)  
pitches\_dl\_predict$trf\_num\_SC <- trf\_func(pitches\_dl\_predict$num\_SC)  
pitches\_dl\_predict$trf\_num\_SI <- trf\_func(pitches\_dl\_predict$num\_SI)  
pitches\_dl\_predict$trf\_num\_SL <- trf\_func(pitches\_dl\_predict$num\_SL)  
pitches\_dl\_predict$trf\_num\_UN <- trf\_func(pitches\_dl\_predict$num\_UN)  
  
  
  
model\_dataset <- pitches\_dl\_dataset[,-grep( "^num\_" , names( pitches\_dl\_dataset ) )];  
predict\_dataset <- pitches\_dl\_predict[,-grep( "^num\_" , names( pitches\_dl\_predict ) )];

# Exploratory Analysis

## Summary of data set

### List of variables

dependent\_var <- colnames(model\_dataset[,-grep( "^OnDL" , names( model\_dataset ) )])[-1];  
original\_var <- dependent\_var[1:22];  
count\_var <- dependent\_var[22:40];  
response\_var <- "OnDL";

### Summary of continous variables

summary(model\_dataset[which(colnames(model\_dataset) %in% original\_var)]);

## x y start\_speed end\_speed   
## Min. : 75.54 Min. : 97.7 Min. :53.9 Min. :49.80   
## 1st Qu.: 97.86 1st Qu.:145.1 1st Qu.:87.5 1st Qu.:80.70   
## Median :102.15 Median :148.5 Median :89.8 Median :82.70   
## Mean :103.65 Mean :151.0 Mean :89.4 Mean :82.33   
## 3rd Qu.:107.30 3rd Qu.:152.8 3rd Qu.:91.9 3rd Qu.:84.50   
## Max. :139.98 Max. :195.7 Max. :99.4 Max. :91.10   
## sz\_top sz\_bot px pz   
## Min. :0.000 Min. :0.000 Min. :-1.43400 Min. :1.185   
## 1st Qu.:3.390 1st Qu.:1.550 1st Qu.:-0.18875 1st Qu.:2.194   
## Median :3.420 Median :1.580 Median :-0.06750 Median :2.322   
## Mean :3.416 Mean :1.583 Mean :-0.07214 Mean :2.334   
## 3rd Qu.:3.450 3rd Qu.:1.610 3rd Qu.: 0.05550 3rd Qu.:2.459   
## Max. :3.750 Max. :1.805 Max. : 1.33350 Max. :3.638   
## x0 y0 z0 vx0   
## Min. :-4.085 Min. :50 Min. :1.960 Min. :-15.965   
## 1st Qu.:-2.030 1st Qu.:50 1st Qu.:5.643 1st Qu.: -4.080   
## Median :-1.381 Median :50 Median :5.894 Median : 4.511   
## Mean :-0.717 Mean :50 Mean :5.854 Mean : 2.084   
## 3rd Qu.: 1.047 3rd Qu.:50 3rd Qu.:6.136 3rd Qu.: 6.216   
## Max. : 5.229 Max. :50 Max. :7.306 Max. : 11.441   
## vy0 vz0 ax ay   
## Min. :-145.42 Min. :-9.690 Min. :-23.480 Min. :10.34   
## 1st Qu.:-134.48 1st Qu.:-5.546 1st Qu.:-10.075 1st Qu.:25.43   
## Median :-131.43 Median :-4.654 Median : -5.433 Median :26.93   
## Mean :-130.83 Mean :-4.445 Mean : -2.959 Mean :26.98   
## 3rd Qu.:-128.08 3rd Qu.:-3.641 3rd Qu.: 4.211 3rd Qu.:28.60   
## Max. : -78.97 Max. :10.264 Max. : 22.284 Max. :35.57   
## az break\_y break\_length spin\_dir   
## Min. :-44.314 Min. :23.7 Min. : 2.600 Min. : 70.01   
## 1st Qu.:-24.074 1st Qu.:23.8 1st Qu.: 4.900 1st Qu.:161.16   
## Median :-20.948 Median :23.8 Median : 5.800 Median :200.38   
## Mean :-21.181 Mean :23.8 Mean : 5.884 Mean :191.94   
## 3rd Qu.:-17.874 3rd Qu.:23.8 3rd Qu.: 6.700 3rd Qu.:217.43   
## Max. : -6.576 Max. :23.9 Max. :16.650 Max. :325.10   
## spin\_rate trf\_num\_pitches   
## Min. : 445.4 Min. : 2.345   
## 1st Qu.:1663.2 1st Qu.: 29.521   
## Median :1867.4 Median : 54.346   
## Mean :1846.9 Mean : 56.209   
## 3rd Qu.:2053.3 3rd Qu.: 72.811   
## Max. :3000.0 Max. :133.287

### Summary of count variables after transforming

summary(model\_dataset[which(colnames(model\_dataset) %in% count\_var)]);

## trf\_num\_pitches trf\_num\_AB trf\_num\_AS trf\_num\_CH   
## Min. : 2.345 Min. :1.225 Min. :1.225 Min. : 1.225   
## 1st Qu.: 29.521 1st Qu.:1.225 1st Qu.:1.225 1st Qu.: 3.674   
## Median : 54.346 Median :1.225 Median :1.225 Median :10.464   
## Mean : 56.209 Mean :1.227 Mean :1.225 Mean :14.888   
## 3rd Qu.: 72.811 3rd Qu.:1.225 3rd Qu.:1.225 3rd Qu.:22.749   
## Max. :133.287 Max. :2.345 Max. :2.345 Max. :67.775   
## trf\_num\_CU trf\_num\_EP trf\_num\_FA trf\_num\_FC   
## Min. : 1.225 Min. : 1.225 Min. : 1.225 Min. : 1.225   
## 1st Qu.: 1.225 1st Qu.: 1.225 1st Qu.: 1.225 1st Qu.: 1.225   
## Median : 4.637 Median : 1.225 Median : 1.225 Median : 1.225   
## Mean :11.697 Mean : 1.322 Mean : 1.345 Mean : 7.299   
## 3rd Qu.:19.532 3rd Qu.: 1.225 3rd Qu.: 1.225 3rd Qu.: 5.431   
## Max. :68.129 Max. :27.009 Max. :16.778 Max. :87.416   
## trf\_num\_FF trf\_num\_FO trf\_num\_FS trf\_num\_FT   
## Min. : 1.225 Min. : 1.225 Min. : 1.225 Min. : 1.225   
## 1st Qu.:11.380 1st Qu.: 1.225 1st Qu.: 1.225 1st Qu.: 1.225   
## Median :26.486 Median : 1.225 Median : 1.225 Median : 6.124   
## Mean :29.871 Mean : 1.293 Mean : 3.100 Mean :13.998   
## 3rd Qu.:44.917 3rd Qu.: 1.225 3rd Qu.: 1.225 3rd Qu.:21.852   
## Max. :98.881 Max. :35.405 Max. :64.942 Max. :86.357   
## trf\_num\_IN trf\_num\_KC trf\_num\_KN trf\_num\_PO   
## Min. : 1.225 Min. : 1.225 Min. : 1.225 Min. :1.225   
## 1st Qu.: 1.225 1st Qu.: 1.225 1st Qu.: 1.225 1st Qu.:1.225   
## Median : 3.674 Median : 1.225 Median : 1.225 Median :1.225   
## Mean : 3.744 Mean : 3.060 Mean : 1.472 Mean :1.724   
## 3rd Qu.: 5.788 3rd Qu.: 1.225 3rd Qu.: 1.225 3rd Qu.:2.345   
## Max. :13.019 Max. :67.420 Max. :108.247 Max. :7.842   
## trf\_num\_SC trf\_num\_SI trf\_num\_SL   
## Min. : 1.225 Min. : 1.225 Min. : 1.225   
## 1st Qu.: 1.225 1st Qu.: 1.225 1st Qu.: 3.082   
## Median : 1.225 Median : 1.225 Median :14.748   
## Mean : 1.252 Mean :10.211 Mean :18.245   
## 3rd Qu.: 1.225 3rd Qu.: 7.583 3rd Qu.:29.521   
## Max. :20.821 Max. :98.982 Max. :78.521

## Histogram of data

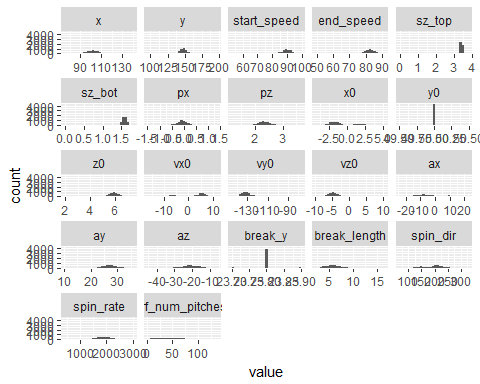
### Continuous Variables

d <- melt(model\_dataset[which(colnames(model\_dataset) %in% original\_var)]);

## No id variables; using all as measure variables

ggplot(d,aes(x = value)) + facet\_wrap(~variable,scales = "free\_x") + geom\_histogram();

## `stat\_bin()` using `bins = 30`. Pick better value with `binwidth`.

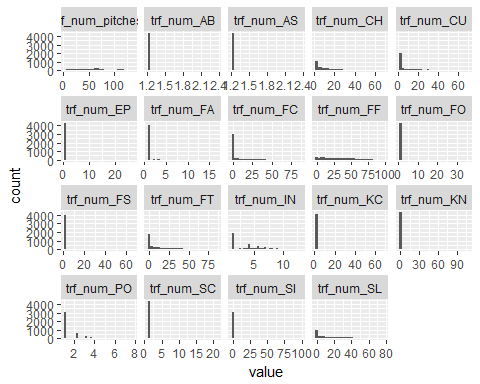
 ### Count Variables

d <- melt(model\_dataset[which(colnames(model\_dataset) %in% count\_var)]);

## No id variables; using all as measure variables

ggplot(d,aes(x = value)) + facet\_wrap(~variable,scales = "free\_x") + geom\_histogram();

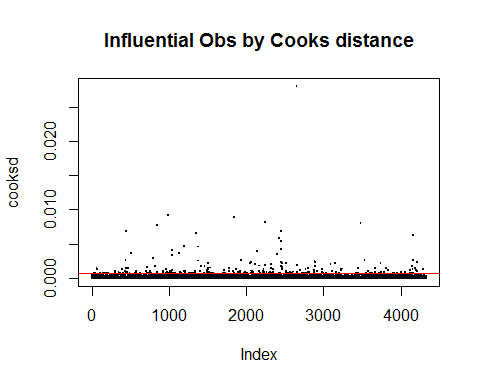
## `stat\_bin()` using `bins = 30`. Pick better value with `binwidth`.



## Outliers

1. Fox, John. (1991). Regression Diagnostics: An Introduction. Sage Publications.

selected\_variables <- dependent\_var;   
  
selected\_i <- which(colnames(model\_dataset) %in% selected\_variables);  
   
formula\_text <- paste(response\_var, "~",  
 paste(names(model\_dataset)[selected\_i], collapse="+"));  
formula <- as.formula(formula\_text);  
  
mod <- glm(formula=formula, data=model\_dataset);  
cooksd <- cooks.distance(mod);  
  
plot(cooksd, pch=".", cex=2, main="Influential Obs by Cooks distance"); # plot cook's distance  
abline(h = 3\*mean(cooksd, na.rm=T), col="red"); # add cutoff line



# text(x=1:length(cooksd)+1, y=cooksd, labels=ifelse(cooksd>10\*mean(cooksd, na.rm=T),names(cooksd),""), col="red")

### Remove outliers

influential <- as.numeric(names(cooksd)[(cooksd > 3\*mean(cooksd, na.rm=T))]);  
  
model\_dataset\_lessOutliers <- model\_dataset[-(influential[!is.na(influential)]), ];

### Summary of data after outliers removed

summary(model\_dataset\_lessOutliers);

## rsID x y start\_speed   
## abadf001: 6 Min. : 75.54 Min. : 97.7 Min. :53.90   
## adamm001: 6 1st Qu.: 97.85 1st Qu.:145.1 1st Qu.:87.60   
## affej001: 6 Median :102.15 Median :148.5 Median :89.85   
## albem001: 6 Mean :103.63 Mean :150.9 Mean :89.47   
## arrij001: 6 3rd Qu.:107.30 3rd Qu.:152.8 3rd Qu.:91.90   
## atchs001: 6 Max. :139.98 Max. :195.7 Max. :99.40   
## (Other) :4084   
## end\_speed sz\_top sz\_bot px   
## Min. :49.80 Min. :0.000 Min. :0.000 Min. :-1.43400   
## 1st Qu.:80.70 1st Qu.:3.390 1st Qu.:1.550 1st Qu.:-0.18862   
## Median :82.70 Median :3.420 Median :1.580 Median :-0.06750   
## Mean :82.39 Mean :3.418 Mean :1.584 Mean :-0.07198   
## 3rd Qu.:84.50 3rd Qu.:3.450 3rd Qu.:1.610 3rd Qu.: 0.05563   
## Max. :91.10 Max. :3.750 Max. :1.805 Max. : 1.33350   
##   
## pz x0 y0 z0   
## Min. :1.185 Min. :-4.0845 Min. :50 Min. :1.960   
## 1st Qu.:2.197 1st Qu.:-2.0326 1st Qu.:50 1st Qu.:5.649   
## Median :2.323 Median :-1.3820 Median :50 Median :5.897   
## Mean :2.335 Mean :-0.7262 Mean :50 Mean :5.860   
## 3rd Qu.:2.460 3rd Qu.: 1.0146 3rd Qu.:50 3rd Qu.:6.137   
## Max. :3.638 Max. : 5.2295 Max. :50 Max. :7.306   
##   
## vx0 vy0 vz0 ax   
## Min. :-15.965 Min. :-145.42 Min. :-9.690 Min. :-23.480   
## 1st Qu.: -3.996 1st Qu.:-134.50 1st Qu.:-5.558 1st Qu.:-10.087   
## Median : 4.506 Median :-131.50 Median :-4.671 Median : -5.486   
## Mean : 2.113 Mean :-130.93 Mean :-4.483 Mean : -3.003   
## 3rd Qu.: 6.231 3rd Qu.:-128.19 3rd Qu.:-3.680 3rd Qu.: 4.138   
## Max. : 11.441 Max. : -78.97 Max. :10.264 Max. : 22.284   
##   
## ay az break\_y break\_length   
## Min. :10.34 Min. :-44.314 Min. :23.7 Min. : 2.600   
## 1st Qu.:25.47 1st Qu.:-23.965 1st Qu.:23.8 1st Qu.: 4.900   
## Median :26.95 Median :-20.862 Median :23.8 Median : 5.700   
## Mean :27.00 Mean :-21.074 Mean :23.8 Mean : 5.848   
## 3rd Qu.:28.62 3rd Qu.:-17.827 3rd Qu.:23.8 3rd Qu.: 6.700   
## Max. :35.44 Max. : -8.132 Max. :23.9 Max. :16.650   
##   
## spin\_dir spin\_rate OnDL trf\_num\_pitches   
## Min. : 70.01 Min. : 507.6 Min. :0.0000 Min. : 2.345   
## 1st Qu.:161.79 1st Qu.:1668.6 1st Qu.:0.0000 1st Qu.: 29.283   
## Median :200.39 Median :1870.7 Median :0.0000 Median : 53.568   
## Mean :192.05 Mean :1850.2 Mean :0.2371 Mean : 55.531   
## 3rd Qu.:217.19 3rd Qu.:2055.2 3rd Qu.:0.0000 3rd Qu.: 72.024   
## Max. :325.10 Max. :3000.0 Max. :1.0000 Max. :133.287   
##   
## trf\_num\_AB trf\_num\_AS trf\_num\_CH trf\_num\_CU   
## Min. :1.225 Min. :1.225 Min. : 1.225 Min. : 1.225   
## 1st Qu.:1.225 1st Qu.:1.225 1st Qu.: 3.674 1st Qu.: 1.225   
## Median :1.225 Median :1.225 Median :10.654 Median : 4.637   
## Mean :1.225 Mean :1.225 Mean :14.780 Mean :11.646   
## 3rd Qu.:1.225 3rd Qu.:1.225 3rd Qu.:22.506 3rd Qu.:19.326   
## Max. :1.225 Max. :2.345 Max. :67.775 Max. :68.129   
##   
## trf\_num\_EP trf\_num\_FA trf\_num\_FC trf\_num\_FF   
## Min. : 1.225 Min. : 1.225 Min. : 1.225 Min. : 1.225   
## 1st Qu.: 1.225 1st Qu.: 1.225 1st Qu.: 1.225 1st Qu.:11.554   
## Median : 1.225 Median : 1.225 Median : 1.225 Median :26.486   
## Mean : 1.287 Mean : 1.329 Mean : 7.061 Mean :29.889   
## 3rd Qu.: 1.225 3rd Qu.: 1.225 3rd Qu.: 5.050 3rd Qu.:44.805   
## Max. :27.009 Max. :16.778 Max. :87.416 Max. :98.598   
##   
## trf\_num\_FO trf\_num\_FS trf\_num\_FT trf\_num\_IN   
## Min. : 1.225 Min. : 1.225 Min. : 1.225 Min. : 1.225   
## 1st Qu.: 1.225 1st Qu.: 1.225 1st Qu.: 1.225 1st Qu.: 1.225   
## Median : 1.225 Median : 1.225 Median : 6.124 Median : 3.674   
## Mean : 1.233 Mean : 2.949 Mean :14.025 Mean : 3.717   
## 3rd Qu.: 1.225 3rd Qu.: 1.225 3rd Qu.:22.125 3rd Qu.: 5.788   
## Max. :11.023 Max. :57.180 Max. :86.357 Max. :13.019   
##   
## trf\_num\_KC trf\_num\_KN trf\_num\_PO trf\_num\_SC   
## Min. : 1.225 Min. : 1.225 Min. :1.225 Min. : 1.225   
## 1st Qu.: 1.225 1st Qu.: 1.225 1st Qu.:1.225 1st Qu.: 1.225   
## Median : 1.225 Median : 1.225 Median :1.225 Median : 1.225   
## Mean : 2.858 Mean : 1.434 Mean :1.714 Mean : 1.249   
## 3rd Qu.: 1.225 3rd Qu.: 1.225 3rd Qu.:2.345 3rd Qu.: 1.225   
## Max. :65.920 Max. :108.247 Max. :6.745 Max. :20.821   
##   
## trf\_num\_SI trf\_num\_SL trf\_num\_UN   
## Min. : 1.225 Min. : 1.225 Min. : 1.225   
## 1st Qu.: 1.225 1st Qu.: 3.082 1st Qu.:11.726   
## Median : 1.225 Median :14.883 Median :20.821   
## Mean : 9.778 Mean :18.228 Mean :23.235   
## 3rd Qu.: 6.745 3rd Qu.:29.283 3rd Qu.:32.535   
## Max. :92.030 Max. :78.521 Max. :79.231   
##

## Histogram of data

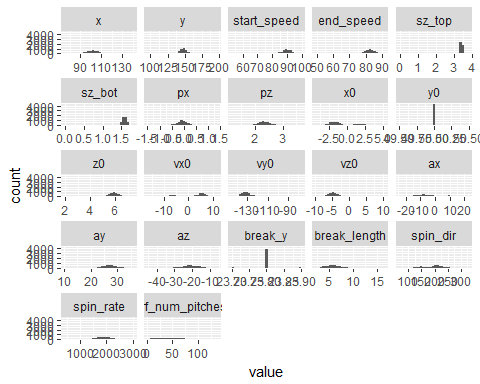
### Continuous Variables with outliers

d <- melt(model\_dataset[which(colnames(model\_dataset) %in% original\_var)]);

## No id variables; using all as measure variables

ggplot(d,aes(x = value)) + facet\_wrap(~variable,scales = "free\_x") + geom\_histogram();

## `stat\_bin()` using `bins = 30`. Pick better value with `binwidth`.

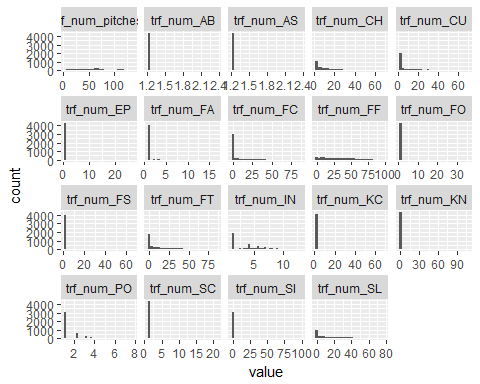
 ### Count Variables with outliers

d <- melt(model\_dataset[which(colnames(model\_dataset) %in% count\_var)]);

## No id variables; using all as measure variables

ggplot(d,aes(x = value)) + facet\_wrap(~variable,scales = "free\_x") + geom\_histogram();

## `stat\_bin()` using `bins = 30`. Pick better value with `binwidth`.

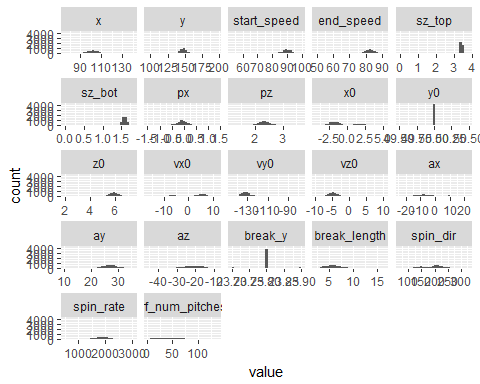
 ### Continuous Variables without outliers

d <- melt(model\_dataset\_lessOutliers[which(colnames(model\_dataset\_lessOutliers) %in% original\_var)]);

## No id variables; using all as measure variables

ggplot(d,aes(x = value)) + facet\_wrap(~variable,scales = "free\_x") + geom\_histogram();

## `stat\_bin()` using `bins = 30`. Pick better value with `binwidth`.

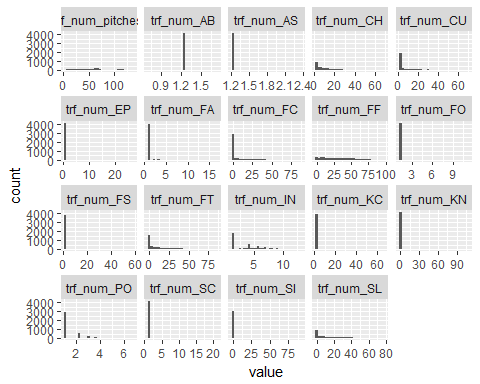
 ### Count Variables without outlisers

d <- melt(model\_dataset\_lessOutliers[which(colnames(model\_dataset\_lessOutliers) %in% count\_var)]);

## No id variables; using all as measure variables

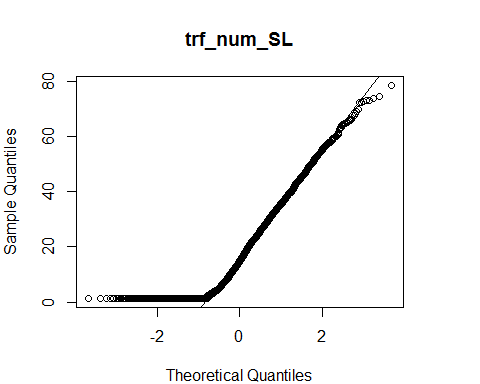
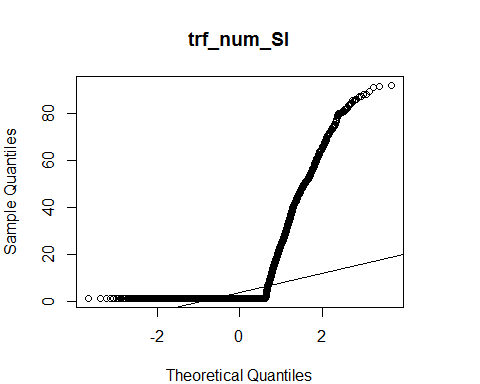
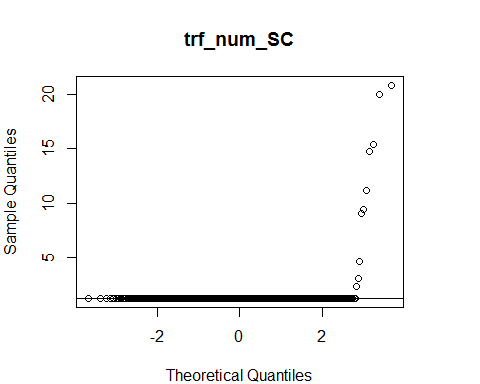
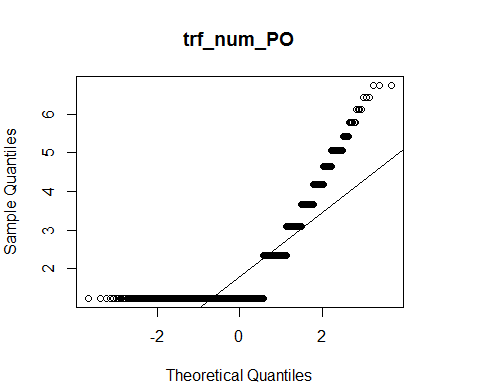
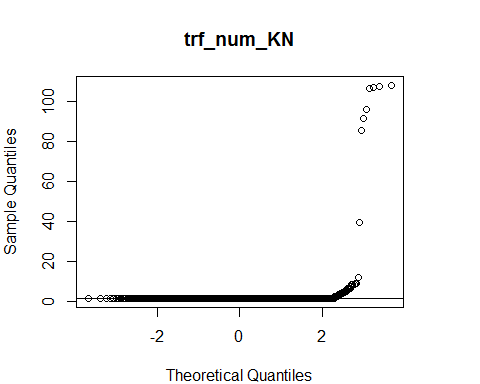
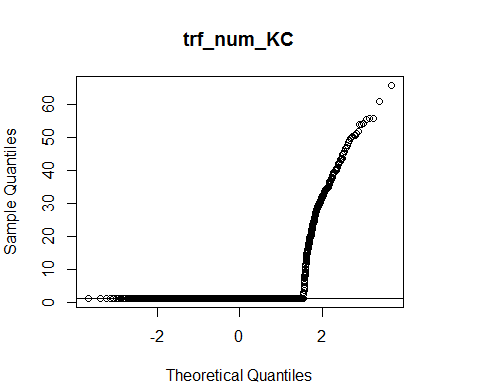
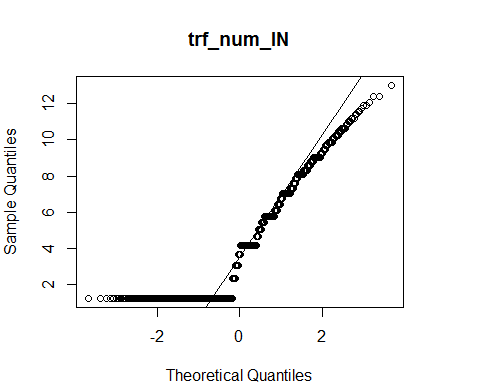
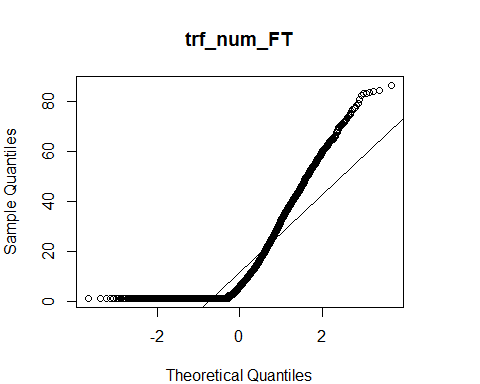
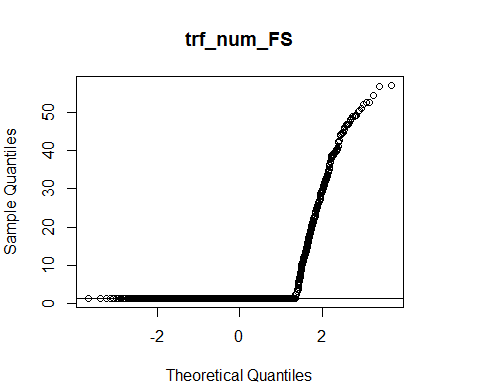
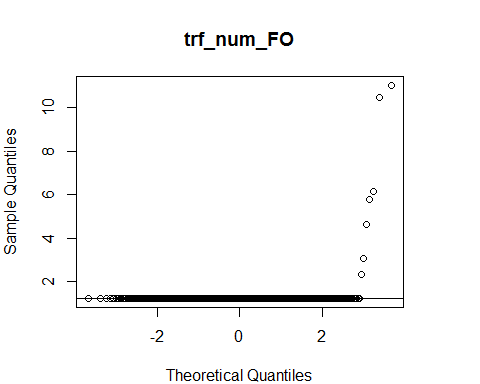
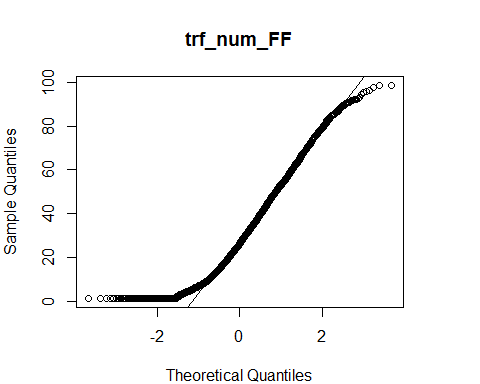
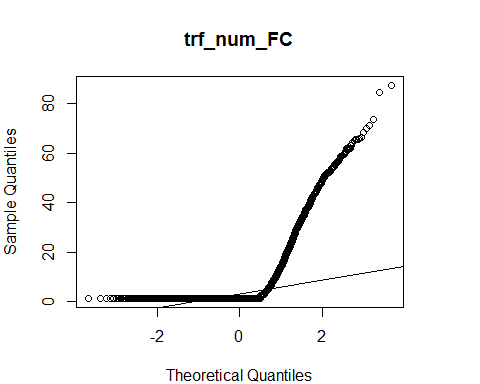
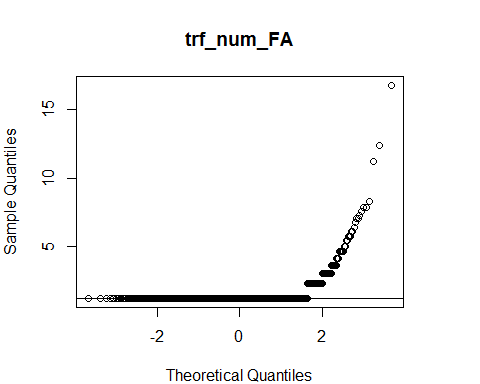
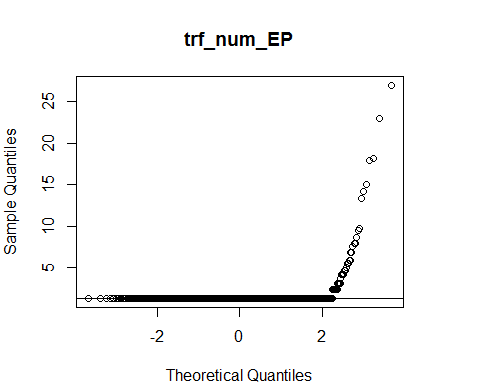
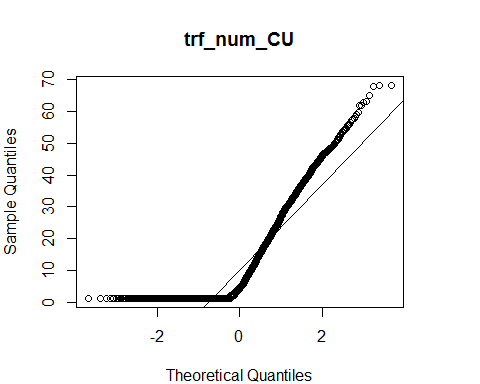
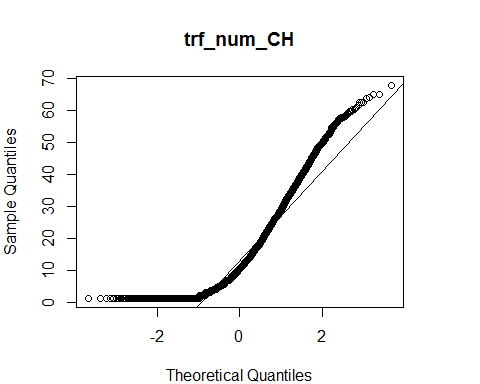
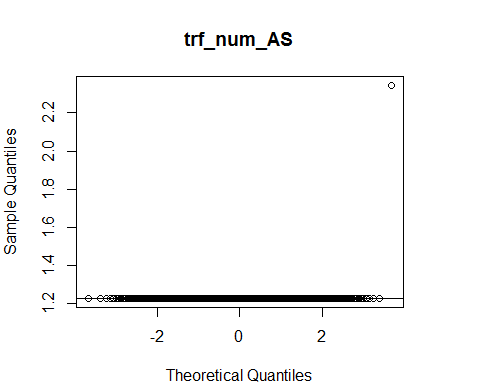
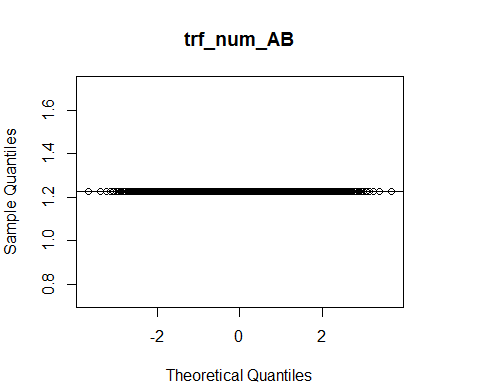
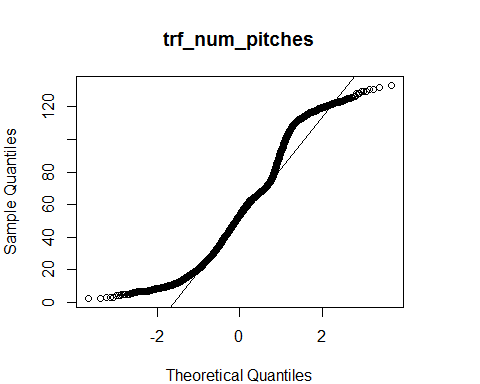
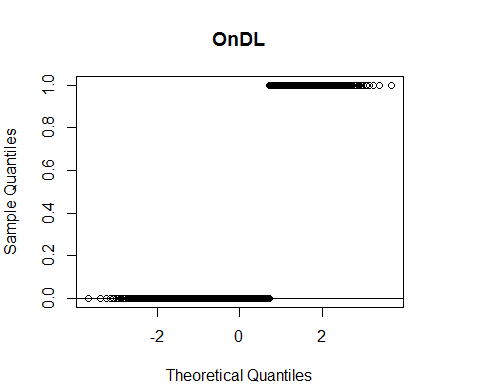
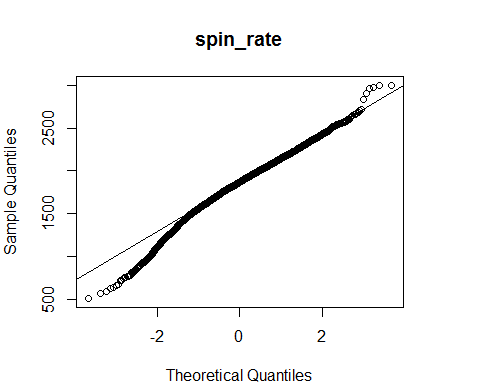
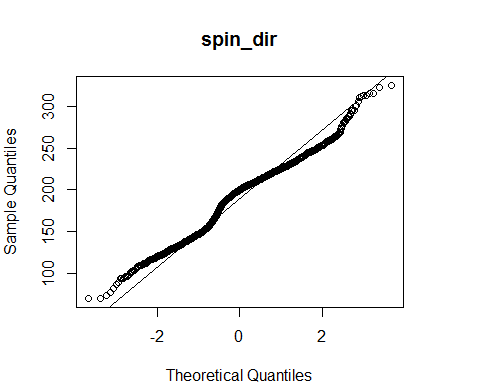
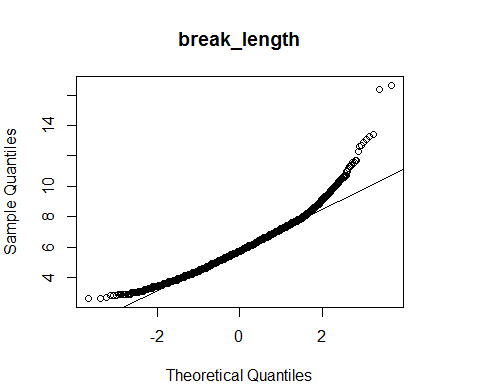
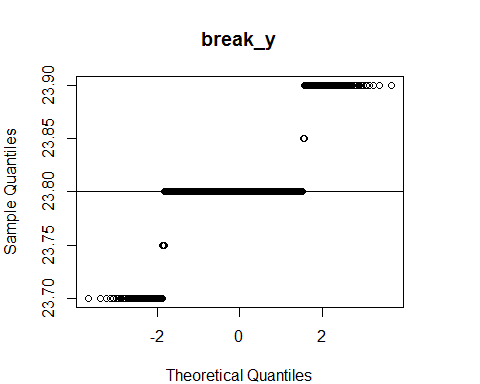
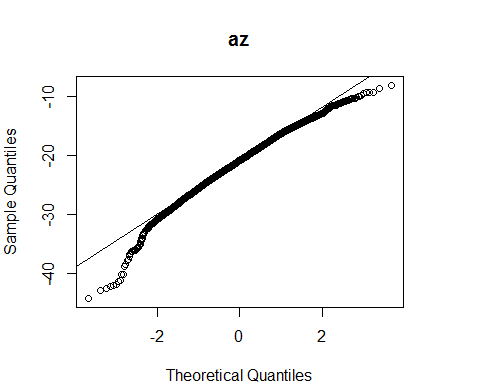
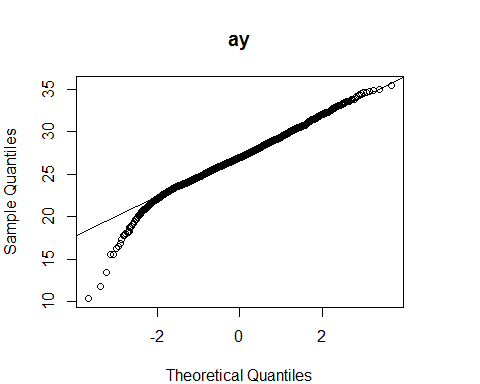
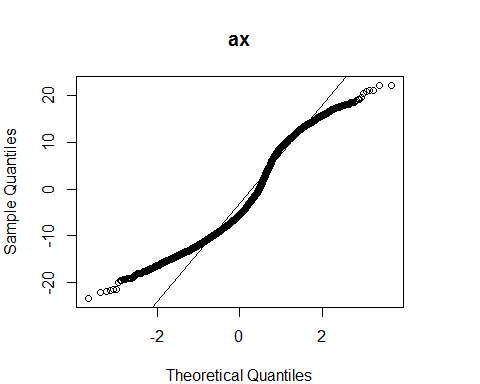
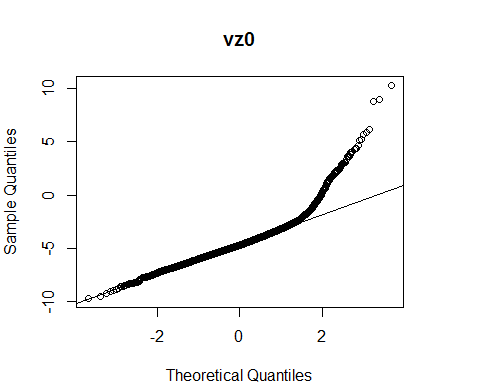
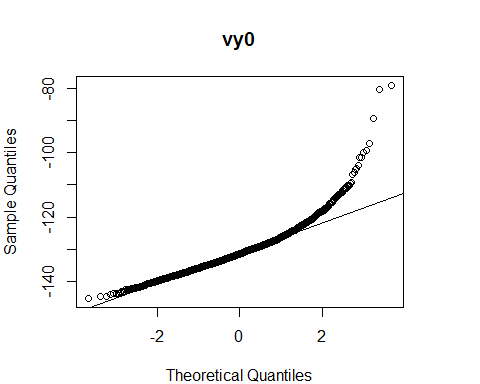
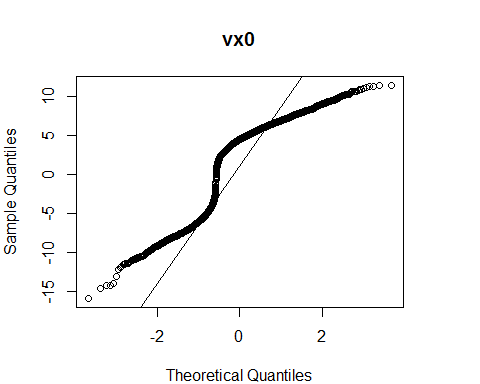
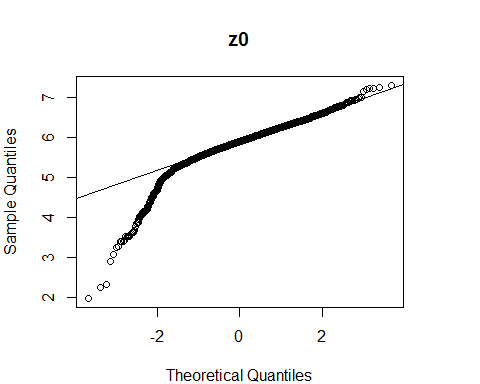
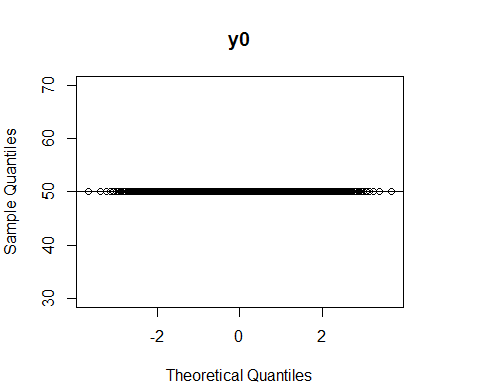
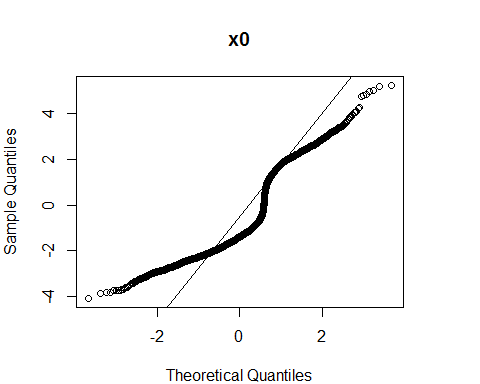
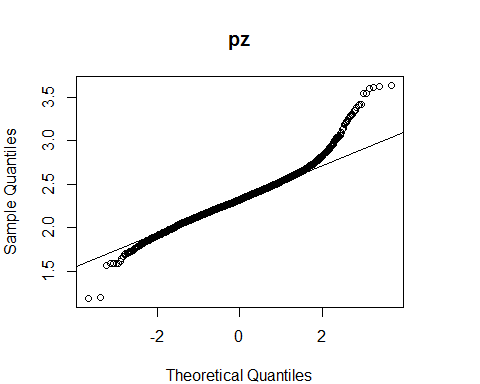
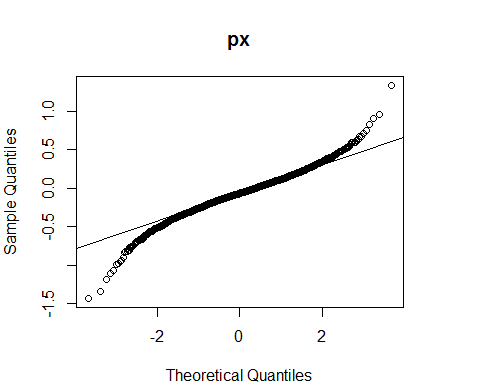
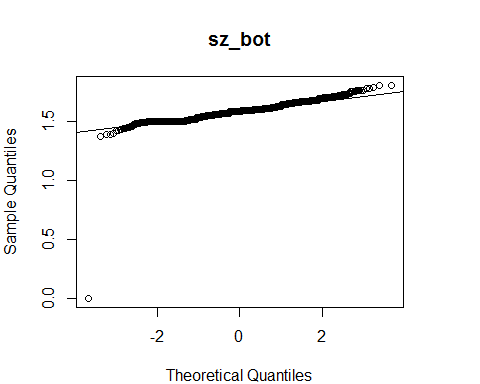
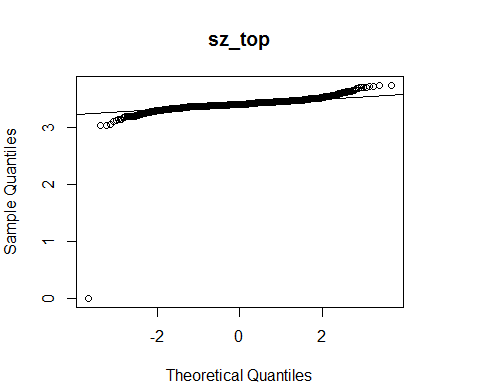
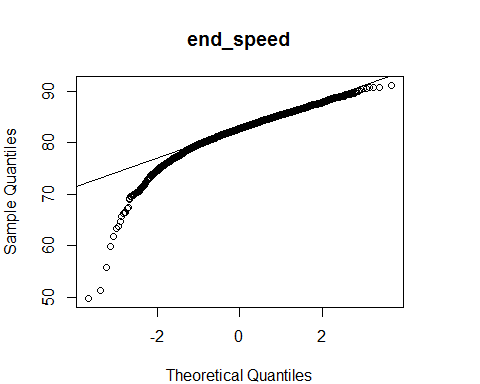
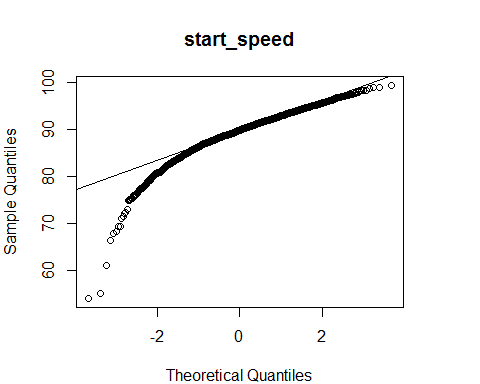
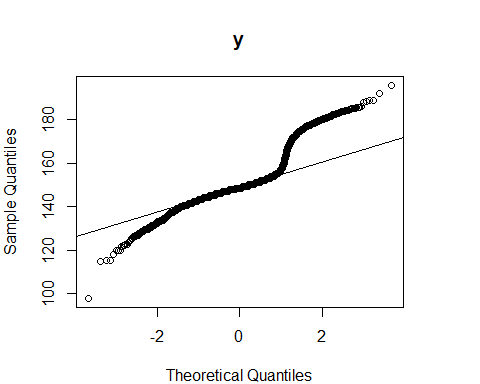
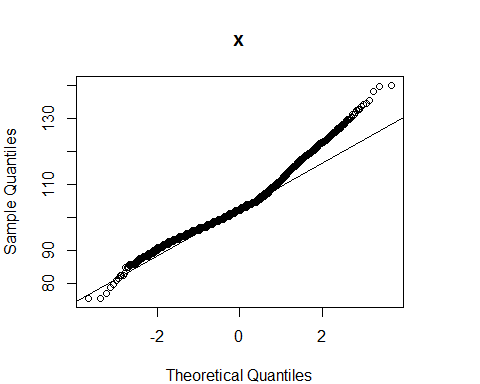
ggplot(d,aes(x = value)) + facet\_wrap(~variable,scales = "free\_x") + geom\_histogram();

## `stat\_bin()` using `bins = 30`. Pick better value with `binwidth`.



## Check normality using QQ plot without outliers

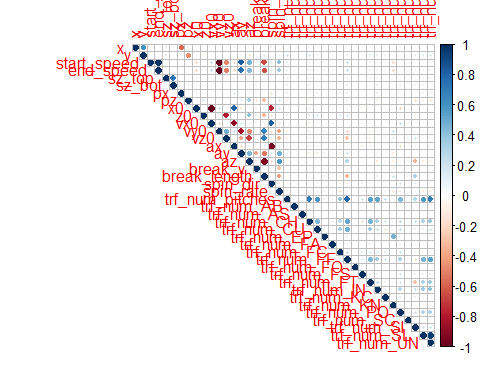
par(mar=c(4,4,4,4))  
  
for (i in 2:(ncol(model\_dataset\_lessOutliers)-1)){   
 tmp <- model\_dataset\_lessOutliers[, i];  
 qqnorm(tmp, main = colnames(model\_dataset\_lessOutliers[i]));  
 qqline(tmp);  
}



## Display correlation

### All variables

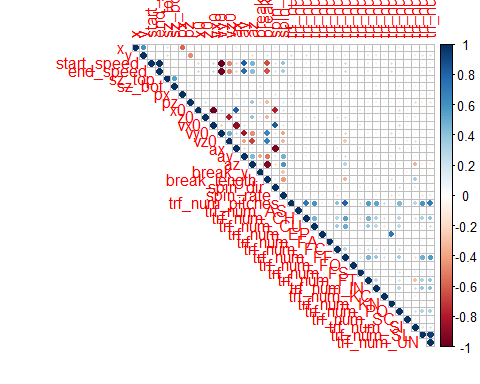
numeric\_dataset <- model\_dataset[sapply(model\_dataset, is.numeric)];  
  
#ignore column y0 since there is 0 variance  
numeric\_dataset <- numeric\_dataset[ , !(names(numeric\_dataset) %in% c("y0", "OnDL"))];  
  
#numeric\_dataset <- numeric\_dataset[1:(ncol(numeric\_dataset))];  
m <- cor(numeric\_dataset);  
corrplot::corrplot(m, type="upper");



#corrplot::corrplot.mixed(m);

### All variables without outliers

numeric\_dataset\_lessOutliers <- model\_dataset\_lessOutliers[sapply(model\_dataset\_lessOutliers, is.numeric)];  
  
#ignore column y0 since there is 0 variance  
numeric\_dataset\_lessOutliers <- numeric\_dataset\_lessOutliers[ , !(names(numeric\_dataset\_lessOutliers) %in% c("y0", "trf\_num\_AB", "OnDL"))];  
  
#numeric\_dataset <- numeric\_dataset[1:(ncol(numeric\_dataset))];  
m\_lessOutliers <- cor(numeric\_dataset\_lessOutliers);  
corrplot::corrplot(m\_lessOutliers, type="upper");



#corrplot::corrplot.mixed(m\_lessOutliers);

# Model Building with all data

## Model 1 All variables

### Create training and testing set using 75% training and 25% testing

set.seed(123)  
  
train <- createDataPartition(model\_dataset$OnDL, p=0.75, list=FALSE);  
  
training <- model\_dataset[train,];  
write.csv(training, "training.csv");  
  
testing <- model\_dataset[-train,];  
  
  
threshold <- 0.4;

### Construct Model

selected\_variables <- dependent\_var;   
  
selected\_i <- which(colnames(training) %in% selected\_variables);  
   
formula\_text <- paste(response\_var, "~",  
 paste(names(training)[selected\_i], collapse="+"));  
formula <- as.formula(formula\_text);  
  
mod\_1 = glm(formula = formula , family=binomial(logit), data=training);

### Summary

The summary shows that variables which have some significance to the outcomes are: end\_speed, sz\_bot, pz, z0, vz0, break\_y, break\_length, trf\_num\_pitches, trf\_num\_CH, trf\_num\_FT, trf\_num\_SI, trf\_num\_UN

summary(mod\_1);

##   
## Call:  
## glm(formula = formula, family = binomial(logit), data = training)  
##   
## Deviance Residuals:   
## Min 1Q Median 3Q Max   
## -1.6157 -0.7642 -0.5675 0.8835 2.4404   
##   
## Coefficients: (1 not defined because of singularities)  
## Estimate Std. Error z value Pr(>|z|)   
## (Intercept) 4.791e+01 4.331e+03 0.011 0.99117   
## x -1.438e-02 1.621e-02 -0.887 0.37498   
## y 1.796e-02 1.062e-02 1.691 0.09086 .   
## start\_speed -3.305e-01 5.816e-01 -0.568 0.56982   
## end\_speed 1.305e-01 1.581e-01 0.825 0.40922   
## sz\_top 8.380e-01 1.011e+00 0.829 0.40712   
## sz\_bot 3.676e-01 1.052e+00 0.350 0.72666   
## px -3.698e-01 5.575e-01 -0.663 0.50714   
## pz 5.774e-01 4.710e-01 1.226 0.22027   
## x0 2.618e-01 3.470e-01 0.754 0.45061   
## y0 NA NA NA NA   
## z0 -2.719e-01 3.982e-01 -0.683 0.49480   
## vx0 1.136e-01 1.381e-01 0.823 0.41072   
## vy0 -2.127e-01 4.071e-01 -0.522 0.60140   
## vz0 -1.967e-01 1.473e-01 -1.335 0.18183   
## ax 3.755e-02 3.098e-02 1.212 0.22551   
## ay -2.311e-02 5.081e-02 -0.455 0.64918   
## az -3.089e-03 6.215e-02 -0.050 0.96035   
## break\_y -3.087e+00 1.952e+00 -1.582 0.11366   
## break\_length 2.243e-01 1.959e-01 1.145 0.25221   
## spin\_dir 2.884e-03 4.366e-03 0.660 0.50894   
## spin\_rate -1.923e-05 3.034e-04 -0.063 0.94947   
## trf\_num\_pitches 2.782e-02 9.278e-03 2.999 0.00271 \*\*  
## trf\_num\_AB 3.076e-01 7.137e-01 0.431 0.66642   
## trf\_num\_AS 1.708e+01 3.531e+03 0.005 0.99614   
## trf\_num\_CH -1.502e-02 5.985e-03 -2.510 0.01206 \*   
## trf\_num\_CU 3.915e-03 5.493e-03 0.713 0.47598   
## trf\_num\_EP -8.667e-03 3.930e-02 -0.221 0.82547   
## trf\_num\_FA 4.091e-02 5.976e-02 0.685 0.49366   
## trf\_num\_FC -3.243e-03 4.874e-03 -0.665 0.50578   
## trf\_num\_FF 2.446e-04 6.576e-03 0.037 0.97034   
## trf\_num\_FO -1.027e-02 3.799e-02 -0.270 0.78703   
## trf\_num\_FS -7.668e-03 6.925e-03 -1.107 0.26814   
## trf\_num\_FT -4.951e-03 5.041e-03 -0.982 0.32606   
## trf\_num\_IN -4.556e-02 1.895e-02 -2.404 0.01621 \*   
## trf\_num\_KC -1.855e-03 6.919e-03 -0.268 0.78867   
## trf\_num\_KN -1.538e-02 1.501e-02 -1.025 0.30532   
## trf\_num\_PO 2.985e-02 4.956e-02 0.602 0.54695   
## trf\_num\_SC -1.106e+01 1.908e+02 -0.058 0.95379   
## trf\_num\_SI -8.891e-03 5.803e-03 -1.532 0.12550   
## trf\_num\_SL -1.868e-03 9.015e-03 -0.207 0.83583   
## trf\_num\_UN 6.808e-04 1.120e-02 0.061 0.95154   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## (Dispersion parameter for binomial family taken to be 1)  
##   
## Null deviance: 3667.1 on 3245 degrees of freedom  
## Residual deviance: 3322.1 on 3205 degrees of freedom  
## AIC: 3404.1  
##   
## Number of Fisher Scoring iterations: 16

### COefficients

mod\_1$coefficients;

## (Intercept) x y start\_speed   
## 4.790750e+01 -1.438374e-02 1.795870e-02 -3.305324e-01   
## end\_speed sz\_top sz\_bot px   
## 1.305043e-01 8.380272e-01 3.676150e-01 -3.698234e-01   
## pz x0 y0 z0   
## 5.773934e-01 2.617837e-01 NA -2.718682e-01   
## vx0 vy0 vz0 ax   
## 1.136216e-01 -2.126678e-01 -1.967318e-01 3.754882e-02   
## ay az break\_y break\_length   
## -2.311265e-02 -3.089469e-03 -3.087421e+00 2.243344e-01   
## spin\_dir spin\_rate trf\_num\_pitches trf\_num\_AB   
## 2.883562e-03 -1.922527e-05 2.782237e-02 3.076493e-01   
## trf\_num\_AS trf\_num\_CH trf\_num\_CU trf\_num\_EP   
## 1.707741e+01 -1.502439e-02 3.915119e-03 -8.666885e-03   
## trf\_num\_FA trf\_num\_FC trf\_num\_FF trf\_num\_FO   
## 4.090886e-02 -3.243093e-03 2.445568e-04 -1.026509e-02   
## trf\_num\_FS trf\_num\_FT trf\_num\_IN trf\_num\_KC   
## -7.668036e-03 -4.951131e-03 -4.555533e-02 -1.854666e-03   
## trf\_num\_KN trf\_num\_PO trf\_num\_SC trf\_num\_SI   
## -1.538422e-02 2.985496e-02 -1.105629e+01 -8.890802e-03   
## trf\_num\_SL trf\_num\_UN   
## -1.868304e-03 6.808348e-04

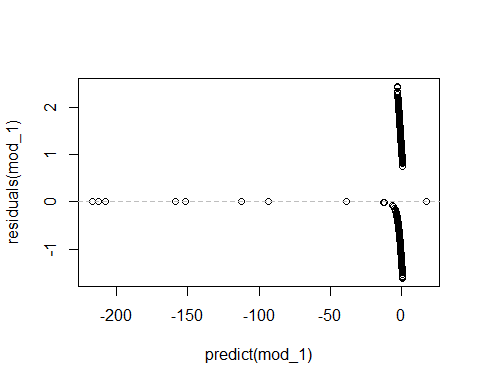
### Odds Ratio

exp(mod\_1$coefficients);

## (Intercept) x y start\_speed   
## 6.396778e+20 9.857192e-01 1.018121e+00 7.185411e-01   
## end\_speed sz\_top sz\_bot px   
## 1.139403e+00 2.311802e+00 1.444286e+00 6.908563e-01   
## pz x0 y0 z0   
## 1.781389e+00 1.299246e+00 NA 7.619547e-01   
## vx0 vy0 vz0 ax   
## 1.120328e+00 8.084246e-01 8.214109e-01 1.038263e+00   
## ay az break\_y break\_length   
## 9.771524e-01 9.969153e-01 4.561945e-02 1.251489e+00   
## spin\_dir spin\_rate trf\_num\_pitches trf\_num\_AB   
## 1.002888e+00 9.999808e-01 1.028213e+00 1.360224e+00   
## trf\_num\_AS trf\_num\_CH trf\_num\_CU trf\_num\_EP   
## 2.609906e+07 9.850879e-01 1.003923e+00 9.913706e-01   
## trf\_num\_FA trf\_num\_FC trf\_num\_FF trf\_num\_FO   
## 1.041757e+00 9.967622e-01 1.000245e+00 9.897874e-01   
## trf\_num\_FS trf\_num\_FT trf\_num\_IN trf\_num\_KC   
## 9.923613e-01 9.950611e-01 9.554667e-01 9.981471e-01   
## trf\_num\_KN trf\_num\_PO trf\_num\_SC trf\_num\_SI   
## 9.847335e-01 1.030305e+00 1.578746e-05 9.911486e-01   
## trf\_num\_SL trf\_num\_UN   
## 9.981334e-01 1.000681e+00

### Residual

plot(predict(mod\_1),residuals(mod\_1));  
abline(h=0,lty=2,col="grey");



### Performance

pred <- ifelse(predict(mod\_1, testing, type='response') > threshold, 1, 0)  
confusionMatrix(data=pred, reference=testing$OnDL, positive='1');

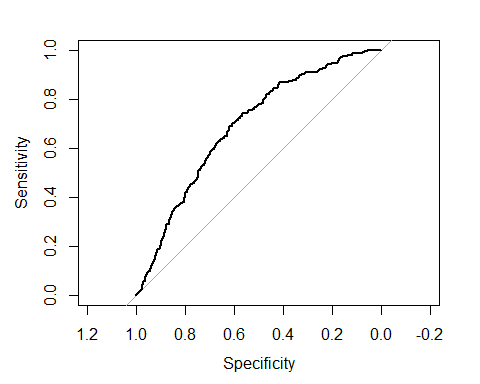
## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction 0 1  
## 0 673 244  
## 1 85 79  
##   
## Accuracy : 0.6957   
## 95% CI : (0.6673, 0.723)  
## No Information Rate : 0.7012   
## P-Value [Acc > NIR] : 0.6684   
##   
## Kappa : 0.1542   
## Mcnemar's Test P-Value : <2e-16   
##   
## Sensitivity : 0.24458   
## Specificity : 0.88786   
## Pos Pred Value : 0.48171   
## Neg Pred Value : 0.73391   
## Prevalence : 0.29880   
## Detection Rate : 0.07308   
## Detection Prevalence : 0.15171   
## Balanced Accuracy : 0.56622   
##   
## 'Positive' Class : 1   
##

### ROC curve

prob <- predict(mod\_1, testing, type='response');  
g1 <- roc(OnDL ~ prob, data = testing);  
roc.curve(testing$OnDL, prob, plotit =F)

## Area under the curve (AUC): 0.690

plot(g1)



# Model Building without outliers

## Model 1(b) All variables

### Create training and testing set using 75% training and 25% testing

train\_lessOutliers <- createDataPartition(model\_dataset\_lessOutliers$OnDL, p=0.75, list=FALSE);  
  
training\_lessOutliers <- model\_dataset\_lessOutliers[train\_lessOutliers,];  
  
testing\_lessOutliers <- model\_dataset\_lessOutliers[-train\_lessOutliers,];

### Construct Model

selected\_variables <- dependent\_var;   
  
selected\_i <- which(colnames(training\_lessOutliers) %in% selected\_variables);  
   
formula\_text <- paste(response\_var, "~",  
 paste(names(training\_lessOutliers)[selected\_i], collapse="+"));  
formula <- as.formula(formula\_text);  
  
mod\_1b = glm(formula = formula , family=binomial(logit), data=training\_lessOutliers);

### Summary

The summary shows that variables which have some significance to the outcomes are: end\_speed, break\_length, trf\_num\_pitches, trf\_num\_FA, trf\_num\_FT, trf\_num\_KN

summary(mod\_1b);

##   
## Call:  
## glm(formula = formula, family = binomial(logit), data = training\_lessOutliers)  
##   
## Deviance Residuals:   
## Min 1Q Median 3Q Max   
## -1.6558 -0.7240 -0.5000 -0.1471 2.4884   
##   
## Coefficients: (2 not defined because of singularities)  
## Estimate Std. Error z value Pr(>|z|)   
## (Intercept) 8.745e+01 1.941e+04 0.005 0.996406   
## x -1.927e-02 2.064e-02 -0.934 0.350411   
## y 2.512e-02 1.334e-02 1.883 0.059667 .   
## start\_speed -1.310e+00 6.491e-01 -2.018 0.043632 \*   
## end\_speed -2.380e-02 1.773e-01 -0.134 0.893222   
## sz\_top -2.253e-01 1.090e+00 -0.207 0.836265   
## sz\_bot 1.255e+00 1.144e+00 1.097 0.272603   
## px 6.341e-02 6.925e-01 0.092 0.927040   
## pz 1.608e+00 5.581e-01 2.881 0.003964 \*\*   
## x0 -6.459e-02 3.913e-01 -0.165 0.868904   
## y0 NA NA NA NA   
## z0 -1.352e+00 4.603e-01 -2.937 0.003316 \*\*   
## vx0 1.400e-02 1.543e-01 0.091 0.927698   
## vy0 -1.025e+00 4.554e-01 -2.251 0.024390 \*   
## vz0 -6.471e-01 1.709e-01 -3.786 0.000153 \*\*\*  
## ax 3.611e-02 3.505e-02 1.030 0.302869   
## ay -1.395e-01 5.681e-02 -2.455 0.014090 \*   
## az -6.398e-02 7.528e-02 -0.850 0.395350   
## break\_y -4.295e+00 2.111e+00 -2.035 0.041877 \*   
## break\_length 3.159e-01 2.342e-01 1.349 0.177276   
## spin\_dir 1.688e-03 5.564e-03 0.303 0.761585   
## spin\_rate 2.461e-04 3.632e-04 0.677 0.498137   
## trf\_num\_pitches 1.804e-02 1.058e-02 1.705 0.088223 .   
## trf\_num\_AB NA NA NA NA   
## trf\_num\_AS 1.991e+01 1.582e+04 0.001 0.998996   
## trf\_num\_CH -6.590e-03 6.584e-03 -1.001 0.316856   
## trf\_num\_CU 5.427e-03 5.989e-03 0.906 0.364896   
## trf\_num\_EP -2.044e-01 1.251e-01 -1.634 0.102317   
## trf\_num\_FA -3.213e-01 1.056e-01 -3.044 0.002336 \*\*   
## trf\_num\_FC 4.582e-04 5.381e-03 0.085 0.932137   
## trf\_num\_FF 1.002e-02 7.739e-03 1.295 0.195475   
## trf\_num\_FO -7.047e-02 2.939e-01 -0.240 0.810492   
## trf\_num\_FS 3.050e-03 7.829e-03 0.390 0.696827   
## trf\_num\_FT 2.912e-03 5.675e-03 0.513 0.607871   
## trf\_num\_IN -7.009e-02 2.049e-02 -3.421 0.000623 \*\*\*  
## trf\_num\_KC -2.082e-03 7.514e-03 -0.277 0.781735   
## trf\_num\_KN -9.878e+00 2.335e+02 -0.042 0.966259   
## trf\_num\_PO 3.238e-02 5.354e-02 0.605 0.545362   
## trf\_num\_SC -1.321e+01 8.855e+02 -0.015 0.988097   
## trf\_num\_SI -3.658e-03 6.668e-03 -0.549 0.583337   
## trf\_num\_SL 8.721e-03 1.002e-02 0.870 0.384217   
## trf\_num\_UN -7.347e-03 1.242e-02 -0.592 0.554040   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## (Dispersion parameter for binomial family taken to be 1)  
##   
## Null deviance: 3345.5 on 3089 degrees of freedom  
## Residual deviance: 2897.1 on 3050 degrees of freedom  
## AIC: 2977.1  
##   
## Number of Fisher Scoring iterations: 19

### COefficients

mod\_1b$coefficients;

## (Intercept) x y start\_speed   
## 8.744651e+01 -1.927374e-02 2.512222e-02 -1.309720e+00   
## end\_speed sz\_top sz\_bot px   
## -2.379882e-02 -2.252678e-01 1.255197e+00 6.340907e-02   
## pz x0 y0 z0   
## 1.607753e+00 -6.459042e-02 NA -1.351931e+00   
## vx0 vy0 vz0 ax   
## 1.399884e-02 -1.025046e+00 -6.470604e-01 3.611276e-02   
## ay az break\_y break\_length   
## -1.394579e-01 -6.398053e-02 -4.295483e+00 3.159281e-01   
## spin\_dir spin\_rate trf\_num\_pitches trf\_num\_AB   
## 1.688166e-03 2.460692e-04 1.804388e-02 NA   
## trf\_num\_AS trf\_num\_CH trf\_num\_CU trf\_num\_EP   
## 1.990627e+01 -6.589889e-03 5.426562e-03 -2.044280e-01   
## trf\_num\_FA trf\_num\_FC trf\_num\_FF trf\_num\_FO   
## -3.212850e-01 4.582021e-04 1.001870e-02 -7.047383e-02   
## trf\_num\_FS trf\_num\_FT trf\_num\_IN trf\_num\_KC   
## 3.050274e-03 2.911716e-03 -7.008995e-02 -2.081670e-03   
## trf\_num\_KN trf\_num\_PO trf\_num\_SC trf\_num\_SI   
## -9.878404e+00 3.237507e-02 -1.321116e+01 -3.657734e-03   
## trf\_num\_SL trf\_num\_UN   
## 8.720584e-03 -7.346978e-03

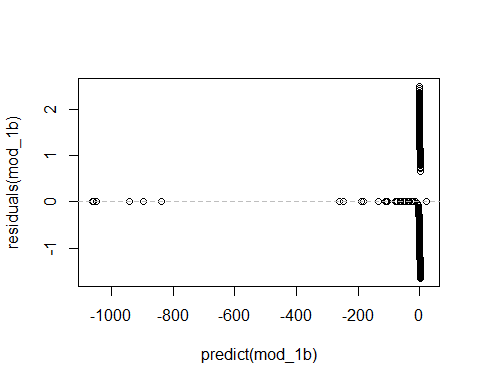
### Odds Ratio

exp(mod\_1b$coefficients);

## (Intercept) x y start\_speed   
## 9.495889e+37 9.809108e-01 1.025440e+00 2.698956e-01   
## end\_speed sz\_top sz\_bot px   
## 9.764821e-01 7.983024e-01 3.508529e+00 1.065463e+00   
## pz x0 y0 z0   
## 4.991581e+00 9.374513e-01 NA 2.587402e-01   
## vx0 vy0 vz0 ax   
## 1.014097e+00 3.587801e-01 5.235827e-01 1.036773e+00   
## ay az break\_y break\_length   
## 8.698297e-01 9.380233e-01 1.362999e-02 1.371532e+00   
## spin\_dir spin\_rate trf\_num\_pitches trf\_num\_AB   
## 1.001690e+00 1.000246e+00 1.018208e+00 NA   
## trf\_num\_AS trf\_num\_CH trf\_num\_CU trf\_num\_EP   
## 4.417553e+08 9.934318e-01 1.005441e+00 8.151135e-01   
## trf\_num\_FA trf\_num\_FC trf\_num\_FF trf\_num\_FO   
## 7.252165e-01 1.000458e+00 1.010069e+00 9.319521e-01   
## trf\_num\_FS trf\_num\_FT trf\_num\_IN trf\_num\_KC   
## 1.003055e+00 1.002916e+00 9.323100e-01 9.979205e-01   
## trf\_num\_KN trf\_num\_PO trf\_num\_SC trf\_num\_SI   
## 5.127003e-05 1.032905e+00 1.830061e-06 9.963489e-01   
## trf\_num\_SL trf\_num\_UN   
## 1.008759e+00 9.926799e-01

### Residual

plot(predict(mod\_1b),residuals(mod\_1b));  
abline(h=0,lty=2,col="grey");



### Performance

pred <- ifelse(predict(mod\_1b, testing\_lessOutliers, type='response') > threshold, 1, 0)  
confusionMatrix(data=pred, reference=testing\_lessOutliers$OnDL, positive='1');

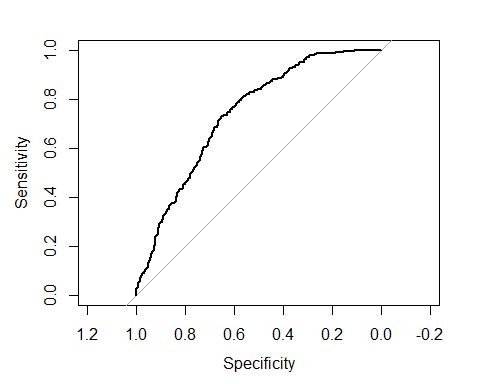
## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction 0 1  
## 0 678 174  
## 1 91 87  
##   
## Accuracy : 0.7427   
## 95% CI : (0.7149, 0.7692)  
## No Information Rate : 0.7466   
## P-Value [Acc > NIR] : 0.6284   
##   
## Kappa : 0.2402   
## Mcnemar's Test P-Value : 4.723e-07   
##   
## Sensitivity : 0.33333   
## Specificity : 0.88166   
## Pos Pred Value : 0.48876   
## Neg Pred Value : 0.79577   
## Prevalence : 0.25340   
## Detection Rate : 0.08447   
## Detection Prevalence : 0.17282   
## Balanced Accuracy : 0.60750   
##   
## 'Positive' Class : 1   
##

### ROC curve

prob <- predict(mod\_1b, testing\_lessOutliers, type='response');  
g1b <- roc(OnDL ~ prob, data = testing\_lessOutliers);  
roc.curve(testing\_lessOutliers$OnDL, prob, plotit =F)

## Area under the curve (AUC): 0.738

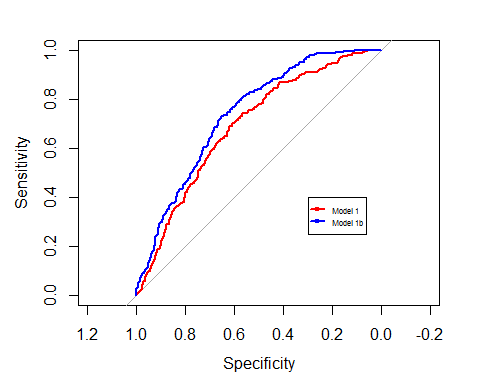
plot(g1b)



### Compare ROC of model with outliers vs model without outliers

The comparison of the ROC curve between the model with outliers (red) and one without outliers (blue) shows that the model has better performance once outliers are removed.

plot(g1, col='red')   
plot(g1b, add=TRUE, col='blue')  
  
legend(0.3,0.4, c("Model 1","Model 1b"), lty=c(1,1), lwd=c(2.5,2.5),col=c("red","blue"), pch=1, cex=0.5);



## Model 2 Only low correlation variables (less than 0.25)

### Low correlation variables

highlyCorDescr <- findCorrelation(m\_lessOutliers, cutoff = .25)  
  
lowCorColNames<- colnames(numeric\_dataset[,-highlyCorDescr]);  
  
print(lowCorColNames);

## [1] "x" "sz\_top" "pz" "x0" "vz0"   
## [6] "break\_y" "trf\_num\_AB" "trf\_num\_CU" "trf\_num\_EP" "trf\_num\_FA"  
## [11] "trf\_num\_FF" "trf\_num\_FO" "trf\_num\_FT" "trf\_num\_IN" "trf\_num\_PO"  
## [16] "trf\_num\_SC" "trf\_num\_UN"

### Construct Model

selected\_variables <- lowCorColNames;   
  
selected\_i <- which(colnames(training\_lessOutliers) %in% selected\_variables);  
   
formula\_text <- paste(response\_var, "~",  
 paste(names(training\_lessOutliers)[selected\_i], collapse="+"));  
formula <- as.formula(formula\_text);  
  
mod\_2 = glm(formula = formula , family=binomial(logit), data=training\_lessOutliers);

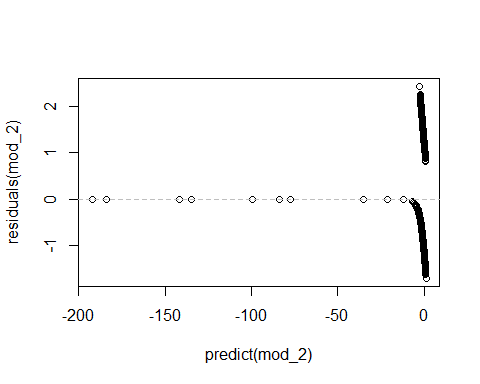
### Summary

summary(mod\_2);

##   
## Call:  
## glm(formula = formula, family = binomial(logit), data = training\_lessOutliers)  
##   
## Deviance Residuals:   
## Min 1Q Median 3Q Max   
## -1.6961 -0.7253 -0.5435 -0.3083 2.4255   
##   
## Coefficients: (1 not defined because of singularities)  
## Estimate Std. Error z value Pr(>|z|)   
## (Intercept) 65.165651 249.595378 0.261 0.79403   
## x 0.001452 0.006082 0.239 0.81131   
## sz\_top 0.720973 0.904091 0.797 0.42519   
## pz -0.467048 0.239939 -1.947 0.05159 .   
## x0 -0.024572 0.027743 -0.886 0.37578   
## vz0 -0.152128 0.032356 -4.702 2.58e-06 \*\*\*  
## break\_y -2.395364 1.647853 -1.454 0.14605   
## trf\_num\_AB NA NA NA NA   
## trf\_num\_CU 0.006271 0.003773 1.662 0.09649 .   
## trf\_num\_EP -0.224243 0.116947 -1.917 0.05518 .   
## trf\_num\_FA -0.283122 0.100639 -2.813 0.00490 \*\*   
## trf\_num\_FF 0.019190 0.003042 6.308 2.82e-10 \*\*\*  
## trf\_num\_FO -0.131495 0.277602 -0.474 0.63573   
## trf\_num\_FT 0.011969 0.002833 4.225 2.39e-05 \*\*\*  
## trf\_num\_IN -0.029048 0.019056 -1.524 0.12742   
## trf\_num\_PO 0.041881 0.051433 0.814 0.41549   
## trf\_num\_SC -9.763217 201.230702 -0.049 0.96130   
## trf\_num\_UN 0.012164 0.003798 3.203 0.00136 \*\*   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## (Dispersion parameter for binomial family taken to be 1)  
##   
## Null deviance: 3345.5 on 3089 degrees of freedom  
## Residual deviance: 3018.0 on 3073 degrees of freedom  
## AIC: 3052  
##   
## Number of Fisher Scoring iterations: 16

### Residual

plot(predict(mod\_2),residuals(mod\_2));  
abline(h=0,lty=2,col="grey");



### COefficients

mod\_2$coefficients;

## (Intercept) x sz\_top pz x0   
## 65.165650877 0.001451894 0.720972800 -0.467047727 -0.024571987   
## vz0 break\_y trf\_num\_AB trf\_num\_CU trf\_num\_EP   
## -0.152127964 -2.395363802 NA 0.006271082 -0.224243237   
## trf\_num\_FA trf\_num\_FF trf\_num\_FO trf\_num\_FT trf\_num\_IN   
## -0.283122269 0.019189752 -0.131495281 0.011968967 -0.029048234   
## trf\_num\_PO trf\_num\_SC trf\_num\_UN   
## 0.041880740 -9.763217477 0.012163559

### Odds ratio

exp(mod\_2$coefficients);

## (Intercept) x sz\_top pz x0   
## 2.000242e+28 1.001453e+00 2.056433e+00 6.268502e-01 9.757274e-01   
## vz0 break\_y trf\_num\_AB trf\_num\_CU trf\_num\_EP   
## 8.588784e-01 9.113952e-02 NA 1.006291e+00 7.991207e-01   
## trf\_num\_FA trf\_num\_FF trf\_num\_FO trf\_num\_FT trf\_num\_IN   
## 7.534277e-01 1.019375e+00 8.767834e-01 1.012041e+00 9.713696e-01   
## trf\_num\_PO trf\_num\_SC trf\_num\_UN   
## 1.042770e+00 5.752923e-05 1.012238e+00

### Performance

pred <- ifelse(predict(mod\_2, testing\_lessOutliers, type='response') > threshold, 1, 0);  
confusionMatrix(data=pred, reference=testing\_lessOutliers$OnDL, positive='1');

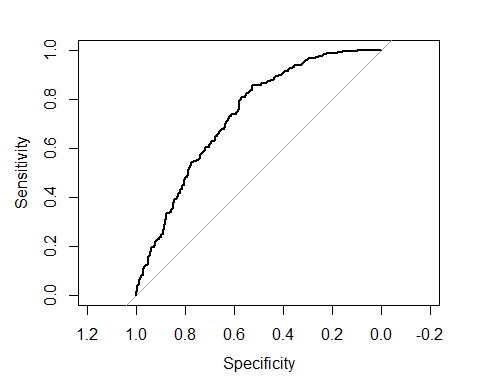
## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction 0 1  
## 0 688 196  
## 1 81 65  
##   
## Accuracy : 0.7311   
## 95% CI : (0.7029, 0.7579)  
## No Information Rate : 0.7466   
## P-Value [Acc > NIR] : 0.881   
##   
## Kappa : 0.1682   
## Mcnemar's Test P-Value : 7.406e-12   
##   
## Sensitivity : 0.24904   
## Specificity : 0.89467   
## Pos Pred Value : 0.44521   
## Neg Pred Value : 0.77828   
## Prevalence : 0.25340   
## Detection Rate : 0.06311   
## Detection Prevalence : 0.14175   
## Balanced Accuracy : 0.57186   
##   
## 'Positive' Class : 1   
##

### ROC curve

prob <- predict(mod\_2, testing\_lessOutliers, type='response');  
g2 <- roc(OnDL ~ prob, data = testing\_lessOutliers);  
roc.curve(testing\_lessOutliers$OnDL, prob, plotit = F);

## Area under the curve (AUC): 0.734

plot(g2)



## Model 2(b) Only low correlation variables (less than 0.5)

### Low correlation variables

highlyCorDescr <- findCorrelation(m\_lessOutliers, cutoff = .5)  
  
lowCorColNames<- colnames(numeric\_dataset[,-highlyCorDescr]);  
  
print(lowCorColNames);

## [1] "sz\_top" "px" "pz" "x0" "z0"   
## [6] "ay" "break\_y" "spin\_rate" "trf\_num\_AB" "trf\_num\_AS"  
## [11] "trf\_num\_CH" "trf\_num\_EP" "trf\_num\_FA" "trf\_num\_FF" "trf\_num\_FO"  
## [16] "trf\_num\_FS" "trf\_num\_FT" "trf\_num\_IN" "trf\_num\_KC" "trf\_num\_KN"  
## [21] "trf\_num\_PO" "trf\_num\_SC" "trf\_num\_SI" "trf\_num\_UN"

### Construct Model

selected\_variables <- lowCorColNames;   
  
selected\_i <- which(colnames(training\_lessOutliers) %in% selected\_variables);  
   
formula\_text <- paste(response\_var, "~",  
 paste(names(training\_lessOutliers)[selected\_i], collapse="+"));  
formula <- as.formula(formula\_text);  
  
mod\_2b = glm(formula = formula , family=binomial(logit), data=training\_lessOutliers);

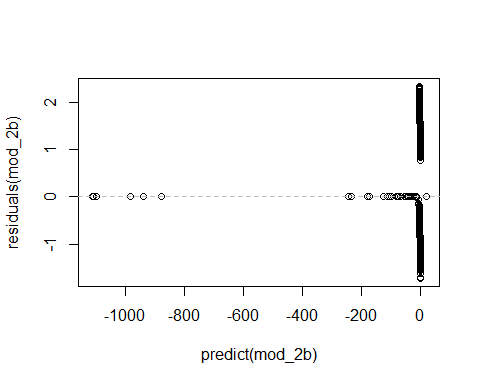
### Summary

summary(mod\_2b);

##   
## Call:  
## glm(formula = formula, family = binomial(logit), data = training\_lessOutliers)  
##   
## Deviance Residuals:   
## Min 1Q Median 3Q Max   
## -1.7172 -0.7251 -0.5341 -0.2341 2.3303   
##   
## Coefficients: (1 not defined because of singularities)  
## Estimate Std. Error z value Pr(>|z|)   
## (Intercept) -2.013e+01 1.941e+04 -0.001 0.999172   
## sz\_top 5.412e-01 8.885e-01 0.609 0.542424   
## px 7.273e-01 2.483e-01 2.929 0.003402 \*\*   
## pz -6.711e-01 2.522e-01 -2.662 0.007778 \*\*   
## x0 -8.171e-03 2.957e-02 -0.276 0.782308   
## z0 2.890e-01 1.168e-01 2.475 0.013319 \*   
## ay 9.663e-02 2.728e-02 3.542 0.000397 \*\*\*  
## break\_y 7.546e-01 1.877e+00 0.402 0.687755   
## spin\_rate 3.942e-05 1.956e-04 0.202 0.840266   
## trf\_num\_AB NA NA NA NA   
## trf\_num\_AS 1.948e+01 1.582e+04 0.001 0.999018   
## trf\_num\_CH -6.219e-03 5.001e-03 -1.244 0.213621   
## trf\_num\_EP -2.006e-01 1.198e-01 -1.675 0.094014 .   
## trf\_num\_FA -2.956e-01 1.036e-01 -2.853 0.004329 \*\*   
## trf\_num\_FF 2.764e-02 3.511e-03 7.873 3.45e-15 \*\*\*  
## trf\_num\_FO -1.471e-01 3.005e-01 -0.490 0.624463   
## trf\_num\_FS 4.882e-03 6.628e-03 0.737 0.461358   
## trf\_num\_FT 2.047e-02 3.412e-03 5.999 1.99e-09 \*\*\*  
## trf\_num\_IN -5.681e-02 1.956e-02 -2.905 0.003670 \*\*   
## trf\_num\_KC -4.111e-03 5.666e-03 -0.725 0.468177   
## trf\_num\_KN -1.036e+01 2.377e+02 -0.044 0.965251   
## trf\_num\_PO 9.465e-03 5.221e-02 0.181 0.856149   
## trf\_num\_SC -1.244e+01 8.730e+02 -0.014 0.988627   
## trf\_num\_SI 1.504e-02 3.443e-03 4.368 1.25e-05 \*\*\*  
## trf\_num\_UN 4.738e-03 3.990e-03 1.188 0.234980   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## (Dispersion parameter for binomial family taken to be 1)  
##   
## Null deviance: 3345.5 on 3089 degrees of freedom  
## Residual deviance: 2976.2 on 3066 degrees of freedom  
## AIC: 3024.2  
##   
## Number of Fisher Scoring iterations: 19

### Residual

plot(predict(mod\_2b),residuals(mod\_2b));  
abline(h=0,lty=2,col="grey");



### COefficients

mod\_2b$coefficients;

## (Intercept) sz\_top px pz x0   
## -2.013386e+01 5.412250e-01 7.272884e-01 -6.711160e-01 -8.170812e-03   
## z0 ay break\_y spin\_rate trf\_num\_AB   
## 2.890383e-01 9.663075e-02 7.545614e-01 3.941815e-05 NA   
## trf\_num\_AS trf\_num\_CH trf\_num\_EP trf\_num\_FA trf\_num\_FF   
## 1.948419e+01 -6.219403e-03 -2.005616e-01 -2.955782e-01 2.764281e-02   
## trf\_num\_FO trf\_num\_FS trf\_num\_FT trf\_num\_IN trf\_num\_KC   
## -1.470926e-01 4.882289e-03 2.046642e-02 -5.681122e-02 -4.110602e-03   
## trf\_num\_KN trf\_num\_PO trf\_num\_SC trf\_num\_SI trf\_num\_UN   
## -1.035637e+01 9.464679e-03 -1.244472e+01 1.503747e-02 4.738141e-03

### Odds ratio

exp(mod\_2b$coefficients);

## (Intercept) sz\_top px pz x0   
## 1.802916e-09 1.718110e+00 2.069461e+00 5.111378e-01 9.918625e-01   
## z0 ay break\_y spin\_rate trf\_num\_AB   
## 1.335143e+00 1.101454e+00 2.126679e+00 1.000039e+00 NA   
## trf\_num\_AS trf\_num\_CH trf\_num\_EP trf\_num\_FA trf\_num\_FF   
## 2.896521e+08 9.937999e-01 8.182710e-01 7.441013e-01 1.028028e+00   
## trf\_num\_FO trf\_num\_FS trf\_num\_FT trf\_num\_IN trf\_num\_KC   
## 8.632140e-01 1.004894e+00 1.020677e+00 9.447724e-01 9.958978e-01   
## trf\_num\_KN trf\_num\_PO trf\_num\_SC trf\_num\_SI trf\_num\_UN   
## 3.178965e-05 1.009510e+00 3.938474e-06 1.015151e+00 1.004749e+00

### Performance

pred <- ifelse(predict(mod\_2b, testing\_lessOutliers, type='response') > threshold, 1, 0);  
confusionMatrix(data=pred, reference=testing\_lessOutliers$OnDL, positive='1');

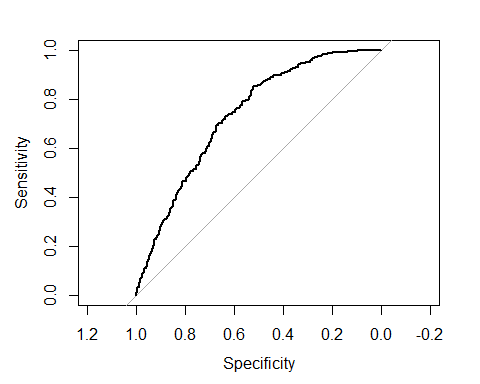
## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction 0 1  
## 0 688 184  
## 1 81 77  
##   
## Accuracy : 0.7427   
## 95% CI : (0.7149, 0.7692)  
## No Information Rate : 0.7466   
## P-Value [Acc > NIR] : 0.6284   
##   
## Kappa : 0.2181   
## Mcnemar's Test P-Value : 3.709e-10   
##   
## Sensitivity : 0.29502   
## Specificity : 0.89467   
## Pos Pred Value : 0.48734   
## Neg Pred Value : 0.78899   
## Prevalence : 0.25340   
## Detection Rate : 0.07476   
## Detection Prevalence : 0.15340   
## Balanced Accuracy : 0.59484   
##   
## 'Positive' Class : 1   
##

### ROC curve

prob <- predict(mod\_2b, testing\_lessOutliers, type='response');  
g2b <- roc(OnDL ~ prob, data = testing\_lessOutliers);  
roc.curve(testing\_lessOutliers$OnDL, prob, plotit = F);

## Area under the curve (AUC): 0.735

plot(g2b)



## Model 2(c) Only low correlation variables (less than 0.75)

### Low correlation variables

highlyCorDescr <- findCorrelation(m\_lessOutliers, cutoff = .75);  
  
lowCorColNames <- colnames(numeric\_dataset\_lessOutliers[,-highlyCorDescr]);  
  
  
#filter\_pitches\_dl\_dataset <- filteredDescr[,-highlyCorDescr]  
#all\_variables <- colnames(numeric\_dataset);  
print(lowCorColNames);

## [1] "x" "y" "sz\_top" "sz\_bot" "px"   
## [6] "pz" "z0" "vx0" "ay" "az"   
## [11] "break\_y" "spin\_rate" "trf\_num\_AS" "trf\_num\_CH" "trf\_num\_CU"  
## [16] "trf\_num\_EP" "trf\_num\_FA" "trf\_num\_FC" "trf\_num\_FF" "trf\_num\_FO"  
## [21] "trf\_num\_FS" "trf\_num\_FT" "trf\_num\_IN" "trf\_num\_KC" "trf\_num\_KN"  
## [26] "trf\_num\_PO" "trf\_num\_SC" "trf\_num\_SI" "trf\_num\_SL"

### Construct Model

selected\_variables <- lowCorColNames;   
  
selected\_i <- which(colnames(training) %in% selected\_variables);  
   
formula\_text <- paste(response\_var, "~",  
 paste(names(training\_lessOutliers)[selected\_i], collapse="+"));  
formula <- as.formula(formula\_text);  
  
mod\_2c = glm(formula = formula , family=binomial(logit), data=training\_lessOutliers);

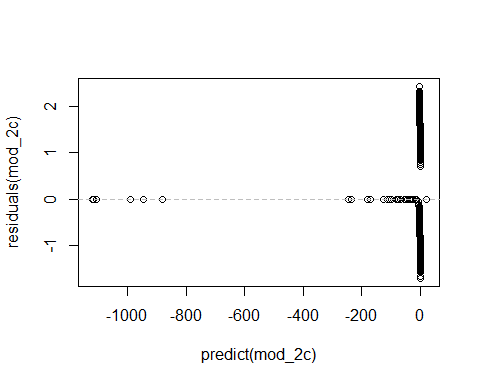
### Summary

summary(mod\_2c);

##   
## Call:  
## glm(formula = formula, family = binomial(logit), data = training\_lessOutliers)  
##   
## Deviance Residuals:   
## Min 1Q Median 3Q Max   
## -1.7091 -0.7280 -0.5256 -0.2240 2.4339   
##   
## Coefficients:  
## Estimate Std. Error z value Pr(>|z|)   
## (Intercept) -2.813e+01 1.941e+04 -0.001 0.998844   
## x -2.709e-02 2.024e-02 -1.338 0.180755   
## y 3.107e-02 1.307e-02 2.377 0.017431 \*   
## sz\_top -2.171e-01 8.399e-01 -0.259 0.796017   
## sz\_bot 2.763e-01 1.050e+00 0.263 0.792530   
## px -7.305e-02 6.004e-01 -0.122 0.903156   
## pz 6.976e-02 3.797e-01 0.184 0.854236   
## z0 3.181e-01 1.280e-01 2.485 0.012949 \*   
## vx0 -1.930e-04 9.099e-03 -0.021 0.983078   
## ay 1.082e-01 2.900e-02 3.731 0.000191 \*\*\*  
## az -1.993e-02 1.916e-02 -1.040 0.298275   
## break\_y 9.775e-01 1.908e+00 0.512 0.608425   
## spin\_rate 2.693e-04 2.436e-04 1.105 0.268963   
## trf\_num\_AS 1.949e+01 1.582e+04 0.001 0.999017   
## trf\_num\_CH -6.668e-03 5.034e-03 -1.325 0.185325   
## trf\_num\_CU 2.059e-03 4.711e-03 0.437 0.662108   
## trf\_num\_EP -2.198e-01 1.236e-01 -1.778 0.075402 .   
## trf\_num\_FA -3.088e-01 1.046e-01 -2.952 0.003162 \*\*   
## trf\_num\_FC 8.043e-03 4.013e-03 2.004 0.045063 \*   
## trf\_num\_FF 2.571e-02 4.227e-03 6.081 1.19e-09 \*\*\*  
## trf\_num\_FO -1.258e-01 2.958e-01 -0.425 0.670727   
## trf\_num\_FS 4.370e-03 6.745e-03 0.648 0.517059   
## trf\_num\_FT 1.729e-02 3.812e-03 4.535 5.77e-06 \*\*\*  
## trf\_num\_IN -6.094e-02 1.972e-02 -3.090 0.002003 \*\*   
## trf\_num\_KC -3.521e-03 6.457e-03 -0.545 0.585570   
## trf\_num\_KN -1.041e+01 2.354e+02 -0.044 0.964725   
## trf\_num\_PO 1.520e-02 5.289e-02 0.287 0.773757   
## trf\_num\_SC -1.247e+01 8.749e+02 -0.014 0.988629   
## trf\_num\_SI 1.128e-02 4.006e-03 2.815 0.004878 \*\*   
## trf\_num\_SL 7.549e-03 4.098e-03 1.842 0.065428 .   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## (Dispersion parameter for binomial family taken to be 1)  
##   
## Null deviance: 3345.5 on 3089 degrees of freedom  
## Residual deviance: 2959.9 on 3060 degrees of freedom  
## AIC: 3019.9  
##   
## Number of Fisher Scoring iterations: 19

### Residual

plot(predict(mod\_2c),residuals(mod\_2c));  
abline(h=0,lty=2,col="grey");



### COefficients

mod\_2c$coefficients;

## (Intercept) x y sz\_top sz\_bot   
## -2.813061e+01 -2.708802e-02 3.106750e-02 -2.171255e-01 2.762835e-01   
## px pz z0 vx0 ay   
## -7.304863e-02 6.976023e-02 3.181179e-01 -1.929885e-04 1.081875e-01   
## az break\_y spin\_rate trf\_num\_AS trf\_num\_CH   
## -1.993275e-02 9.774508e-01 2.692993e-04 1.948753e+01 -6.667687e-03   
## trf\_num\_CU trf\_num\_EP trf\_num\_FA trf\_num\_FC trf\_num\_FF   
## 2.058678e-03 -2.198445e-01 -3.087691e-01 8.043021e-03 2.570561e-02   
## trf\_num\_FO trf\_num\_FS trf\_num\_FT trf\_num\_IN trf\_num\_KC   
## -1.257593e-01 4.369897e-03 1.728698e-02 -6.093793e-02 -3.521058e-03   
## trf\_num\_KN trf\_num\_PO trf\_num\_SC trf\_num\_SI trf\_num\_SL   
## -1.041132e+01 1.520469e-02 -1.246918e+01 1.127814e-02 7.549246e-03

### Performance

pred <- ifelse(predict(mod\_2c, testing\_lessOutliers, type='response') > threshold, 1, 0);  
confusionMatrix(data=pred, reference=testing\_lessOutliers$OnDL, positive='1');

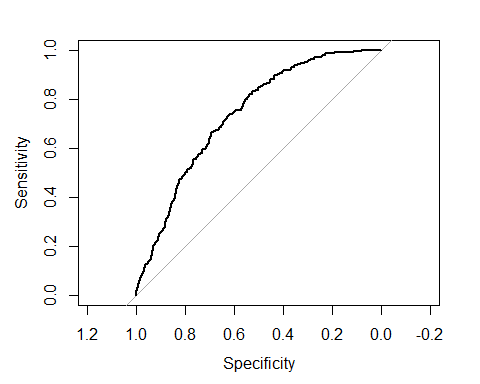
## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction 0 1  
## 0 680 189  
## 1 89 72  
##   
## Accuracy : 0.7301   
## 95% CI : (0.7019, 0.757)  
## No Information Rate : 0.7466   
## P-Value [Acc > NIR] : 0.8944   
##   
## Kappa : 0.1833   
## Mcnemar's Test P-Value : 2.892e-09   
##   
## Sensitivity : 0.2759   
## Specificity : 0.8843   
## Pos Pred Value : 0.4472   
## Neg Pred Value : 0.7825   
## Prevalence : 0.2534   
## Detection Rate : 0.0699   
## Detection Prevalence : 0.1563   
## Balanced Accuracy : 0.5801   
##   
## 'Positive' Class : 1   
##

### ROC curve

prob <- predict(mod\_2c, testing\_lessOutliers, type='response');  
g2c <- roc(OnDL ~ prob, data = testing\_lessOutliers);  
roc.curve(testing\_lessOutliers$OnDL, prob, plotit = F)

## Area under the curve (AUC): 0.736

plot(g2c)



## Model 2(d) Only low correlation variables (less than 0.85)

### Low correlation variables

highlyCorDescr <- findCorrelation(m\_lessOutliers, cutoff =0.85);  
  
lowCorColNames <- colnames(numeric\_dataset\_lessOutliers[,-highlyCorDescr]);  
  
  
#filter\_pitches\_dl\_dataset <- filteredDescr[,-highlyCorDescr]  
#all\_variables <- colnames(numeric\_dataset);  
print(lowCorColNames);

## [1] "x" "y" "end\_speed"   
## [4] "sz\_top" "sz\_bot" "px"   
## [7] "pz" "z0" "vx0"   
## [10] "vz0" "ay" "az"   
## [13] "break\_y" "spin\_rate" "trf\_num\_pitches"  
## [16] "trf\_num\_AS" "trf\_num\_CH" "trf\_num\_CU"   
## [19] "trf\_num\_EP" "trf\_num\_FA" "trf\_num\_FC"   
## [22] "trf\_num\_FF" "trf\_num\_FO" "trf\_num\_FS"   
## [25] "trf\_num\_FT" "trf\_num\_IN" "trf\_num\_KC"   
## [28] "trf\_num\_KN" "trf\_num\_PO" "trf\_num\_SC"   
## [31] "trf\_num\_SI" "trf\_num\_SL"

### Construct Model

selected\_variables <- lowCorColNames;   
  
selected\_i <- which(colnames(training) %in% selected\_variables);  
   
formula\_text <- paste(response\_var, "~",  
 paste(names(training\_lessOutliers)[selected\_i], collapse="+"));  
formula <- as.formula(formula\_text);  
  
mod\_2d = glm(formula = formula , family=binomial(logit), data=training\_lessOutliers);

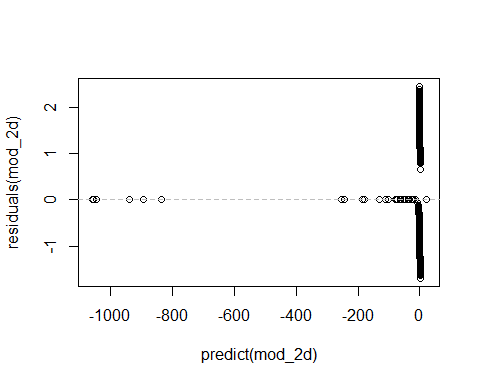
### Summary

summary(mod\_2d);

##   
## Call:  
## glm(formula = formula, family = binomial(logit), data = training\_lessOutliers)  
##   
## Deviance Residuals:   
## Min 1Q Median 3Q Max   
## -1.6953 -0.7265 -0.5076 -0.1613 2.4513   
##   
## Coefficients:  
## Estimate Std. Error z value Pr(>|z|)   
## (Intercept) 8.086e+01 1.941e+04 0.004 0.996676   
## x -2.148e-02 2.028e-02 -1.059 0.289458   
## y 2.628e-02 1.308e-02 2.008 0.044595 \*   
## end\_speed 1.271e-01 3.646e-02 3.485 0.000492 \*\*\*  
## sz\_top -2.695e-01 1.062e+00 -0.254 0.799664   
## sz\_bot 1.269e+00 1.088e+00 1.166 0.243416   
## px 8.385e-02 5.972e-01 0.140 0.888343   
## pz 1.437e+00 5.271e-01 2.726 0.006401 \*\*   
## z0 -1.137e+00 4.344e-01 -2.616 0.008885 \*\*   
## vx0 -1.161e-02 9.446e-03 -1.229 0.219073   
## vz0 -5.541e-01 1.569e-01 -3.532 0.000412 \*\*\*  
## ay -8.587e-02 4.104e-02 -2.092 0.036427 \*   
## az -1.236e-01 3.085e-02 -4.005 6.2e-05 \*\*\*  
## break\_y -3.890e+00 2.094e+00 -1.857 0.063249 .   
## spin\_rate 3.544e-04 2.459e-04 1.441 0.149553   
## trf\_num\_pitches 1.397e-02 9.447e-03 1.479 0.139142   
## trf\_num\_AS 1.998e+01 1.582e+04 0.001 0.998993   
## trf\_num\_CH -5.264e-03 6.314e-03 -0.834 0.404403   
## trf\_num\_CU 5.516e-03 5.821e-03 0.948 0.343335   
## trf\_num\_EP -2.093e-01 1.241e-01 -1.687 0.091592 .   
## trf\_num\_FA -3.115e-01 1.056e-01 -2.949 0.003185 \*\*   
## trf\_num\_FC 2.916e-03 5.205e-03 0.560 0.575280   
## trf\_num\_FF 1.153e-02 7.379e-03 1.563 0.118078   
## trf\_num\_FO -5.915e-02 2.892e-01 -0.205 0.837960   
## trf\_num\_FS 3.038e-03 7.642e-03 0.398 0.690997   
## trf\_num\_FT 5.631e-03 5.361e-03 1.050 0.293493   
## trf\_num\_IN -6.793e-02 2.034e-02 -3.340 0.000838 \*\*\*  
## trf\_num\_KC -9.364e-04 7.354e-03 -0.127 0.898675   
## trf\_num\_KN -9.848e+00 2.302e+02 -0.043 0.965874   
## trf\_num\_PO 2.881e-02 5.343e-02 0.539 0.589717   
## trf\_num\_SC -1.291e+01 8.653e+02 -0.015 0.988094   
## trf\_num\_SI -1.352e-03 6.268e-03 -0.216 0.829158   
## trf\_num\_SL 4.945e-03 5.532e-03 0.894 0.371385   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## (Dispersion parameter for binomial family taken to be 1)  
##   
## Null deviance: 3345.5 on 3089 degrees of freedom  
## Residual deviance: 2910.1 on 3057 degrees of freedom  
## AIC: 2976.1  
##   
## Number of Fisher Scoring iterations: 19

### Residual

plot(predict(mod\_2d),residuals(mod\_2d));  
abline(h=0,lty=2,col="grey");



### COefficients

exp(mod\_2d$coefficients);

## (Intercept) x y end\_speed   
## 1.309692e+35 9.787465e-01 1.026627e+00 1.135513e+00   
## sz\_top sz\_bot px pz   
## 7.637463e-01 3.558145e+00 1.087466e+00 4.208748e+00   
## z0 vx0 vz0 ay   
## 3.209218e-01 9.884575e-01 5.745706e-01 9.177123e-01   
## az break\_y spin\_rate trf\_num\_pitches   
## 8.837658e-01 2.044149e-02 1.000354e+00 1.014070e+00   
## trf\_num\_AS trf\_num\_CH trf\_num\_CU trf\_num\_EP   
## 4.738614e+08 9.947494e-01 1.005531e+00 8.111271e-01   
## trf\_num\_FA trf\_num\_FC trf\_num\_FF trf\_num\_FO   
## 7.323249e-01 1.002920e+00 1.011600e+00 9.425663e-01   
## trf\_num\_FS trf\_num\_FT trf\_num\_IN trf\_num\_KC   
## 1.003043e+00 1.005647e+00 9.343268e-01 9.990641e-01   
## trf\_num\_KN trf\_num\_PO trf\_num\_SC trf\_num\_SI   
## 5.284055e-05 1.029233e+00 2.464992e-06 9.986484e-01   
## trf\_num\_SL   
## 1.004957e+00

### Performance

pred <- ifelse(predict(mod\_2d, testing\_lessOutliers, type='response') > threshold, 1, 0);  
confusionMatrix(data=pred, reference=testing\_lessOutliers$OnDL, positive='1');

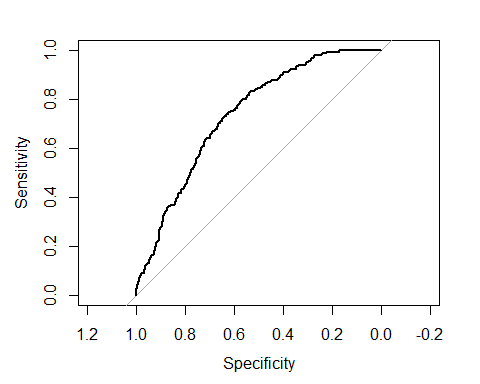
## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction 0 1  
## 0 677 174  
## 1 92 87  
##   
## Accuracy : 0.7417   
## 95% CI : (0.7139, 0.7682)  
## No Information Rate : 0.7466   
## P-Value [Acc > NIR] : 0.655   
##   
## Kappa : 0.2384   
## Mcnemar's Test P-Value : 6.82e-07   
##   
## Sensitivity : 0.33333   
## Specificity : 0.88036   
## Pos Pred Value : 0.48603   
## Neg Pred Value : 0.79553   
## Prevalence : 0.25340   
## Detection Rate : 0.08447   
## Detection Prevalence : 0.17379   
## Balanced Accuracy : 0.60685   
##   
## 'Positive' Class : 1   
##

### ROC curve

prob <- predict(mod\_2d, testing\_lessOutliers, type='response');  
g2d <- roc(OnDL ~ prob, data = testing\_lessOutliers);  
roc.curve(testing\_lessOutliers$OnDL, prob, plotit = F)

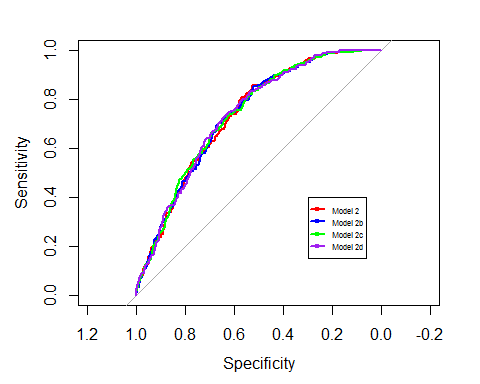
## Area under the curve (AUC): 0.736

plot(g2d)



## Compare ROC Curve of model by correlation

plot(g2, col='red'); #cutoff 0.25  
plot(g2b, add=TRUE, col='blue') #cutoff 0.5  
plot(g2c, add=TRUE, col='green') #cutoff 0.75  
plot(g2d, add=TRUE, col='purple') #cutoff 0.85  
legend(0.3,0.4, c("Model 2","Model 2b","Model 2c", "Model 2d"), lty=c(1,1), lwd=c(2.5,2.5),col=c("red","blue","green", "purple"), pch=1, cex=0.5);



## Model 3 Original continuous variables

### Construct Model

selected\_variables <- original\_var;  
  
selected\_i <- which(colnames(training\_lessOutliers) %in% selected\_variables);  
  
formula\_text <- paste(response\_var, "~",  
 paste(names(training)[selected\_i], collapse="+"));  
formula <- as.formula(formula\_text);  
  
mod\_3 = glm(formula = formula , family=binomial(logit), data=training\_lessOutliers);

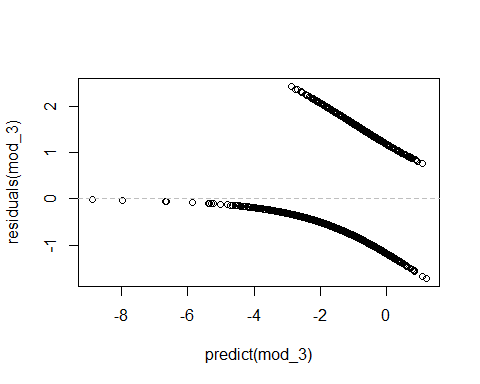
### Summary

summary(mod\_3);

##   
## Call:  
## glm(formula = formula, family = binomial(logit), data = training\_lessOutliers)  
##   
## Deviance Residuals:   
## Min 1Q Median 3Q Max   
## -1.7124 -0.7388 -0.5148 -0.2044 2.4239   
##   
## Coefficients: (1 not defined because of singularities)  
## Estimate Std. Error z value Pr(>|z|)   
## (Intercept) 8.951e+01 4.913e+01 1.822 0.06846 .   
## x -2.057e-02 2.072e-02 -0.992 0.32097   
## y 2.459e-02 1.340e-02 1.834 0.06660 .   
## start\_speed -1.166e+00 6.382e-01 -1.827 0.06765 .   
## end\_speed -6.360e-03 1.722e-01 -0.037 0.97054   
## sz\_top -3.317e-01 1.036e+00 -0.320 0.74880   
## sz\_bot 1.298e+00 1.077e+00 1.205 0.22827   
## px 1.987e-01 6.938e-01 0.286 0.77463   
## pz 1.652e+00 5.323e-01 3.103 0.00192 \*\*   
## x0 -2.991e-01 3.855e-01 -0.776 0.43785   
## y0 NA NA NA NA   
## z0 -1.370e+00 4.224e-01 -3.243 0.00118 \*\*   
## vx0 -7.884e-02 1.518e-01 -0.519 0.60356   
## vy0 -9.135e-01 4.475e-01 -2.041 0.04120 \*   
## vz0 -6.350e-01 1.575e-01 -4.031 5.54e-05 \*\*\*  
## ax 6.168e-03 3.448e-02 0.179 0.85803   
## ay -1.472e-01 5.504e-02 -2.674 0.00749 \*\*   
## az -5.634e-02 7.168e-02 -0.786 0.43180   
## break\_y -4.462e+00 2.074e+00 -2.151 0.03147 \*   
## break\_length 2.084e-01 2.174e-01 0.959 0.33760   
## spin\_dir -2.112e-03 5.579e-03 -0.379 0.70501   
## spin\_rate 7.839e-05 3.400e-04 0.231 0.81766   
## trf\_num\_pitches 2.083e-02 1.542e-03 13.507 < 2e-16 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## (Dispersion parameter for binomial family taken to be 1)  
##   
## Null deviance: 3345.5 on 3089 degrees of freedom  
## Residual deviance: 2954.9 on 3068 degrees of freedom  
## AIC: 2998.9  
##   
## Number of Fisher Scoring iterations: 5

### Residual

plot(predict(mod\_3),residuals(mod\_3));  
abline(h=0,lty=2,col="grey");



### COefficients

mod\_3$coefficients;

## (Intercept) x y start\_speed   
## 8.951037e+01 -2.056870e-02 2.458839e-02 -1.166118e+00   
## end\_speed sz\_top sz\_bot px   
## -6.360043e-03 -3.317224e-01 1.298170e+00 1.986510e-01   
## pz x0 y0 z0   
## 1.651737e+00 -2.990770e-01 NA -1.369666e+00   
## vx0 vy0 vz0 ax   
## -7.884158e-02 -9.135098e-01 -6.349508e-01 6.167722e-03   
## ay az break\_y break\_length   
## -1.471963e-01 -5.634498e-02 -4.461766e+00 2.084452e-01   
## spin\_dir spin\_rate trf\_num\_pitches   
## -2.111879e-03 7.839036e-05 2.083002e-02

### Odds Raio

exp(mod\_3$coefficients);

## (Intercept) x y start\_speed   
## 7.479279e+38 9.796414e-01 1.024893e+00 3.115742e-01   
## end\_speed sz\_top sz\_bot px   
## 9.936601e-01 7.176866e-01 3.662588e+00 1.219756e+00   
## pz x0 y0 z0   
## 5.216030e+00 7.415023e-01 NA 2.541918e-01   
## vx0 vy0 vz0 ax   
## 9.241863e-01 4.011139e-01 5.299615e-01 1.006187e+00   
## ay az break\_y break\_length   
## 8.631246e-01 9.452130e-01 1.154196e-02 1.231761e+00   
## spin\_dir spin\_rate trf\_num\_pitches   
## 9.978903e-01 1.000078e+00 1.021048e+00

### Performance

pred <- ifelse(predict(mod\_3, testing\_lessOutliers, type='response') > threshold, 1, 0);  
confusionMatrix(data=pred, reference=testing\_lessOutliers$OnDL, positive='1');

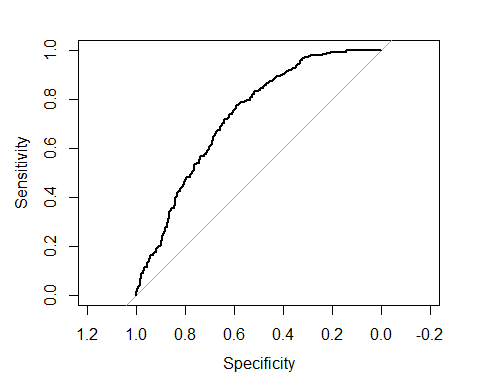
## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction 0 1  
## 0 668 180  
## 1 101 81  
##   
## Accuracy : 0.7272   
## 95% CI : (0.6989, 0.7542)  
## No Information Rate : 0.7466   
## P-Value [Acc > NIR] : 0.9281   
##   
## Kappa : 0.1989   
## Mcnemar's Test P-Value : 3.27e-06   
##   
## Sensitivity : 0.31034   
## Specificity : 0.86866   
## Pos Pred Value : 0.44505   
## Neg Pred Value : 0.78774   
## Prevalence : 0.25340   
## Detection Rate : 0.07864   
## Detection Prevalence : 0.17670   
## Balanced Accuracy : 0.58950   
##   
## 'Positive' Class : 1   
##

### ROC curve

prob <- predict(mod\_3, testing\_lessOutliers, type='response');  
g3 <- roc(OnDL ~ prob, data = testing\_lessOutliers);  
roc.curve(testing\_lessOutliers$OnDL, prob, plotit = F)

## Area under the curve (AUC): 0.729

plot(g3)



## Model 3(b) Siginificant Continuous Variables + num pitches

### Construct Model

selected\_variables <- c("trf\_num\_pitches", "start\_speed", "vy0", "vz0", "break\_y" );  
  
selected\_i <- which(colnames(training\_lessOutliers) %in% selected\_variables);  
  
formula\_text <- paste(response\_var, "~",  
 paste(names(training)[selected\_i], collapse="+"));  
formula <- as.formula(formula\_text);  
  
mod\_3b = glm(formula = formula , family=binomial(logit), data=training\_lessOutliers);

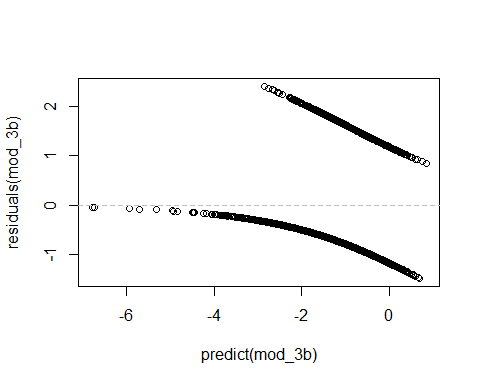
### Summary

summary(mod\_3b);

##   
## Call:  
## glm(formula = formula, family = binomial(logit), data = training\_lessOutliers)  
##   
## Deviance Residuals:   
## Min 1Q Median 3Q Max   
## -1.4861 -0.7449 -0.5293 -0.2556 2.4066   
##   
## Coefficients:  
## Estimate Std. Error z value Pr(>|z|)   
## (Intercept) 5.580861 39.454286 0.141 0.88751   
## start\_speed -0.609483 0.579532 -1.052 0.29295   
## vy0 -0.486067 0.397291 -1.223 0.22116   
## vz0 -0.110262 0.034178 -3.226 0.00125 \*\*   
## break\_y -0.746877 1.650973 -0.452 0.65099   
## trf\_num\_pitches 0.021762 0.001471 14.793 < 2e-16 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## (Dispersion parameter for binomial family taken to be 1)  
##   
## Null deviance: 3345.5 on 3089 degrees of freedom  
## Residual deviance: 3005.3 on 3084 degrees of freedom  
## AIC: 3017.3  
##   
## Number of Fisher Scoring iterations: 5

### Residual

plot(predict(mod\_3b),residuals(mod\_3b));  
abline(h=0,lty=2,col="grey");



### COefficients

mod\_3b$coefficients

## (Intercept) start\_speed vy0 vz0   
## 5.58086102 -0.60948285 -0.48606728 -0.11026238   
## break\_y trf\_num\_pitches   
## -0.74687689 0.02176217

exp(mod\_3b$coefficients);

## (Intercept) start\_speed vy0 vz0   
## 265.2999366 0.5436319 0.6150404 0.8955991   
## break\_y trf\_num\_pitches   
## 0.4738441 1.0220007

### Performance

pred <- ifelse(predict(mod\_3b, testing\_lessOutliers, type='response') > threshold, 1, 0);  
confusionMatrix(data=pred, reference=testing\_lessOutliers$OnDL, positive='1');

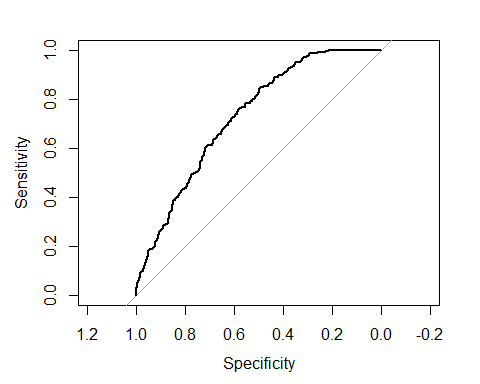
## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction 0 1  
## 0 678 187  
## 1 91 74  
##   
## Accuracy : 0.7301   
## 95% CI : (0.7019, 0.757)  
## No Information Rate : 0.7466   
## P-Value [Acc > NIR] : 0.8944   
##   
## Kappa : 0.188   
## Mcnemar's Test P-Value : 1.214e-08   
##   
## Sensitivity : 0.28352   
## Specificity : 0.88166   
## Pos Pred Value : 0.44848   
## Neg Pred Value : 0.78382   
## Prevalence : 0.25340   
## Detection Rate : 0.07184   
## Detection Prevalence : 0.16019   
## Balanced Accuracy : 0.58259   
##   
## 'Positive' Class : 1   
##

### ROC curve

prob <- predict(mod\_3b, testing\_lessOutliers, type='response');  
g3b <- roc(OnDL ~ prob, data = testing\_lessOutliers);  
roc.curve(testing\_lessOutliers$OnDL, prob, plotit = F)

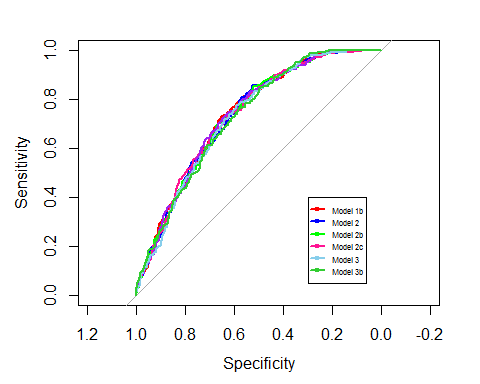
## Area under the curve (AUC): 0.729

plot(g3b)



### Compare all models by ROC curve

plot(g1b, col='red');  
plot(g2, col='blue', add=TRUE);  
plot(g2b, col='green', add=TRUE);  
plot(g2c, col='deeppink', add=TRUE);  
plot(g2d, col='purple', add=TRUE);  
plot(g3, col='skyblue', add=TRUE);  
plot(g3b, col='limegreen', add=TRUE);  
  
legend(0.3,0.4, c("Model 1b","Model 2","Model 2b","Model 2c","Model 3","Model 3b"), lty=c(1,1), lwd=c(2.5,2.5),col=c("red","blue","green","deeppink","skyblue","limegreen"), pch=1, cex=0.5);



# Cross Validation on significant variables

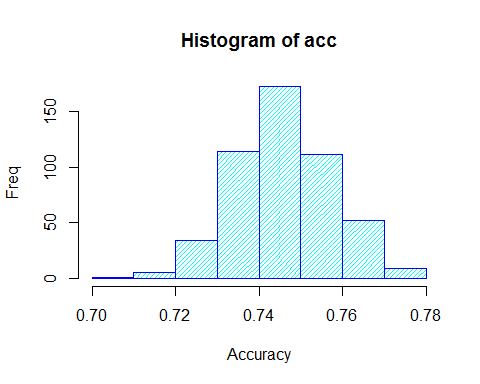
# highlyCorDescr <- findCorrelation(m\_lessOutliers, cutoff = .85);  
#   
# lowCorColNames <- colnames(numeric\_dataset\_lessOutliers[,-highlyCorDescr]);  
#   
#   
# #filter\_pitches\_dl\_dataset <- filteredDescr[,-highlyCorDescr]  
# #all\_variables <- colnames(numeric\_dataset);  
# print(lowCorColNames);  
  
selected\_variables <- c("y","z0", "ay", "trf\_num\_FA", "trf\_num\_FC", "trf\_num\_FF", "trf\_num\_FT", "trf\_num\_IN", "trf\_num\_SI");   
  
selected\_i <- which(colnames(training) %in% selected\_variables);  
   
formula\_text <- paste(response\_var, "~",  
 paste(names(training\_lessOutliers)[selected\_i], collapse="+"));  
formula <- as.formula(formula\_text);  
  
  
# False positive rate  
fpr <- NULL  
  
# False negative rate  
fnr <- NULL  
  
# True positive rate  
tpr <- NULL  
  
# True negative rate  
tnr <- NULL  
  
auc <- NULL  
  
# Number of iterations  
k <- 500  
  
# # Initialize progress bar  
# pbar <- create\_progress\_bar('text')  
# pbar$init(k)  
  
# Accuracy  
acc <- NULL  
  
set.seed(123)  
  
  
for(i in 1:k)  
{  
 # Train-test splitting  
 # 95% of samples -> fitting  
 # 5% of samples -> testing  
 smp\_size <- floor(0.75 \* nrow(model\_dataset\_lessOutliers))  
 index <- sample(seq\_len(nrow(model\_dataset\_lessOutliers)),size=smp\_size)  
 train <- model\_dataset\_lessOutliers[index, ]  
 test <- model\_dataset\_lessOutliers[-index, ]  
   
 # Fitting  
 model <- glm(formula=formula,family=binomial,data=model\_dataset\_lessOutliers)  
   
   
 # Predict results  
 results\_prob <- predict(model,test,type='response')  
   
 # If prob > 0.4 then 1, else 0  
 results <- ifelse(results\_prob > 0.4,1,0)  
   
 # Actual answers  
 answers <- test$OnDL;  
   
 # Accuracy calculation  
 misClasificError <- mean(answers != results)  
   
 # Collecting results  
 acc[i] <- 1-misClasificError  
   
 # Confusion matrix  
 cm <- confusionMatrix(data=results, reference=answers, positive='1')  
 tnr[i] <- cm$table[1]/(cm$table[1]+cm$table[2])  
 tpr[i] <- cm$table[4]/(cm$table[3]+cm$table[4])  
 fpr[i] <- cm$table[2]/(cm$table[1]+cm$table[2])  
 fnr[i] <- cm$table[3]/(cm$table[3]+cm$table[4])  
 auc[i] <- roc.curve(test$OnDL, results\_prob, plotit = F)$auc  
   
 # pbar$step()  
}  
  
# Average accuracy of the model  
mean(acc)

## [1] 0.7458835

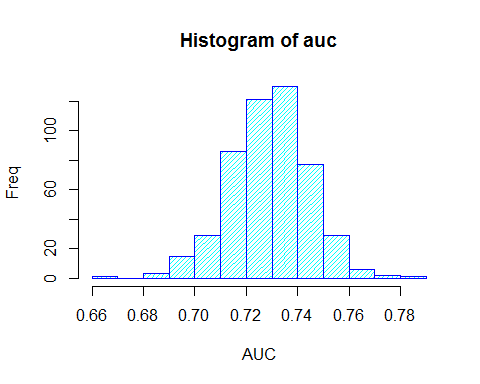
# Average auc of the model  
mean(auc)

## [1] 0.7289232

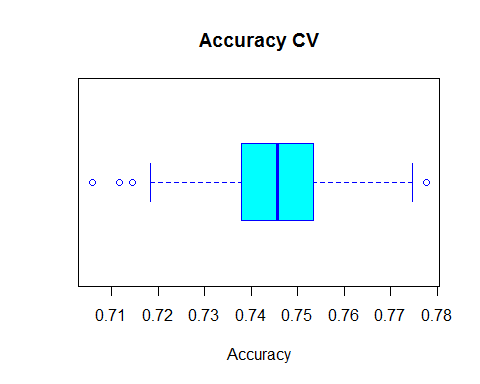
# Histogram of accuracy  
hist(acc,xlab='Accuracy',ylab='Freq', col='cyan',border='blue',density=30)



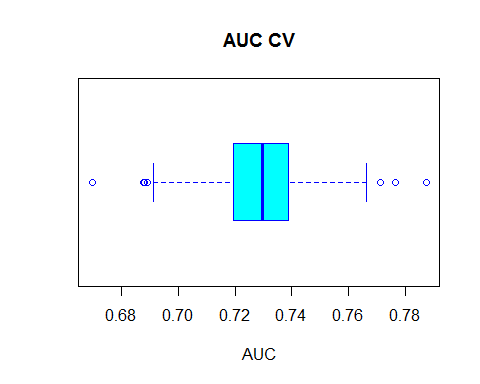
# Histogram of auc  
hist(auc,xlab='AUC',ylab='Freq', col='cyan',border='blue',density=30)



# Boxplot of accuracy  
boxplot(acc,col='cyan',border='blue',horizontal=T,xlab='Accuracy', main='Accuracy CV')



# Boxplot of auc  
boxplot(auc,col='cyan',border='blue',horizontal=T,xlab='AUC', main='AUC CV')



# Confusion matrix and plots of fpr and fnr  
  
mean(fpr)

## [1] 0.1087389

mean(fnr)

## [1] 0.7206481

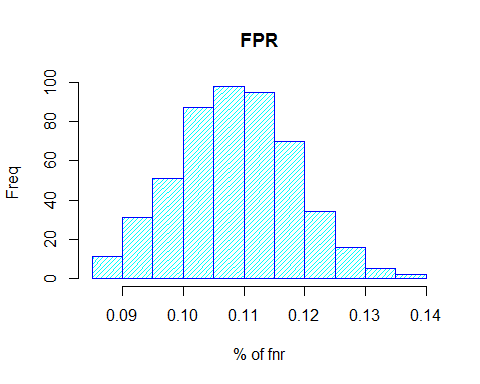
mean(tpr)

## [1] 0.2793519

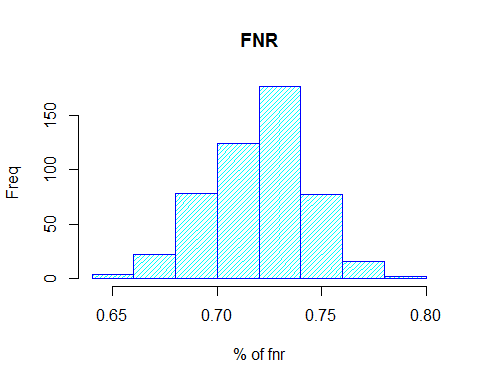
mean(tnr)

## [1] 0.8912611

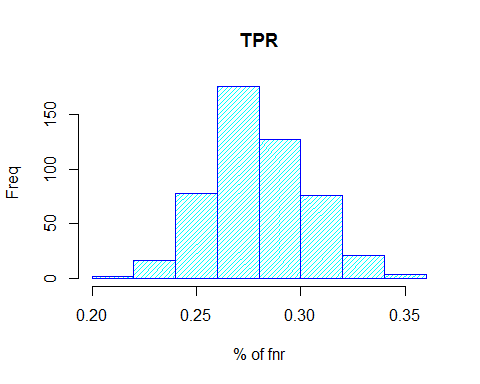
hist(fpr,xlab='% of fnr',ylab='Freq',main='FPR',  
 col='cyan',border='blue',density=30)



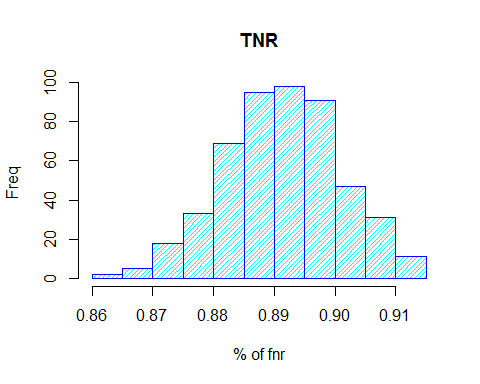
hist(fnr,xlab='% of fnr',ylab='Freq',main='FNR',  
 col='cyan',border='blue',density=30)



hist(tpr,xlab='% of fnr',ylab='Freq',main='TPR',  
 col='cyan',border='blue',density=30)



hist(tnr,xlab='% of fnr',ylab='Freq',main='TNR',  
 col='cyan',border='blue',density=30)



# Predictions for 2017

Model 2b has the highest accuracry rate.

pred <- (predict(mod\_2b, pitches\_dl\_predict, type='response'))\*100;  
  
predictions <- data.frame(rsid=pitches\_dl\_predict$rsID, probabilty=pred);  
  
dbhandle <- odbcDriverConnect('driver={SQL Server};server=localhost;database=Lahman;trusted\_connection=true');  
  
query <- "SELECT retroID as rsid, nameFirst, nameLast FROM Master";  
   
players <-sqlQuery(dbhandle, query);  
  
predictions\_players <- merge(x=predictions, y=players, by="rsid", all.x=TRUE);  
  
head(predictions\_players[rev(order(predictions\_players$probabilty)),], 20);

## rsid probabilty nameFirst nameLast  
## 201 happj001 78.69762 J. A. Happ  
## 170 gausk001 74.92285 Kevin Gausman  
## 437 sanca006 73.32695 Aaron Sanchez  
## 348 odorj001 72.10956 Jake Odorizzi  
## 341 nolar001 70.83420 Ricky Nolasco  
## 398 ray-r002 69.88841 Robbie Ray  
## 387 quinj001 66.97850 Jose Quintana  
## 241 jimeu001 65.75076 Ubaldo Jimenez  
## 276 lestj001 65.68297 Jon Lester  
## 371 perem004 65.51329 Martin Perez  
## 379 pinem001 65.34702 Michael Pineda  
## 370 peraw001 64.67090 Wily Peralta  
## 381 porcr001 64.59890 Rick Porcello  
## 15 arrij001 64.59393 Jake Arrieta  
## 199 hammj002 64.19380 Jason Hammel  
## 509 walkt004 64.17469 Taijuan Walker  
## 81 chatt001 63.70163 Tyler Chatwood  
## 252 kenni001 62.88287 Ian Kennedy  
## 472 strom001 62.70291 Marcus Stroman  
## 266 lackj001 61.72387 John Lackey