

Nolan Ryan or Tommy John?

Matt Misterka

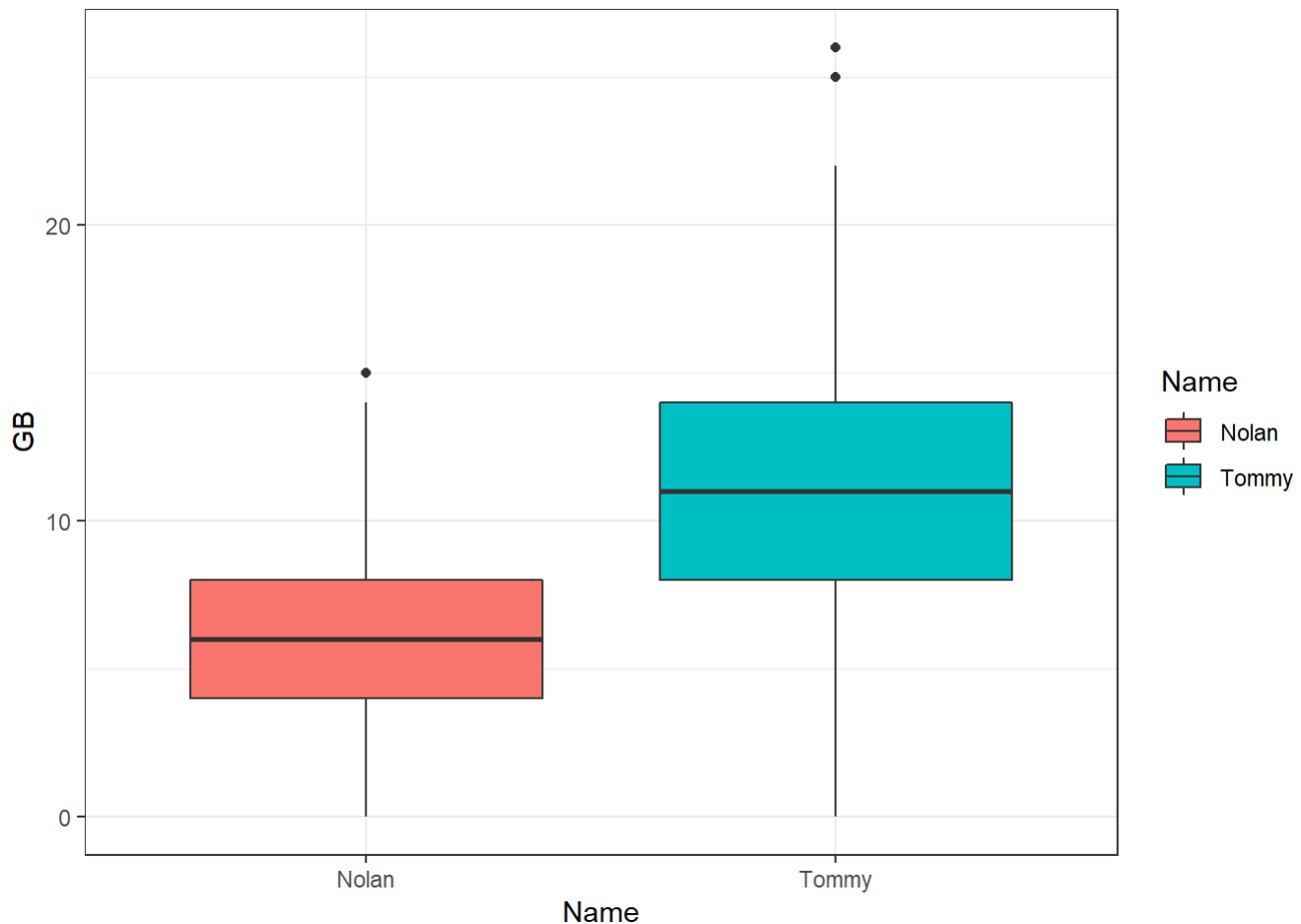
June 3, 2019

```
setwd("C:\\Users\\Matt\\Documents\\Statistics\\Stat_Learning")  
load("baseball_test.Rda")  
load("baseball_train.Rda")
```

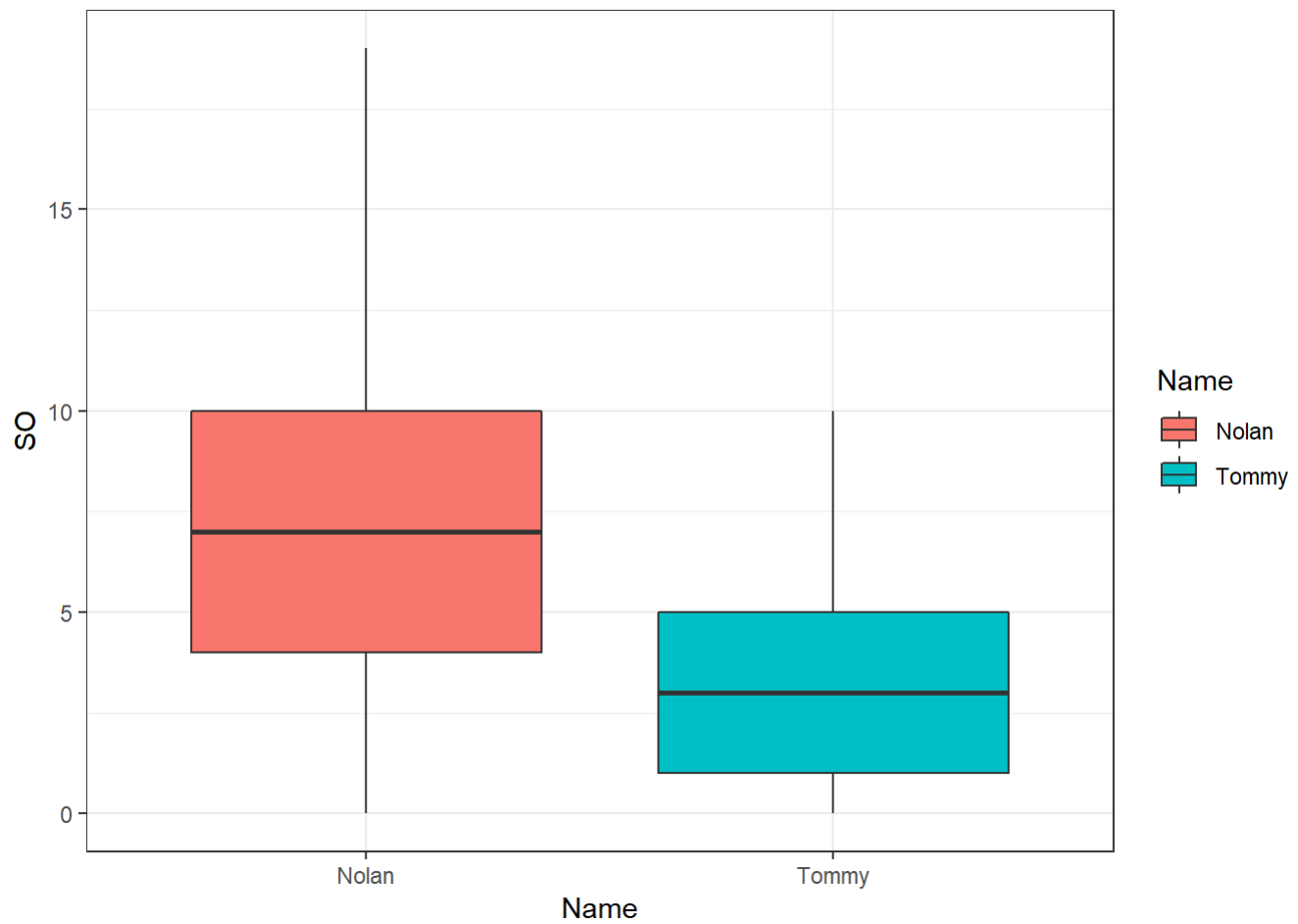
Data Visualization

Let's look at boxplots of strike outs, ground balls, and walks (BB).

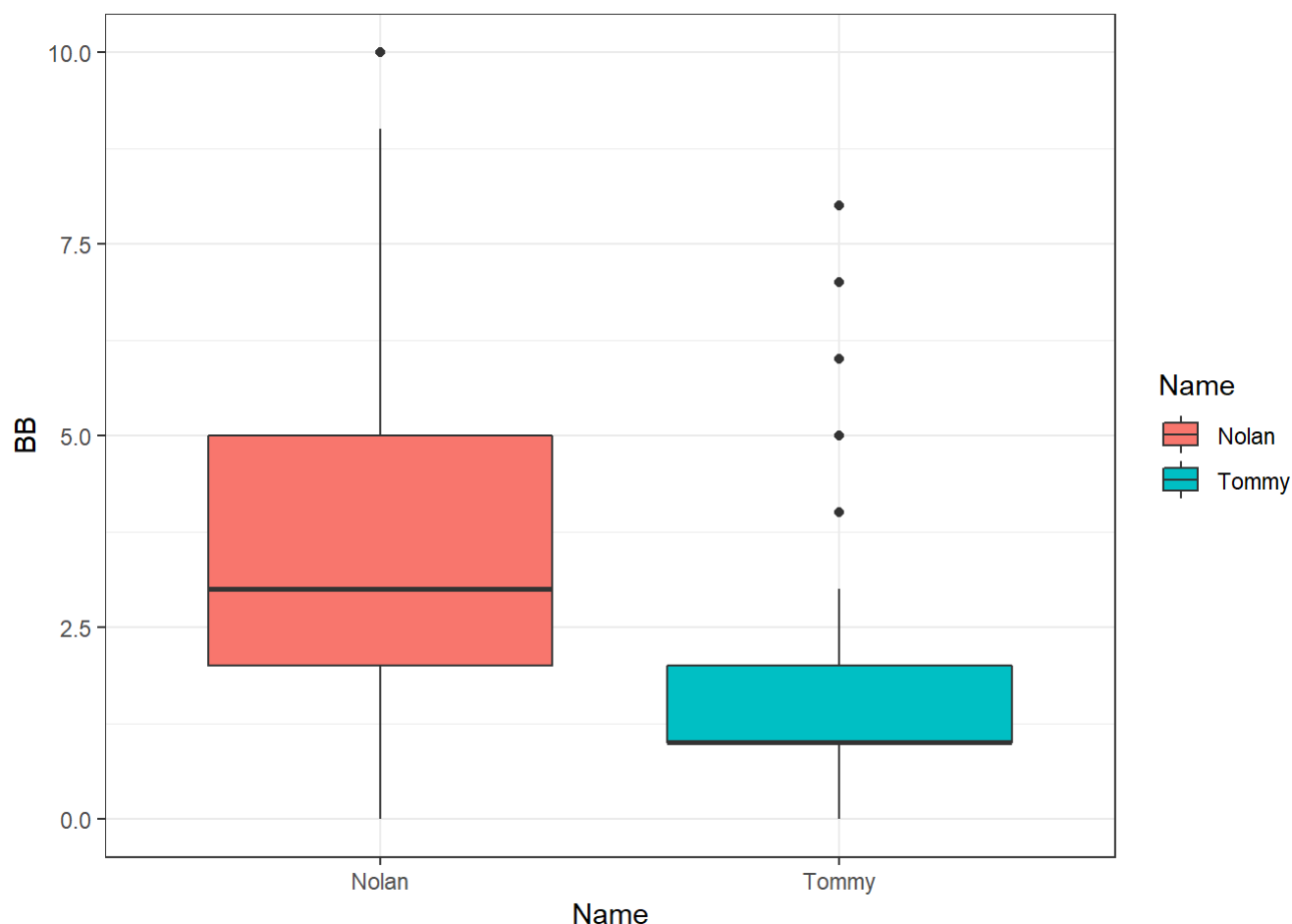
```
ggplot(baseball.train, aes(x=Name, y=GB, fill=Name)) +  
  geom_boxplot()+theme_bw()
```



```
ggplot(baseball.train, aes(x=Name, y=SO, fill=Name)) +  
  geom_boxplot()+theme_bw()
```



```
ggplot(baseball.train, aes(x=Name, y=BB, fill=Name)) +  
  geom_boxplot()+theme_bw()
```



Logistic Model

Start off with a logistic model.

```
ctrl <- trainControl(method="repeatedcv", number=5, repeats=10,
                     classProbs=TRUE, summaryFunction=twoClassSummary)

set.seed(21191)
logit <- train(Name ~.,
               data=baseball.train,
               method="glm",
               metric="ROC",
               preProcess=c("center", "scale"),
               family=binomial(link="logit"),
               trControl=ctrl)
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

Forward Logistic Model

I removed the opponent variables in this model to speed up the computation.

```
set.seed(21191)
forward_logit <- train(Name ~.-Opp,
  data=baseball.train,
  method="glmStepAIC",
  direction="forward",
  preProcess=c("center", "scale"),
  family=binomial(link="logit"),
  trace=FALSE,
  trControl=ctrl)
```

```
## Warning in train.default(x, y, weights = w, ...): The metric "Accuracy" was
## not in the result set. ROC will be used instead.
```

```
forward_logit$finalModel
```

```
##
## Call:  NULL
##
## Coefficients:
## (Intercept)          SO          GB          BB          FB
##   -0.5859      -2.2773      2.0378     -1.0383     -1.3816
##      Unk          SB          LD          R          PU
##    0.7173     -0.5631     0.5051     -0.7279     -0.5429
##      H          PO          DR          ERA          IBB
##    0.7723     -0.1949     0.2043     -0.2158     0.1989
##
## Degrees of Freedom: 1545 Total (i.e. Null);  1531 Residual
## Null Deviance:      2142
## Residual Deviance: 672   AIC: 702
```

Lasso Model

Let's look at the lasso model. This model does implicit variable selection - setting some coefficients to 0.

```
set.seed(21191)
lasso.mod <- train(Name ~.,
  data=baseball.train,
  method="glmnet",
  metric="ROC",
  preProcess=c("center", "scale"),
  trControl=ctrl,
  tuneGrid=expand.grid(alpha=1,
    lambda=10^seq(-2,0,0.01)))

kable(data.frame(Variable=c("Intercept",lasso.mod$coefnames),
  Coef=as.numeric(predict(lasso.mod$finalModel, type="coefficients",
    s=as.numeric(lasso.mod$bestTune[2])))) %>%
  kable_styling(c("bordered"),full_width = FALSE,position = "left")
```

Variable	Coef
Intercept	-0.3208870
OppBAL	0.1105138
OppBOS	0.0000000
OppCAL	0.1469484
OppCHC	-0.0947144
OppCHW	0.0000000
OppCIN	-0.0614289
OppCLE	0.0713630
OppDET	0.1279084
OppHOU	0.0000000
OppKCA	0.0811649
OppKCR	0.0000000
OppLAA	0.0000000
OppLAD	-0.3519144
OppMIL	0.0000000
OppMIN	0.0362213
OppMON	-0.1458874
OppNYM	0.0000000
OppNYY	0.0000000
OppOAK	0.1060652
OppPHI	-0.0921314
OppPIT	-0.1038198
OppSDP	-0.1364058
OppSEA	0.0000000
OppSEP	0.0000000

Variable	Coef
OppSFG	-0.0408885
OppSTL	-0.0494123
OppTEX	0.0000000
OppTOR	0.0000000
OppWSA	0.2515334
DR	0.0000000
IP	0.0000000
H	0.3922035
R	-0.2209528
ER	0.0000000
BB	-0.8687787
SO	-1.5791283
HR	0.0000000
HBP	0.0000000
ERA	-0.0364412
BF	0.0000000
GB	1.4969367
FB	-0.6865643
LD	0.0070053
PU	-0.3398111
Unk	0.2909305
SB	-0.2894411
CS	0.0000000
PO	-0.0263786
X2B	0.0000000

Variable	Coef
X3B	0.0000000
IBB	0.0284480
GDP	0.0000000
SF	0.0000000
ROE	0.0000000
ResultW	0.0000000

Elastic Model

This model finds a balance between ridge and lasso models. Investigate the coefficients.

```
set.seed(21191)
elastic.mod <- train(Name ~.,
                     data=baseball.train,
                     method="glmnet",
                     metric="ROC",
                     preProcess=c("center", "scale"),
                     trControl=ctrl,
                     tuneGrid=expand.grid(alpha=seq(0,1,0.1),
                                           lambda=10^seq(-2,0,0.01)))

kable(data.frame(Variable=c("Intercept",elastic.mod$coefnames),
                        Coef=as.numeric(predict(elastic.mod$finalModel, type="coefficients",
                                                s=as.numeric(elastic.mod$bestTune[2]))))) %>%
  kable_styling(c("bordered"),full_width = FALSE,position = "left")
```

Variable	Coef
Intercept	-0.2673120
OppBAL	0.1965888
OppBOS	0.0833879
OppCAL	0.2489383
OppCHC	-0.1280430
OppCHW	-0.0556663
OppCIN	-0.1029527
OppCLE	0.1712717

Variable	Coef
OppDET	0.2360914
OppHOU	0.0555533
OppKCA	0.1926901
OppKCR	0.0437148
OppLAA	0.0790201
OppLAD	-0.4067321
OppMIL	0.0758785
OppMIN	0.1496479
OppMON	-0.1731363
OppNYM	-0.0159089
OppNYY	0.0793489
OppOAK	0.2223586
OppPHI	-0.1318333
OppPIT	-0.1498254
OppSDP	-0.1551157
OppSEA	0.0558354
OppSEP	0.0207793
OppSFG	-0.0878814
OppSTL	-0.1029952
OppTEX	0.0019301
OppTOR	0.0934764
OppWSA	0.3338432
DR	0.0667944
IP	-0.0494596
H	0.4357988

Variable	Coef
R	-0.2261503
ER	-0.1006617
BB	-0.8240784
SO	-1.2875154
HR	-0.0008173
HBP	-0.0920938
ERA	-0.1129718
BF	-0.0841588
GB	1.2654914
FB	-0.6691734
LD	0.2131488
PU	-0.4562659
Unk	0.4404366
SB	-0.3344358
CS	0.0055955
PO	-0.0946002
X2B	0.0000000
X3B	0.0036886
IBB	0.1143809
GDP	0.0877615
SF	-0.0548286
ROE	0.0000000
ResultW	0.0583913

Dummy Variables and LDA

I create five dummy variables for opponents. Variables were chosen by looking at coefficients of the elastic model. I also took into consideration the teams that each player played on as well as divisional opponents. I selected a subset of numeric variables. Train LDA model - assumes a common covariance structure. Examine the covariance structure.

```
baseball.train$MON<-ifelse(baseball.train$Opp=="MON",1,0)
baseball.train$CAL<-ifelse(baseball.train$Opp=="CAL",1,0)
baseball.train$LAD<-ifelse(baseball.train$Opp=="LAD",1,0)
baseball.train$OAK<-ifelse(baseball.train$Opp=="OAK",1,0)
baseball.train$WSA<-ifelse(baseball.train$Opp=="WSA",1,0)

cols <- c("MON","CAL","LAD","OAK","WSA")
baseball.train[cols] <- lapply(baseball.train[cols],as.factor)

ggscatmat(baseball.train, c(4:5,7:8,13:14,16:18), color="Name",alpha = 1)
```



```
set.seed(21191)
mod.lda <- train(Name ~ H+R+BB+SO+GB+FB+PU+Unk+SB+MON+CAL+LAD+OAK+WSA,
  data=baseball.train,
  method="lda",
  metric="ROC",
  output=FALSE,
  trControl=ctrl1)
```

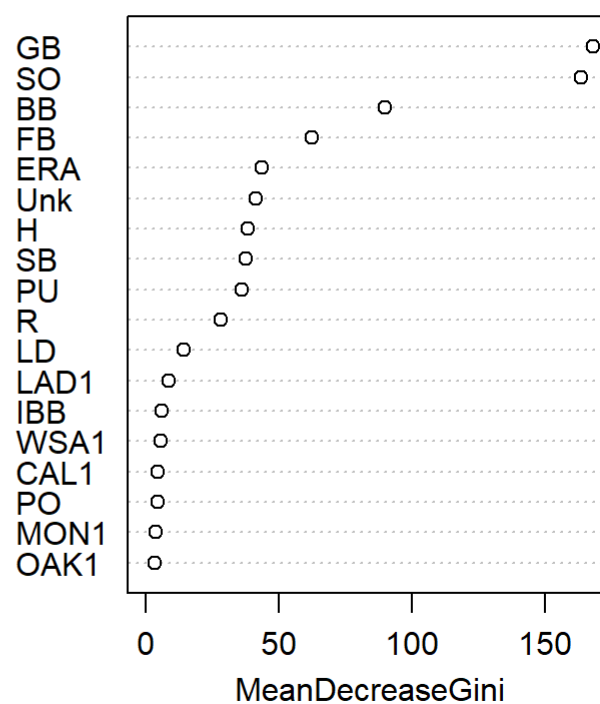
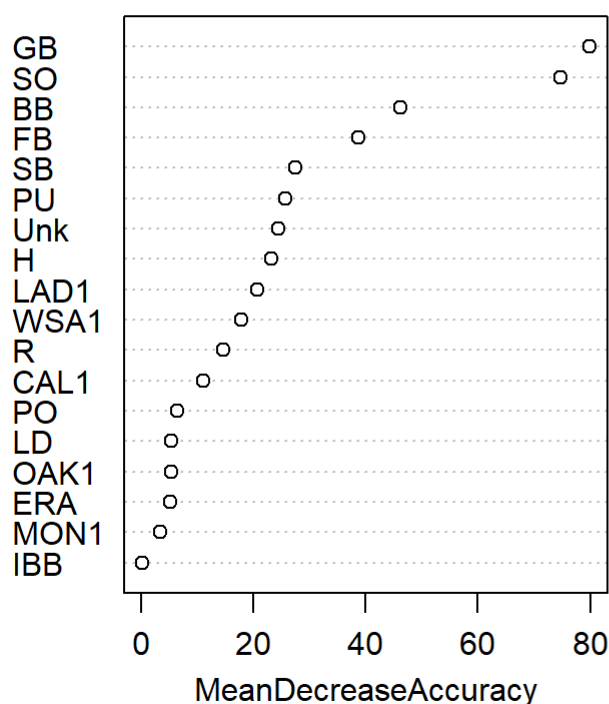
Random Forest

A random forest with selected opponents and numeric variables present in forward, lasso, and elastic models simultaneously. Let's look at a variable importance plot.

```
set.seed(21191)
mod_rf <- train(Name ~ H+R+ERA+BB+SO+GB+FB+LD+PO+PU+Unk+SB+IBB+MON+CAL+LAD+OAK+WSA,
  data=baseball.train,
  method="rf",
  metric="ROC",
  preProcess=c("center", "scale"),
  trControl=ctrl,
  tuneGrid=expand.grid(mtry=c(4,5)),
  importance=TRUE)

varImpPlot(mod_rf$finalModel)
```

mod_rf\$finalModel



Boosted Tree

```
set.seed(21191)
mod.boosttree <- train(Name ~ H+R+ERA+BB+SO+GB+FB+LD+PO+PU+Unk+SB+IBB+MON+CAL+LAD+OAK+WSA,
  data=baseball.train,
  method="gbm",
  metric="ROC",
  preProcess=c("center", "scale"),
  trControl=ctrl,
  distribution="bernoulli",
  verbose=FALSE,
  tuneGrid=expand.grid(n.trees=c(1000),
    interaction.depth=1:2,
    shrinkage=c(0.1,0.01,0.001),
    n.minobsinnode=c(10,20) ) )
```

KNN Model

```
set.seed(21191)
mod.knn <- train(Name ~ H+R+ERA+BB+SO+GB+FB+LD+PO+PU+Unk+SB+IBB+MON+CAL+LAD+OAK+WSA,
  data=baseball.train,
  method="knn",
  preProcess=c("center", "scale"),
  metric="ROC",
  trControl=ctrl,
  tuneGrid=expand.grid(k=seq(1,49,2)))
```

Comparison of Models

It looks like the elastic model does well in terms of ROC.

```
out <- resamples(list(Logistic=logit, StepLogit=forward_logit,
  Lasso=lasso.mod, Elastic=elastic.mod,
  LDA=mod.lda, RandomForest=mod_rf,
  Boosted=mod.boosttree, KNN=mod.knn))
summary(out)$statistics$ROC %>%
kable() %>% kable_styling(full_width=FALSE)
```

	Min.	1st Qu.	Median	Mean	3rd Qu.	Max.	NA's
Logistic	0.9586164	0.9716781	0.9756946	0.9750286	0.9796646	0.9901048	0
StepLogit	0.9419706	0.9632075	0.9675262	0.9673915	0.9727044	0.9825577	0
Lasso	0.9565199	0.9701048	0.9747500	0.9741430	0.9788574	0.9872117	0
Elastic	0.9603354	0.9733229	0.9767505	0.9767082	0.9823166	0.9896855	0
LDA	0.9514885	0.9703197	0.9751153	0.9734875	0.9785010	0.9854927	0
RandomForest	0.9294549	0.9591929	0.9624896	0.9629822	0.9697013	0.9800629	0

	Min.	1st Qu.	Median	Mean	3rd Qu.	Max.	NA's
Boosted	0.9421384	0.9667505	0.9731656	0.9717552	0.9775472	0.9865409	0
KNN	0.9270650	0.9535744	0.9599790	0.9586416	0.9640503	0.9774843	0

To Be Continued...