**Breakout:**

Using the Caravan data set in the ISLR package:

1. Split the data, 2/3 for training and 1/3 for testing.

2. Build a Random Forest classification model to predict Purchase using the training subset and mtry=20.

3. Now, predict results for test data. Set type="response" first.

4. Use the function provided to calculate error rate, sensitivity and specificity.

5. Now predict with type="prob". Try a higher threshold and a lower threshold and recalculate error metrics (keep classes as "Yes" and "No" for function to work).

6. Which direction is better and why?

7. Repeat this process with mtry=10. Any opinions on which is better?

8. If you have extra time, what is a general process you would use to tune in this case?

**Code:**

#Random Forest Exercise

errorStats=function(preds, actuals) {

rf.error.rate=length(which(preds!=actuals))/length(preds)

rf.sensitivity=length(intersect(which(preds=="Yes"),which(actuals=="Yes")))/

length(which(actuals=="Yes"))

rf.specificity=length(intersect(which(preds=="No"),which(actuals=="No")))/

length(which(actuals=="No"))

rf.false.pos <- 1 - rf.specificity

return(c(rf.error.rate, rf.sensitivity, rf.specificity, rf.false.pos))

}

require(ISLR)

dim(Caravan)

attach(Caravan)

head(Caravan)

summary(Purchase)

?Caravan

train=sample(1:nrow(Caravan), 2/3\*nrow(Caravan))

test=(-train)

test.Y=Purchase[test]

rf.fit=randomForest(Purchase~.,data=Caravan,subset=train,

mtry=10,ntree=50)

rf.preds=predict(rf.fit,Caravan[test,],type="response")

#error metrics

orig.stats=errorStats(rf.preds, Caravan$Purchase[test])

table(rf.preds, Caravan$Purchase[test])

1-(1797+7)/(1797+118+7+19)

rf.probs=predict(rf.fit,Caravan[test,],type="prob")

#diff thresholds

rf.preds = ifelse(rf.probs[,2]>.9, "Yes", "No")

threshHigh.stats=errorStats(rf.preds, Caravan$Purchase[test])

rf.preds = ifelse(rf.probs[,2]>.1, "Yes", "No")

threshLow.stats=errorStats(rf.preds, Caravan$Purchase[test])

data=rbind(threshHigh.stats,orig.stats, threshLow.stats)

data=as.data.frame(data)

names(data)=

c("Lambda","Error Rate", "Sensitivity", "Specificity", "FalsePositiveRt")

data