Ensemble learners using the bagging method: ***Data-mining approach to defining health professional shortage areas***

CSC 529: Advance Data Mining Winter 2016
Mai Yang Xiong
Naga Venkateshwarlu Yadav Dokku

Abstract:

Abstract Ensemble methods are learning algorithms that construct a set of classifiers. Ensemble learning technique attracted much attention in the past few years. Instead of using a single predictor, this approach utilizes a number of diverse accurate predictors to do the job. Many methods have been proposed to build such accurate diverse ensembles, of which bagging was the most popular. Here we review bagging method and explains why ensembles can often perform better than any single classifier. Bagging is one of the most effective computationally intensive procedures to improve on unstable estimators or classifiers, useful especially for high dimensional data set problems. Here we formalize the notion of instability and derive theoretical results to analyze the variance reduction effect of bagging (or variants thereof) in mainly hard decision problems, which include estimation after testing in regression and decision trees for regression functions and classifiers. Hard decisions create instability, and bagging is shown to smooth such hard decisions, yielding smaller variance and mean squared error. Bagging predictors is a method for generating multiple versions of a predictor and using these to get an aggregated predictor. The aggregation averages over the versions when predicting a numerical outcome and does a plurality vote when predicting a class. The multiple versions are formed by making bootstrap replicates of the learning set and using these as new learning sets. Tests on real and simulated data sets using classification and regression trees and subset selection in linear regression show that bagging can give substantial gains in accuracy. The vital element is the instability of the prediction method. If perturbing the learning set can cause significant changes in the predictor constructed, then bagging can improve accuracy. Recent empirical work has shown that combining predictors can lead to significant reduction in generalization error. The individual predictors (weak learners) can be very simple, such as two terminal-node trees; it is the aggregating scheme that gives them the power of increasing prediction accuracy The task of the classifiers is to develop shortage designation criteria and uses them to decide whether or not a geographic area or population group is a Health Professional Shortage Area (HPSA).

Introduction

Advances in data collection and computing technologies have led to the proliferation of large data sets. Bagging is one of the recent and successful computationally intensive methods for improving unstable estimation or classification schemes. It is extremely useful for large, high dimensional data set problems where finding a good model or classifier in one step is impossible because of the complexity and scale of the problem. Bagging (bootstrap aggregating) was introduced by Breiman (1996a) to reduce the variance of a predictor. It has attracted much attention and is frequently applied, although deep theoretical insight has been lacking. To classify an instance a vote for each class is recorded by every classifier that chooses it, and the class with the most votes is chosen by the aggregating scheme. Bagging, or bootstrap aggregation, was introduced by Breiman (1996, 1999) as a means for improving the performance of a 'predictor', e.g. a classifier, by combining the results of many empirically simulated predictions. Bagging a conventional classifier, in particular one based on random forest, can sometimes, although not always, reduce the error rate. Other recent contributions to bagging and to related methodology include those of Ho (1998a, b), Skurichina and Duin (1998), Bay (1999), Guerra-Salcedo and Whitley (1999), Zemke (1999), Francois et al. (2001), Kuncheva et al. (2002) and Skurichina et al. (2002). Figure 1 shows a diagram of how bagging is done.

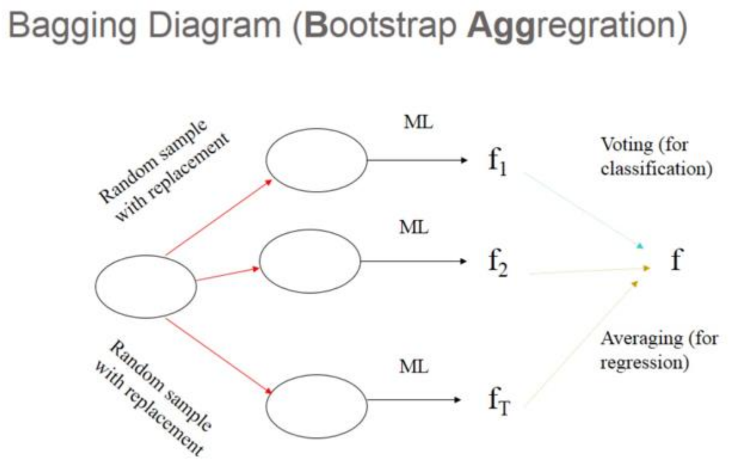
Bagging Diagram (Bootstrap Aggregration)
Random sample with replacement
Random sample with replacement
ML
ML
ML
f_1
f_2
f_T
f
Voting (for classification)
Averaging (for regression)

Figure 1. Ensemble learners using the bagging process from CSC 529 Daniela Stan Raicu, DePaul University

Ensemble learning typically refers to methods that generate several models which are combined to make a prediction, either in classification or regression problems. This approach has been the object of a significant amount of research in recent years and good results have been reported. The advantage of ensembles with respect to single models has been reported in terms of increased robustness and accuracy. Our work on ensemble learning focuses on classification problem. Access to healthcare is a national concern for underserved communities across the country. A variety of federal programs use the Health Professional Shortage Area (HPSA) designation to improve access to healthcare by focusing aid and assistance on areas and populations with the greatest unmet need. A health professional shortage area (HPSA) is a geographic area, population group, or health care facility that has been designated by the United States government as having an insufficient supply of medical providers, based on certain provider-to-population ratios. Primary care physician shortage areas are usually defined as any geography with less than one general practitioner for every 3,500 residents. Dental and mental health shortage areas have fewer than one dentist or mental health professional per 5,000 and 30,000 residents, respectively. States and locales have some financial incentives to receive this designation: Medicare bonuses, loan repayment, etc. The United States health resources and services administration (HPSA) holds the key to this data castle.

Related Work

Ensemble learning is the process of combining decisions of multiple classification models into a single final result (Koziol et al. 2009). The main objective of ensemble methods is not only improving overall classification performance (Dietterich 2000) but also more accurate generalization capability in classifying unseen instances (Yang et al. 2010). There are two key factors that affect ensemble method performance: the accuracy and the diversity of the base classifiers (Dietterich 2000). In this study, our focus is on most popular ensemble techniques: Bagging (Breiman 1996) Boosting (Freund & Schapire 1996) and random forest (Breiman 2001) and so we compare our approach to them.

Several scholars have investigated both Bagging and Boosting in their works. For example, Nagi et al. (Nagi & Bhattacharyya 2013) conducted an empirical study using nine high-dimensional cancer datasets and three classifiers. The researchers proposed a new ensemble method and compared class-specific accuracy of their method versus each single classifier as well as Bagging and Boosting. Another work by Tan et al. (Tan & Gilbert 2003) used seven cancer gene expression datasets along with the C4.5 decision tree classifier, and two ensemble methods: Bagging and Boosting with decision trees as the classifier. Chen (Chen 2014) conducted an experiment using eight microarray datasets and one feature selection technique, Relief-F.

In a Bagging ensemble, each classifier is trained on a set of m training examples, drawn randomly with replacement from the original training set of size m. Such a training set is called a bootstrap replicate of the original set. Each bootstrap replicate contains, on average, 67% of the original training set, with many examples appearing multiple times. Predictions on new examples are made by taking the majority vote of the ensemble. Since each ensemble member is not exposed to the same set of examples, they are different from each other. By voting the predictions of each of these classifiers, Bagging seeks to reduce the error due to variance of the base classifier.

Methodology

We will look at Ensemble Learning using Bagging as our main algorithm to identify if a HPSA site labeled as designated is receiving the necessary assistance before getting withdrawn as HPSA. The motivation is to make sure all shortage areas are addressed and the correct resources are provided to the HPSA sites. To see if Bagging will work as the best algorithm, we are also exploring three other approaches, Decision Tree, Random Forest and Boosting, to run the algorithms on the HPSA dataset and compare them to the Bagging algorithm.

Datasets/pre-processing methods

The dataset used for this experiment is called Health Professional Shortage Areas covering the dental health, mental health and primary care disciplines. The dataset is a collection of 76,880 HPSA sites with 50 attributes classified by three class labels called Designated, Withdrawn and Proposed for Withdrawal. While cleaning the dataset for pre-processing, the 50 attributes were downsized to just 11 attributes. The thought process behind this feature selection is that most of the variables were id reference numbers linking to the HSPA sites and locations such as longitude and latitude, because of this, 40 attributes were omitted. Missing values were observed and these values were replace with methods shown in Table A1 in the appendix.

As mentioned earlier, there were 3 class labels but Proposed for Withdrawal accounted for less than 1% of the class labels. Due to the unbalance nature of the class labels and with the idea that the HPSA sites labeled as "Proposed for Withdrawal" is in the process of applying for withdrawal as a shortage area, the Proposed for Withdrawal class was merged with the Withdrawn class as this will make our algorithm run smoother with binary classifiers. Now with the new two class labels, the proportion between the two classes are 57% Designated to 43% Withdrawn. To further balance the class labels to get a better accuracy, the unbalance package in R was used to output a "balanced" dataset by resampling to generate an even class distribution.

Cross-Validation

With the new balance HPSA data set at 76,880 instances and 11 attributes, the goal is to predict the two classes (Designated and Withdrawn). We used two different classifiers to create inductive models using the sampled data and the chosen features. Cross-validation refers to a technique used to allow for the training and testing of inductive models without resorting to using the same dataset. First, we split the data into two groups: a training set (used to train the classifier) and a test set (used to estimate the error rate of the trained classifier). To do this, the createDataPartition function is used: By default, createDataPartition does a stratified random split of the data. To partition the data: The type of resampling used. The simple bootstrap is used by default with caret function in R. We will have the function use three repeats of 10–fold cross–validation. In this project we use ten-fold cross-validation. Additionally, In the process of performing all ensemble learner methods on every training dataset generated of 10-fold cross validation. The classification performance of each model is evaluated using the Area under the Receiver Operating Characteristic Curve (AUC).To reduce confusion, we use AUC when referring to the performance metric. To modify the resampling method, a trainControl function is used. The option method controls the type of resampling. Method, "repeatedcv", is used to specify repeated K–fold cross–validation (and the argument repeats controls the number of repetitions). K is controlled by the number argument and defaults to 10.The latter we computed measures specific to two–class problems, such as the area under the ROC curve, the sensitivity and specificity. Since the ROC curve is based on the predicted class probabilities.

Classification and Performance Metric

Our Bagging approach applied decision tree as its method because of this we choose to explore other similar algorithms that uses decision tree as its based method. Decision tree uses a top-down strategy with a recursive divide and conquer approach.

<u>Classification Trees</u>

The basic classification tree algorithm uses recursive partition to split a node that learns the best prediction and will stop when it can't learn any new prediction. In out R code, we choose to use the rpart method to create recursive partitioning for our classification labels.

Random Forest

Random Forest is an ensemble learner that is based on the bagging technique that combines learning models to increase accuracy by building multiple trees. In which it works as a large decorrelated decision trees. In turns the name "Forest" is acquired from the fact that multiple decision trees are used to construct the model. In our experiment we used the caret function to use cross validation to select the number of the predictors.

Stochastic Gradient Boosting

The gbm package in R was used to run the gradient boost algorithm, which is an extension of Freund and Schaprie's AdaBoost and Friedman's gradient boosting algorithm.

Results

In our experiment we focused on the Ensemble Leaner using Bagging approach. We compared it to 3 other classifiers and from the accuracies in table 1, Bagging was among one of the better performer with 99.4% accuracy. However, Random Forest slightly had a higher accuracy at 99.5. Both of these approaches performed very well. Looking at the ROC curves in figure 2, both Bagging and Random Forest was almost close to looking perfect.

Accuracy	Classification Tree	Random Forest	Boosting	Bagging
Train	0.923	0.995	0.940	0.999
Test	0.923	0.995	0.938	0.994

Table 1. Accuracy of the 4 classifiers

It was not surprising to see that Classification Tree had the lowest accuracy of the four, although it is not consider a bad accuracy, the overall model had a much lower specificity at 85% that using this model to identify the HPSA site is not recommended. Table A3 in the appendix recorded the sensitivity, specificity and confidence interval of the four approaches. Both approaches with the higher accuracies also showed better performance in sensitivity and specificity.

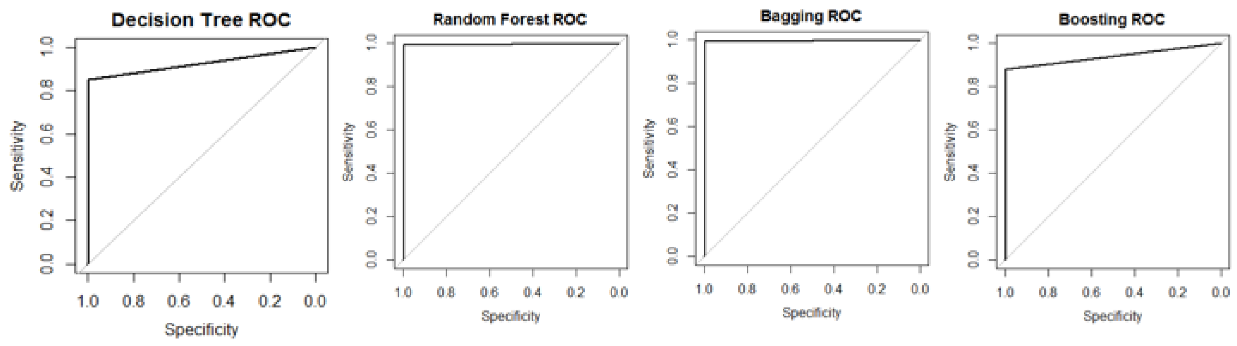

Figure 2. Area under the curve (ROC) chart of the four classifiers

Of the four approaches, Boosting fell off the radar with its bad performer, where it tails the Random Forest and Bagging algorithms. Considering the complexity that Boosting is able to provide, we even used three fold cross validation to test the model and it comes out short in its capability as an ensemble learner.

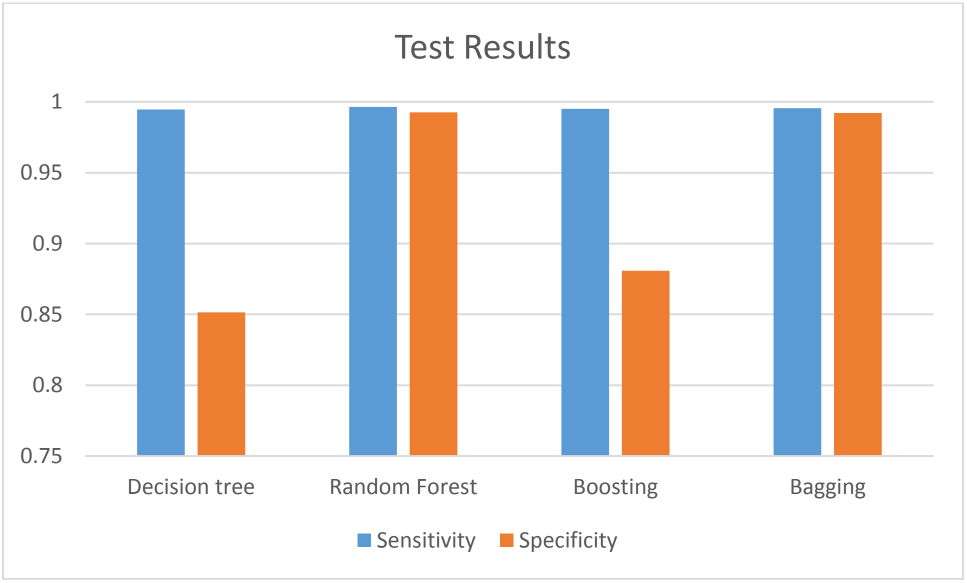

Figure 3. Bar chart of the sensitivity and specificity from the 4 classifiers of the test set

Conclusion

Ensembles are well established as a method for obtaining highly accurate classifiers by combining less accurate ones. We used a set of prediction tools (Tree, Bagging, Boosting and Forest). Tree is a nonlinear method and serves as the building block for the other three tools. The drawback of Tree is that it is instable (sensitive to data noise) and has high variance in the prediction, therefore, it wasn't surprising that Tree had the lowest accuracy of the four algorithms. Bagging reduces such variance by bootstrapping the samples. In contrast, Boosting can be think of as a bias reduction tool. We recommend the Random Forest for routine prediction tasks along with our Ensemble Leaner using Bagging. It wasn't surprising that both Random Forest and Bagging approaches had the highest accuracies as they are higher complexity algorithms that would be able to handle the HPSA dataset. However, it was surprising that Boosting didn't perform as well as the Random Forest and Bagging. Both the Classification Tree and Boosting had very similar results with a high sensitivity but and lower specificity indicating that HPSA sites that were labeled as Withdrawn may of not be ready to be taken off the Designated shortage list as the appropriate resources may of not been met as displayed in figure 3. Now, this tells us that both of the lower accuracy classifiers shouldn't be use to identity the HPSA shortage sites.

References

Breiman, L. (1996). "Bagging predictors." Machine Learning 26(2), 123–140. DOI: 10.1007/BF00058655 1, 1.1, 4.3

Breiman, L. (2001). Random Forests, random features. Berkeley: University of California. 1.1, 4.4

Dietterich T G An experimental comparison of three methods for constructing ensembles of decision trees Bagging boosting and randomization Machine Learning

P. Buhlmann and B. Yu. Analyzing bagging. ¨ The Annals of Statistics, 30:927–961, 2002. A. Buja and W. Stuetzle. Observations on bagging. Statistica Sinica, 16:323–352, 2006

T.G. Dietterich. Ensemble methods in machine learning. In J. Kittler and F. Roli, editors, First International Workshop on Multiple Classifier Systems, Lecture Notes in Computer Science, pages 1–15, New York, 2000. Springer-Verlag.

Appendix

Variable Names	Type	Missing Count
HPSA Status Description	Class variable	0
HPSA Type Description	Categorical value --> changed to dummy variable	0
HPSA Designation Population	Numeric value	7292
HPSA FTE	Numeric value	7828
HPSA Designation Population	Categorical value --> changed to dummy variable	643
HPSA Score	Numeric value	593
HPSA Shortage	Numeric value	21991
Discipline Class Description	Categorical value --> changed to dummy variable	0
HPSA Component Type Description	Categorical value --> changed to dummy variable	0
Break Designation	Categorical value --> changed to dummy variable	706
HPSA Population Type Description	Categorical value --> changed to dummy variable	11566
HPSA Provider Ratio Goal	Categorical value --> changed to dummy variable	5312

Table A1. Pre-processing of missing values for HPSA dataset

Dataset	Class Labels	Instances	Attributes	Missing Values	Balance
HPSA	3	76,880	50	Yes	No
Cleaned HPSA	2	87,168	12	No	Yes

Table A2. Comparison of the original to the cleaned dataset

Test Results	Sensitivity	Specificity	95% CI
Decision tree	0.9946	0.8515	(0.9197, 0.9262)
Random Forest	0.9964	0.9926	(0.9935, 0.9954)
Boosting	0.9951	0.8809	(0.9350, 0.9409)
Bagging	0.9956	0.9922	(0.9929, 0.9948)

Table A3. Sensitivity and Specificity values

Decision Tree

		Actual		
Train		Designated	Withdrawn	
Pred	Designated	30,377	4,541	34,918
	Withdrawn	132	25,968	26,100
		30,509	30,509	61,018
			Accuracy	0.923

		Actual		
Test		Designated	Withdrawn	
Pred	Designated	13,004	1,942	14,946
	Withdrawn	71	11,133	11,204
		13,075	13,075	26,150
			Accuracy	0.923

Random Forest

		Actual		
Train		Designated	Withdrawn	
Pred	Designated	30,414	226	30,640
	Withdrawn	95	30,283	30,378
		30,509	30,509	61,018
			Accuracy	0.995

		Actual		
Test		Designated	Withdrawn	
Pred	Designated	13,028	97	13,125
	Withdrawn	47	12,978	13,025
		13,075	13,075	26,150
			Accuracy	0.9945

Boosting

		Actual		
Train		Designated	Withdrawn	
Pred	Designated	30,379	3,554	33,933
	Withdrawn	130	26,955	27,085
		30,509	30,509	61,018
			Accuracy	0.940

		Actual		
Test		Designated	Withdrawn	
Pred	Designated	13,011	1,557	14568
	Withdrawn	64	11,518	11582
		13075	13075	26150
			Accuracy	0.938

Bagging

		Actual		
Train		Designated	Withdrawn	
Pred	Designated	30,453	35	30,488
	Withdrawn	56	30,474	30,530
		30,509	30,509	61,018
			Accuracy	0.999

		Actual		
Test		Designated	Withdrawn	
Pred	Designated	13018	102	13120
	Withdrawn	57	12973	13030
		13075	13075	26150
			Accuracy	0.9939

Table A4. Confusion matrix of algorithm approaches

R Code

```
library(randomForest)
library(MASS)
library(caret)
#library(DMwR)

HPSA.raw.data_clean_dummy <- read.csv("HPSA_clean_dummy.csv")
head(HPSA.raw.data_clean_dummy)

table(HPSA.raw.data_clean_dummy$HPSA.Status.Description)

### Balance Internet Ads dataset (386 ad to 1983 nonad)
library(unbalanced)

#balance Test Set
x=HPSA.raw.data_clean_dummy[,3:13]
head(x)
y=HPSA.raw.data_clean_dummy[,2]
head(y)

data <- ubOver(x,y, k = 0, verbose=TRUE)
hpsa_balance <- cbind(data$X, data$Y)
head(hpsa_balance)

write.csv(hpsa_balance, file='Internet_Ads_balance')
#-----------------------------------------------

#load balance dataset
HPSA.raw.data_clean_dummy <- read.csv("HPSA_clean_dummy_balance.csv")
head(HPSA.raw.data_clean_dummy)

#Create Training and Testing set
set.seed(123)
split <- createDataPartition(y=HPSA.raw.data_clean_dummy$HPSA.Status.Description, p = 0.7,
list=FALSE)
train <- HPSA.raw.data_clean_dummy[split,]
test<- HPSA.raw.data_clean_dummy[-split,]

library(Amelia)
missmap(test, main = "Missingness Map Test")
```

```
#Classification tree
library(rattle)
library(rpart)
library(rpart.plot)

##  regression tree model
set.seed(124)
Mod0 <- train(HPSA.Status.Description ~ .,data=train, method="rpart")
save(Mod0,file="Mod0.RData")

load("Mod0.RData")
fancyRpartPlot(Mod0$finalModel)

Mod0
Mod0$finalModel

#accuracy for Training set
pred0_train <- predict(Mod0,train)
cm0_train <- confusionMatrix(pred0_train, train$HPSA.Status.Description)
cm0_train$overall
cm0_train

#accuracy for Test set
pred0 <- predict(Mod0, test)
cm0 <- confusionMatrix(pred0, test$HPSA.Status.Description)
cm0$table
cm0


#Plot ROC Curve
library(pROC)
#convert pred0 to numeric
pred0 <- ifelse(pred0=='Designated',0,1)

auc0 <- roc(test$HPSA.Status.Description, pred0)
print(auc0)

plot(auc0, main = 'Decision Tree ROC')
```

```
##Check balance
#bal0 <- SMOTE(HPSA.Status.Description ~ ., data=train, perc.over = 100,
#              perc.under=200)

#table(bal0$HPSA.Status.Description)

#dim(train)

#bal_mod0 <- train(HPSA.Status.Description ~ .,data=bal0, method="rpart")
#bal_pred0 <- predict(bal_mod0, test)
#bal_cm0 <- confusionMatrix(bal_pred0, test$HPSA.Status.Description)
#bal_cm0

#bal_pred0 <- ifelse(bal_pred0=='Designated',0,1)
#bal_acu0 <- roc(test$HPSA.Status.Description, bal_pred0)
#print(bal_acu0)

#library(knitr)
#kable(cm0$table)

# random forest model
set.seed(125)
Mod1 <- train(HPSA.Status.Description ~ ., method = "rf", data = train, importance =
T,trControl = trainControl(method = "cv", number = 10))
Mod1
Mod1$finalModel
vi <- varImp(Mod1)
vi$importance[1:10,]

#accuracy for Training set
pred1_train <- predict(Mod1,train)
cm1_train <- confusionMatrix(pred1_train, train$HPSA.Status.Description)
cm1_train$overall
cm1_train

# out-of-sample errors of random forest model using validation dataset
pred1 <- predict(Mod1, test)
cm1 <- confusionMatrix(pred1, test$HPSA.Status.Description)
cm1
```

```
# plot roc curves
library(pROC)

#convert pred0 to numeric
pred1 <- ifelse(pred1=='Designated',0,1)

auc1 <- roc(test$HPSA.Status.Description, pred1)
plot(auc1, main = 'Random Forest ROC')

pred1.prob <- predict(Mod1, test, type="prob")
pred1.prob$
#roc1 <-  roc(val$total_accel_belt, pred1.prob$E)
# plot(roc1, print.thres="best", print.thres.best.method="closest.topleft")
# coord1 <- coords(roc1, "best", best.method="closest.topleft",
#                          ret=c("threshold", "accuracy"))
# coord1
# summary of final model
Mod1$finalModel
plot(Mod1)
plot(Mod1$finalModel)
plot(varImp(Mod1), top = 10)


library(gbm)
#Boosting
set.seed(126)
Mod2 <- train(HPSA.Status.Description ~ ., method = "gbm", data = train,  verbose = F,
trControl = trainControl(method = "cv", number = 10))
Mod2
plot(Mod2)

#accuracy for Training set
pred2_train <- predict(Mod2, train)
cm2_train <- confusionMatrix(pred2_train, train$HPSA.Status.Description)
cm2_train

#accuracy for Test set
pred2 <- predict(Mod2, test)
cm2 <- confusionMatrix(pred2, test$HPSA.Status.Description)
```

```
cm2
cm2$overall

plot(Mod2$finalModel)
plot(varImp(Mod1),top=10)

# plot roc curves
library(pROC)
#convert pred2 to numeric
pred2 <- ifelse(pred2=='Designated',0,1)

auc2 <- roc(test$HPSA.Status.Description, pred2)
plot(auc2, main = 'Boosting ROC')

## model tuning
gbmGrid <- expand.grid(.interaction.depth=(1:3)*2, .n.trees=(1:5)*20, .shrinkage=.1)
bootControl <- trainControl(number=50)

set.seed(2)
gmbFit<- train(HPSA.Status.Description ~ .,method = "gbm",data = train,verbose = F,trControl
= bootControl,bag.fraction=0.5,tuneGrid=gbmGrid)
plot(gmbFit)
plot(gmbFit,plotType = "level")
resampleHist((gmbFit))

# out-of-sample errors using validation dataset
predgmb <- predict(gmbFit, val)
cmgmb <- confusionMatrix(pred2, val$classe)
cmgmb$overall

##Bagging
set.seed(127)
Mod3 <- train(HPSA.Status.Description ~ .,data=train,method="treebag")
Mod3
Mod3$finalModel

#accuracy for Training set
pred3_train <- predict(Mod3,train)
cm3_train <- confusionMatrix(pred3_train, train$HPSA.Status.Description)
cm3_train$overall
```

```
cm3_train

#accuracy for Test set
pred3 <- predict(Mod3,test)
cm3 <- confusionMatrix(pred3, test$HPSA.Status.Description)
cm3$overall
cm3

varImp(Mod3)
plot(varImp(Mod3), top = 10)

# plot roc curves
library(pROC)
#convert pred2 to numeric
pred3 <- ifelse(pred3=='Designated',0,1)

auc3 <- roc(test$HPSA.Status.Description, pred3)
plot(auc3, main = 'Bagging ROC')

#Prediction Model Selection
re <- data.frame(Tree=cm0$overall[1], rf=cm1$overall[1],
boosting=cm2$overall[1],bagging=cm3$overall[1])
library(knitr)
re
```