Submission 10

Shannon Rush

July 19, 2014

Building on submission 8 which fit 4 random forests based on PRI_num_jet In this submission I'll keep only those variables with a Gini importance > 10 then run another set of random forests I think this will improve accuracy because I think the low Gini importance variables are mostly contributing noise

Load and Source

```
source("../helpers/predictions.R")
```

```
library(caret)
```

```
## Loading required package: lattice
## Loading required package: ggplot2
```

```
library(doMC)
```

```
## Loading required package: foreach
## Loading required package: iterators
## Loading required package: parallel
```

```
train <- read.csv("../../data/processed/processed_train.csv")
test <- read.csv("../../data/original/test.csv")
```

Clean and Transform Data

Since the submission builds on submission 8 it will use the same seeds

```
set.seed(123)
training.indices <- createDataPartition(train$Label, p=0.6, list=F)
training <- train[training.indices,]
validation <- train[-training.indices,]
```

```
train.jets <- split(train, train$PRI_jet_num)
for (i in 0:3) { assign(paste0("train.jets",i),train.jets[[i+1]]) }
val.jets <- split(validation, validation$PRI_jet_num)
for (i in 0:3) { assign(paste0("val.jets",i),val.jets[[i+1]]) }
test.jets <- split(test, test$PRI_jet_num)
for (i in 0:3) { assign(paste0("test.jets",i),test.jets[[i+1]]) }
```

0 Jets

```r
remove<-apply(train.jets0[1:nrow(train.jets0),]==-999, 2, all)
train.j0 <- train.jets0[,!remove]
val.j0 <- val.jets0[,!remove]
test.j0 <- test.jets0[,c("EventId",setdiff(names(!remove),"Label"))]
```

```r
keep <- setdiff(names(train.j0),c("PRI_jet_num","PRI_jet_all_pt"))
train.j0 <- train.j0[,keep]
val.j0 <- val.j0[,keep]
test.j0 <- test.j0[,setdiff(keep,"Label")]
```

```r
load("../submission8/RData/trainj0.RData")
elim.0 <- row.names(subset(varImp(trainj0.fit)$importance, Overall < 10))
keep <- setdiff(names(train.j0),elim.0)
train.j0 <- train.j0[,keep]
val.j0 <- val.j0[,keep]
test.j0 <- test.j0[,setdiff(keep,"Label")]
```

```r
names(train.j0)
```

```
##  [1] "EventId"                     "DER_mass_MMC"
##  [3] "DER_mass_transverse_met_lep" "DER_mass_vis"
##  [5] "DER_deltar_tau_lep"          "DER_sum_pt"
##  [7] "DER_pt_ratio_lep_tau"        "PRI_tau_pt"
##  [9] "PRI_lep_pt"                  "PRI_met"
## [11] "Label"
```

```r
names(test.j0)
```

```
##  [1] "EventId"                     "DER_mass_MMC"
##  [3] "DER_mass_transverse_met_lep" "DER_mass_vis"
##  [5] "DER_deltar_tau_lep"          "DER_sum_pt"
##  [7] "DER_pt_ratio_lep_tau"        "PRI_tau_pt"
##  [9] "PRI_lep_pt"                  "PRI_met"
```

```r
set.seed(999)
predictors <- train.j0[,setdiff(names(train.j0),c("EventId","Label"))]
registerDoMC(cores=4)
j0.fit <- train(x=predictors, y=train.j0$Label, method="rf", proxy=T)
```

```r
j0.fit
```

```
## Random Forest
##
## 99913 samples
##     9 predictors
##     2 classes: 'b', 's'
##
## No pre-processing
## Resampling: Bootstrapped (25 reps)
```

```
##
## Summary of sample sizes: 99913, 99913, 99913, 99913, 99913, 99913, ...
##
## Resampling results across tuning parameters:
##
##   mtry  Accuracy  Kappa  Accuracy SD  Kappa SD
##   2     0.8       0.6    0.001        0.003
##   5     0.8       0.6    0.002        0.004
##   9     0.8       0.6    0.002        0.004
##
## Accuracy was used to select the optimal model using  the largest value.
## The final value used for the model was mtry = 2.
```

```
pred.val0 <- predict(j0.fit, val.j0)
confusionMatrix(pred.val0, val.j0$Label)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction     b     s
##          b 29629     0
##          s     0 10173
##
##                Accuracy : 1
##                  95% CI : (1, 1)
##     No Information Rate : 0.744
##     P-Value [Acc > NIR] : <2e-16
##
##                   Kappa : 1
##  Mcnemar's Test P-Value : NA
##
##             Sensitivity : 1.000
##             Specificity : 1.000
##          Pos Pred Value : 1.000
##          Neg Pred Value : 1.000
##              Prevalence : 0.744
##          Detection Rate : 0.744
##    Detection Prevalence : 0.744
##       Balanced Accuracy : 1.000
##
##        'Positive' Class : b
##
```

```
pred.test0 <- predict(j0.fit, test.j0, type="prob")
pred.0 <- PrepPrediction(pred.test0, test.j0)
```

1 Jet

```
remove <- apply(train.jets1[1:nrow(train.jets1),]==-999, 2, all)
train.j1 <- train.jets1[,!remove]
val.j1 <- val.jets1[,!remove]
test.j1 <- test.jets1[,c("EventId",setdiff(names(!remove),"Label"))]
```

```r
keep <- setdiff(names(train.j1),c("PRI_jet_num"))
train.j1 <- train.j1[,keep]
val.j1 <- val.j1[,keep]
test.j1 <- test.j1[,c("EventId",setdiff(keep,"Label"))]
```

```r
load("../submission8/RData/trainj1.RData")
elim.1 <- row.names(subset(varImp(trainj1.fit)$importance, Overall < 10))
keep <- setdiff(names(train.j1),elim.1)
train.j1 <- train.j1[,keep]
val.j1 <- val.j1[,keep]
test.j1 <- test.j1[,setdiff(keep,"Label")]
```

```r
names(train.j1)
```

```
## [1] "EventId"                       "DER_mass_MMC"
## [3] "DER_mass_transverse_met_lep" "DER_mass_vis"
## [5] "DER_met_phi_centrality"      "PRI_tau_pt"
## [7] "PRI_jet_leading_eta"         "Label"
```

```r
names(test.j1)
```

```
## [1] "EventId"                       "DER_mass_MMC"
## [3] "DER_mass_transverse_met_lep" "DER_mass_vis"
## [5] "DER_met_phi_centrality"      "PRI_tau_pt"
## [7] "PRI_jet_leading_eta"
```

```r
set.seed(999)
predictors <- train.j1[,setdiff(names(train.j1),c("EventId","Label"))]
Sys.time()
```

```
## [1] "2014-07-20 15:34:44 MDT"
```

```r
registerDoMC(cores=4)
j1.fit <- train(x=predictors, y=train.j1$Label, method="rf", proxy=T)
Sys.time()
```

```
## [1] "2014-07-20 16:37:53 MDT"
```

```r
j1.fit
```

```
## Random Forest
##
## 77544 samples
##     6 predictors
##     2 classes: 'b', 's'
##
## No pre-processing
## Resampling: Bootstrapped (25 reps)
##
```

```
## Summary of sample sizes: 77544, 77544, 77544, 77544, 77544, 77544, ...
##
## Resampling results across tuning parameters:
##
##   mtry  Accuracy  Kappa  Accuracy SD  Kappa SD
##   2     0.8       0.6    0.001        0.003
##   4     0.8       0.6    0.001        0.003
##   6     0.8       0.6    0.002        0.004
##
## Accuracy was used to select the optimal model using  the largest value.
## The final value used for the model was mtry = 2.
```

```
pred.val1 <- predict(j1.fit, val.j1)
confusionMatrix(pred.val1, val.j1$Label)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction     b     s
##          b 19880     0
##          s     0 11079
##
##                Accuracy : 1
##                  95% CI : (1, 1)
##     No Information Rate : 0.642
##     P-Value [Acc > NIR] : <2e-16
##
##                   Kappa : 1
##  Mcnemar's Test P-Value : NA
##
##             Sensitivity : 1.000
##             Specificity : 1.000
##          Pos Pred Value : 1.000
##          Neg Pred Value : 1.000
##              Prevalence : 0.642
##          Detection Rate : 0.642
##    Detection Prevalence : 0.642
##       Balanced Accuracy : 1.000
##
##        'Positive' Class : b
##
```

```
pred.test1 <- predict(j1.fit, test.j1, type="prob")
pred.1 <- PrepPrediction(pred.test1, test.j1)
```

2 Jets

```
keep <- setdiff(names(train.jets2),c("PRI_jet_num"))
train.j2 <- train.jets2[,keep]
val.j2 <- val.jets2[,keep]
test.j2 <- test.jets2[,c("EventId",setdiff(keep,"Label"))]
```

```r
load("../submission8/RData/trainj2.RData")
elim.2 <- row.names(subset(varImp(trainj2.fit)$importance, Overall < 10))
keep <- setdiff(names(train.j2),elim.2)
train.j2 <- train.j2[,keep]
val.j2 <- val.j2[,keep]
test.j2 <- test.j2[,setdiff(keep,"Label")]
```

```r
names(train.j2)
```

```
## [1] "EventId"                     "DER_mass_MMC"
## [3] "DER_mass_transverse_met_lep" "DER_mass_vis"
## [5] "DER_deltaeta_jet_jet"        "DER_mass_jet_jet"
## [7] "DER_lep_eta_centrality"      "Label"
```

```r
names(test.j2)
```

```
## [1] "EventId"                     "DER_mass_MMC"
## [3] "DER_mass_transverse_met_lep" "DER_mass_vis"
## [5] "DER_deltaeta_jet_jet"        "DER_mass_jet_jet"
## [7] "DER_lep_eta_centrality"
```

```r
set.seed(999)
predictors <- train.j2[,setdiff(names(train.j2),c("EventId","Label"))]
Sys.time()
```

```
## [1] "2014-07-20 16:46:13 MDT"
```

```r
registerDoMC(cores=4)
j2.fit <- train(x=predictors, y=train.j2$Label, method="rf", proxy=T)
Sys.time()
```

```
## [1] "2014-07-20 17:19:18 MDT"
```

```r
j2.fit
```

```
## Random Forest
##
## 50379 samples
##     6 predictors
##     2 classes: 'b', 's'
##
## No pre-processing
## Resampling: Bootstrapped (25 reps)
##
## Summary of sample sizes: 50379, 50379, 50379, 50379, 50379, 50379, ...
##
## Resampling results across tuning parameters:
##
##   mtry  Accuracy  Kappa  Accuracy SD  Kappa SD
```

```
##   2     0.8       0.7    0.002        0.004
##   4     0.8       0.7    0.002        0.005
##   6     0.8       0.6    0.002        0.004
##
## Accuracy was used to select the optimal model using  the largest value.
## The final value used for the model was mtry = 2.
```

```
pred.val2 <- predict(j2.fit, val.j2)
confusionMatrix(pred.val2, val.j2$Label)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction     b     s
##          b  9916     0
##          s     0 10276
##
##                Accuracy : 1
##                  95% CI : (1, 1)
##     No Information Rate : 0.509
##     P-Value [Acc > NIR] : <2e-16
##
##                   Kappa : 1
##  Mcnemar's Test P-Value : NA
##
##             Sensitivity : 1.000
##             Specificity : 1.000
##          Pos Pred Value : 1.000
##          Neg Pred Value : 1.000
##              Prevalence : 0.491
##          Detection Rate : 0.491
##    Detection Prevalence : 0.491
##       Balanced Accuracy : 1.000
##
##        'Positive' Class : b
##
```

```
pred.test2 <- predict(j2.fit, test.j2, type="prob")
pred.2 <- PrepPrediction(pred.test2, test.j2)
```

3 Jets

```
keep <- setdiff(names(train.jets3),c("PRI_jet_num"))
train.j3 <- train.jets3[,keep]
val.j3 <- val.jets3[,keep]
test.j3 <- test.jets3[,c("EventId",setdiff(keep,"Label"))]
```

```
load("../submission8/RData/trainj3.RData")
elim.3 <- row.names(subset(varImp(trainj3.fit)$importance, Overall < 10))
keep <- setdiff(names(train.j3),elim.3)
```

```r
train.j3 <- train.j3[,keep]
val.j3 <- val.j3[,keep]
test.j3 <- test.j3[,setdiff(keep,"Label")]
```

```r
names(train.j3)
```

```
## [1] "EventId"                     "DER_mass_MMC"
## [3] "DER_mass_transverse_met_lep" "DER_mass_vis"
## [5] "DER_deltar_tau_lep"          "DER_met_phi_centrality"
## [7] "Label"
```

```r
names(test.j3)
```

```
## [1] "EventId"                     "DER_mass_MMC"
## [3] "DER_mass_transverse_met_lep" "DER_mass_vis"
## [5] "DER_deltar_tau_lep"          "DER_met_phi_centrality"
```

```r
set.seed(999)
predictors <- train.j3[,setdiff(names(train.j3),c("EventId","Label"))]
Sys.time()
```

```
## [1] "2014-07-21 06:31:03 MDT"
```

```r
registerDoMC(cores=4)
j3.fit <- train(x=predictors, y=train.j3$Label, method="rf", proxy=T)
Sys.time()
```

```
## [1] "2014-07-21 06:43:43 MDT"
```

```r
j3.fit
```

```
## Random Forest
##
## 22164 samples
##     5 predictors
##     2 classes: 'b', 's'
##
## No pre-processing
## Resampling: Bootstrapped (25 reps)
##
## Summary of sample sizes: 22164, 22164, 22164, 22164, 22164, 22164, ...
##
## Resampling results across tuning parameters:
##
##   mtry  Accuracy  Kappa  Accuracy SD  Kappa SD
##   2     0.8       0.6    0.004        0.009
##   3     0.8       0.6    0.004        0.009
##   5     0.8       0.6    0.004        0.009
##
## Accuracy was used to select the optimal model using  the largest value.
## The final value used for the model was mtry = 2.
```

```r
pred.val3 <- predict(j3.fit, val.j3)
confusionMatrix(pred.val3, val.j3$Label)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    b    s
##          b 6308    0
##          s    0 2738
##
##                Accuracy : 1
##                  95% CI : (1, 1)
##     No Information Rate : 0.697
##     P-Value [Acc > NIR] : <2e-16
##
##                   Kappa : 1
##  Mcnemar's Test P-Value : NA
##
##             Sensitivity : 1.000
##             Specificity : 1.000
##          Pos Pred Value : 1.000
##          Neg Pred Value : 1.000
##              Prevalence : 0.697
##          Detection Rate : 0.697
##    Detection Prevalence : 0.697
##       Balanced Accuracy : 1.000
##
##        'Positive' Class : b
##
```

```r
pred.test3 <- predict(j3.fit, test.j3, type="prob")
pred.3 <- PrepPrediction(pred.test3, test.j3)
```

Submission

```r
pred <- rbind(pred.0,pred.1,pred.2,pred.3)
WriteSubmission(pred, 10)
```

Result

2.69242