#https://www.youtube.com/watch?v=9wn1f-30\_ZY

#https://github.com/h2oai/h2o-tutorials/blob/master/tutorials/gbm-randomforest/GBM\_RandomForest\_Example.R

library(h2o)

h2o.init(nthreads=-1,

max\_mem\_size = "2G")

h2o.removeAll()

#load the file

df <- h2o::h2o.importFile(path = normalizePath("/Users/Joe/Desktop/Region0.csv"))

#Set variables as factors

df$REGION <- as.factor(df$REGION)

df$OWNERSHP<- as.factor(df$OWNERSHP)

df$FOODSTMP<-as.factor(df$FOODSTMP)

df$SEX <- as.factor(df$SEX)

df$MARST <- as.factor(df$MARST)

df$RACE <- as.factor(df$RACE)

df$HISPAN <- as.factor(df$HISPAN)

df$EDUC <- as.factor(df$EDUC)

df$EMPSTAT <- as.factor(df$EMPSTAT)

df$GENERATION <- as.factor(df$GENERATION)

df$VETSTAT <- as.factor(df$VETSTAT)

df$NCHILD <- as.factor(df$NCHILD)

df$POVERTY<-as.factor(df$POVERTY)

df$AGE<-as.factor(df$AGE)

df$INCTOT<-as.factor(df$INCTOT)

df$FTOTINC<-as.factor(df$FTOTINC)

df$HHINCOME<-as.factor(df$HHINCOME)

df$HCOVANY<-as.factor(df$HCOVANY

str(df)

#splits the data into three splits train/test/validate

splits <- h2o.splitFrame(

df,

c(0.6, 0.2),

seed=1234)

#RUN THESE EACH ON THEIR OWN

train <- h2o.assign(splits[[1]], "train.hex")

valid <- h2o.assign(splits[[2]], "train.hex")

test <- h2o.assign(splits[[3]], "train.hex")

train[1:5,]

rf1 <- h2o.randomForest(

training\_frame = train,

validation\_frame = valid,

x=1:19,

y=20,

model\_id = "rf\_Sub2\_v1",

ntrees = 200,

stopping\_rounds = 2,

stopping\_metric = "logloss",

score\_each\_iteration = T,

seed = 1000000)

summary(rf1)

h2o.varimp(rf1)

rf1@model$validation\_metrics

h2o.hit\_ratio\_table(rf1, valid = T)[1,2] #NOT WORKING

#nfolds = 2)

## Now we will try GBM.

## First we will use all default settings, and then make some changes,

## where the parameters and defaults are described.

gbm1 <- h2o.gbm(

training\_frame = train, ## the H2O frame for training

validation\_frame = valid, ## the H2O frame for validation (not required)

x=1:19, ## the predictor columns, by column index

y=20, ## the target index (what we are predicting)

model\_id = "gbm1\_”Sub2", ## name the model in H2O

seed = 2000000) ## Set the random seed for reproducability

summary(gbm1)

h2o.hit\_ratio\_table(gbm1,valid = T)[1,2] #NOT WORKING

gbm2 <- h2o.gbm(

training\_frame = train, ##

validation\_frame = valid, ##

x=1:19, ## the predictor columns, by column index

y=20, ##

ntrees = 20, ## decrease the trees, mostly to allow for run time

## (from 50)

learn\_rate = 0.2, ## increase the learning rate (from 0.1)

max\_depth = 10, ## increase the depth (from 5)

stopping\_rounds = 2, ##

stopping\_tolerance = 0.01,

stopping\_metric = "logloss",##

score\_each\_iteration = T, ##

model\_id = "gbm2\_Region0", ##

seed = 2000000) ##

summary(gbm2)

#BELOW ISNT WORKING

h2o.hit\_ratio\_table(gbm1,valid = T)[1,2] ## review the first model's accuracy

h2o.hit\_ratio\_table(gbm2,valid = T)[1,2] ## review the new model's accuracy

gbm3 <- h2o.gbm(

training\_frame = train, ##

validation\_frame = valid, ##

x=1:19, ##

y=20, ##

ntrees = 30, ## add a few trees (from 20, though default is 50)

learn\_rate = 0.3, ## increase the learning rate even further

max\_depth = 10, ##

sample\_rate = 0.7, ## use a random 70% of the rows to fit each tree

col\_sample\_rate = 0.7, ## use 70% of the columns to fit each tree

stopping\_rounds = 2, ##

stopping\_tolerance = 0.01,

stopping\_metric = "logloss",##

score\_each\_iteration = T, ##

model\_id = "gbm3\_Sub2", ##

seed = 2000000) ##

summary(gbm3)

rf2 <- h2o.randomForest( ##

training\_frame = train, ##

validation\_frame = valid, ##

x=1:19, ##

y=20, ##

model\_id = "rf\_Sub2", ##

ntrees = 200, ##

max\_depth = 30, ## Increase depth, from 20

stopping\_rounds = 2, ##

stopping\_tolerance = 1e-2,

stopping\_metric = "logloss", ##

score\_each\_iteration = T, ##

seed=3000000) ##

summary(rf2)

summary(rf1)

finalRf\_predictions<-h2o.predict(

object = rf1

,newdata = test)

finalRf\_predictions

h2o.hit\_ratio\_table(rf1,valid = T)[1,2]

mean(finalRf\_predictions$predict==test$Cover\_Type)

h2o.shutdown(prompt=FALSE)