```
#Clear workspace.
rm(list=ls())

# Step 1: Get dataset to work with.
cmsrural <- read.csv("~/Box Sync/Spring 2016/N248 Machine Learning/
finalproject/cmsrural.csv")

# Research Question: Do hospitals in rural California populations have
worse CMS-reported risk-adjusted inaptient mortality outcomes?
# The dataset is a CMS inpatient mortality outcomes report
(7/2011-6/2013), joined on Zip code with 2010 U.S. Census data
# Predictor variables of interest are sex (female:male ratio), age,
and population density (n/square mile)


# Step 2: Recode the variables

#create new variable hospital and drop old variable
cmsrural$hospital <- cmsrural$Hospital.Name
cmsrural$Hospital.Name <- NULL

#Dummy code outcome variable.
cmsrural$worse <- ifelse(cmsrural$Deaths.Compared.to.National!="Worse
than the National Rate",0,1)

# create population variable female:male ratio
cmsrural$fmr <- cmsrural$Female/cmsrural$Male
cmsrural$FMR <- NULL

#create new variable age and rop old variable
cmsrural$age <- cmsrural$Median.age..years.
cmsrural$Median.age..years.<- NULL

#create new variable density and drop old variable
cmsrural$density <- cmsrural$Population.Per.Square.Mile
cmsrural$Population.Per.Square.Mile <- NULL

#Generate row id.
cmsrural$rid <- 1:nrow(cmsrural)
#Put the row id column at the front.
cmsrural <- cmsrural[,unique(c("rid",names(cmsrural)))] #the source
code used rid to partition the dataset into train and test


#Step 3: Split sample into random test and train groups

library(caTools)
set.seed(411)
spl=sample.split (cmsrural, SplitRatio=0.5) #50% trainng, 50% testing
Train=cmsrural[spl, 1:12]
```

```
Test=cmsrural[!spl, 1:12]


# Step 4: Model building

# Logistic Regression.
mdl <- glm(worse ~ density + age + fmr,
family="binomial"(link="logit"), data=Train)
#Present a standard model summary.
summary(mdl)

# CIs using profiled log-likelihood
confint(mdl)
## Obtain odds ratios
exp(coef(mdl))

#Get prediction.
cmsrural$prd.glm <- predict(mdl, newdata=cmsrural, type="response")

#Clean up.
rm(mdl)

# Recursive partitioning: decision tree is constructed by either
splitting or not splitting each node on the tree into two daughter
nodes
require(randomForest)           #Random forest.
require(party)                  #Statistically based recursive
partitioning.
require(gbm)                    #Generalized boosted machines.

# Random Forest
set.seed(411)
mdl <- randomForest(worse ~ density + age + fmr, data=Train)

#Dotchart of variable importance as measured by a Random Forest
varImpPlot(mdl)

#Get prediction (new column placed in cmsrural)
cmsrural$prd.rf <- predict(mdl, newdata=cmsrural, type="response")

#Clean up.
rm(mdl)


#Build conditional tree model.
set.seed(411)
mdl <- ctree(  as.factor(worse) ~ density + age + fmr, data=Train)

# Plot it
plot(mdl)
```

```
#Get preciction
cmsrural$prd.ctr <- unlist(predict(mdl, newdata=cmsrural,
type="prob"))[1:(nrow(cmsrural)*2)%%2==0]

#Clean up.
rm(mdl)

# Generalized Boosted Model (Gradient Boosted Machine)

mdl <- gbm(  worse ~ density + age + fmr, data=Train,
distribution="bernoulli", n.trees=10^4)

#Show plot of performance and store best
bst <- gbm.perf(mdl,method="OOB")

#Get prediction.
cmsrural$prd.gbm <- predict(mdl, newdata=cmsrural, bst,
type="response")

#Clean up.
rm(mdl, bst)

# Neural net.
require(nnet)

mdl <- nnet(as.factor(worse) ~ density + age + fmr, data=Train,
size=2)
mdl

#Get prediction.
cmsrural$prd.ann <- predict(mdl, newdata=cmsrural)

#Clean up.
rm(mdl)



# Step 5: Produce summary AUC plots.
require(caTools)

colAUC(X=cmsrural$prd.glm, y=cmsrural$worse, plotROC = TRUE)
title(main="
- Logistic Regression")

colAUC(X=cmsrural$prd.rf, y=cmsrural$worse, plotROC = TRUE)
title(main ="
- Random Forest")

colAUC(X=cmsrural$prd.ctr, y=cmsrural$worse, plotROC = TRUE)
```

```
title(main="
- Conditional Tree Model")

colAUC(X=cmsrural$prd.gbm, y=cmsrural$worse, plotROC = TRUE)
title(main="
- Gen. Boosted Model")

colAUC(X=cmsrural$prd.ann, y=cmsrural$worse, plotROC = TRUE)
title(main="
- Neural Net")


# RF has best AUC at .86
```