Stats 315B: Homework 1

Joe Higgins, Austin Wang, Jessica Wetstone

Due 5/19/2018

Question 1

Random forests predict with an ensemble of bagged trees each trained on a bootstrap sample randomly drawn from the original training data. Additional random variation among the trees is induced by choosing the variable for each split from a small randomly chosen subset of all of the predictor variables when building each tree. What are the advantages and disadvantages of this random variable selection strategy? How can one introduce additional tree variation in the forest without randomly selecting subsets of variables?

The advantages of choosing a random subset of predictor variables are:

- Because we are no longer doing an exhaustive search over all predictor variables, each split requires less processing power during the training process. Given modern computer architecture, this advantage matters less since each tree in the ensemble is independent and can be computed simultaneously in a different parallel process.
- The random variation introduced by choosing the variable for each split from a randomly chosen subset de-correlates the resulting trees in the forest, forcing them to be different from each other. This has the effect of decreasing the variance of the ensemble's estimate, since the variance of the mean of a collection random variables is directly related to the correlation of those variables. Decreasing the correlation of the trees therefore decreases the variance of the forest.

The disadvantage of this approach is that it can result in trees with very weak predictive power. Suppose that none of the best predictors were available for a given tree's first several splits – this would result in a highly sub-optimal tree.

Another way to inject randomness into the process and de-correlate the trees is to decrease the size of the bootstrapped sample of the training data. Each individual tree is now more likely to overfit the smaller sample, increasing tree variation within the forest.

Question 2

Why is it necessary to use regularization in linear regression when the number of predictor variables is greater than the number of observations in the training sample? Explain how regularization helps in this case. Are there other situations where regularization might help? What is the potential disadvantage of introducing regularization? Why is sparsity a reasonable assumption in the boosting context. Is it always? If not, why not?

If the number of predictor variables n is greater than the number of observations in the training sample N, then there is an infinite number of solutions that will satisfy the linear regression problem – and there will be very high variance in the solution derived from that training sample. Regularization helps in this situation because it decreases the size of the function class, reducing variance. Regularization will help in any situation where we have a reason to favor sparse solutions by both encouraging sparsity and preventing high variance.

A potential disadvantage of regularization is that it could increase bias - especially if you have chosen a penalty that doesn't align with the best answer for the problem.

In the boosting context, each predictor is a possible tree that could be built on the training data from the set of all possible trees. It is reasonable to assume that most trees - out of the set of all possible trees (All combinations of variables and all possible splits) - will do very poorly on the overall prediction problem. So we would want most of our coefficients in the linear regression to be 0.

Sparsity is not always a necessary assumption in linear regression, especially if your feature space is small. For example, if we were predicting house prices from zip code and square footage, we would expect both variables to be important.

Question 3

Show that the convex members of the power family of penalties, except for the lasso, have the property that solutions to $\hat{a}(\lambda) = argmin_a \hat{R}(a) + \lambda P_\gamma(a)$ have nonzero values for all coefficients at each path point indexed by λ. By contrast the convex members of the elastic net (except ridge) can produce solutions with many zero valued coefficients at various path points.

Question 4

Show that the variable x_{j^*} that has the maximum absolute correlation with $j^* = argmax_{1 \le j \le J} |E(yx_j)|$ is the same as the one that best predicts y using squared—error loss $j^* = argmin_{1 \le j \le J}\ min_\rho E[y - \rho x_j]^2$. This shows that the base learner most correlated with the generalized residual is the one that best predicts it with squared—error loss.

Let $f(\rho,\ x_j,\ y) = E[y - \rho x_j]^2$. We first find $min_\rho\ f(\rho,\ x_j,\ y)$ by taking the partial derivative of f with respect to ρ and setting it equal to 0:

$$\begin{aligned}
f(\rho,\ x_j,\ y) &= E[y - \rho x_j]^2 \\
&= E[y^2 - 2\rho y x_j + \rho^2 x_j^2] \\
&= E[y^2] - 2\rho E[yx_j] + \rho^2 E[x_j^2] \\
&= E[y^2] - 2\rho E[yx_j] + \rho^2 \qquad \text{since } E[x_j^2] = 1
\end{aligned}$$

$$\begin{aligned}
\implies \frac{\partial f}{\partial \rho} &= -2E[yx_j] + 2\rho \\
\implies 0 &= -2E[yx_j] + 2\rho^* \\
\implies \rho^* &= E[yx_j]
\end{aligned}$$

$$\begin{aligned}
\implies min_\rho\ f(\rho,\ x_j,\ y) &= f(\rho^*,\ x_j,\ y) \\
&= E[y - \rho^* x_j]^2 \\
&= E[y^2] - 2\rho^* E[yx_j] + {\rho^*}^2 \\
&= E[y^2] - 2E[yx_j]E[yx_j] + (E[yx_j])^2 \\
&= E[y^2] - (E[yx_j])^2
\end{aligned}$$

Note that this is a unique minimum value because f is strictly convex in ρ. We know this because $\frac{\partial^2 f}{\partial \rho^2} > 0$:

$$\begin{aligned} \frac{\partial f}{\partial \rho} &= -2E[yx_j] + 2\rho \\ \implies \frac{\partial^2 f}{\partial \rho^2} &= 2 \\ &> 0 \end{aligned}$$

Then:

$$\begin{aligned} argmin_{1\leq j\leq J}\ min_\rho E[y-\rho x_j]^2 &= argmin_{1\leq j\leq J}\ (E[y^2] - (E[yx_j])^2) \\ &= argmin_{1\leq j\leq J}\ -(E[yx_j])^2 \\ &= argmax_{1\leq j\leq J}\ (E[yx_j])^2 \\ &= argmax_{1\leq j\leq J}\ |E[yx_j]|^2 \\ &= argmax_{1\leq j\leq J}\ |E[yx_j]| \qquad \text{since } f(x)=\sqrt{x} \text{ is monotonically increasing for } x>0 \end{aligned}$$

Therefore, $argmax_{1\leq j\leq J}\ |E[yx_j]| = argmin_{1\leq j\leq J}\ min_\rho E[y-\rho x_j]^2$, so the base learner most correlated with the generalized residual is the one that best predicts it with squared—error loss. ■

Question 5

Let $\mathbf{z}_l = \{z_1, ..., z_l\}$ be a subset of the predictor variables $\mathbf{x} = \{x_1, ..., x_n\}$ and $\mathbf{z}_{\backslash l}$ the complement subset, i.e. $\mathbf{z}_l \cup \mathbf{z}_{\backslash l} = \mathbf{x}$. Show that if a function $F(\mathbf{x})$ is additive in $\mathbf{z}_l$ and $\mathbf{z}_{\backslash l}$, i.e.

$$F(\mathbf{x}) = F_l(\mathbf{z}_l) + F_{\backslash l}(\mathbf{z}_{\backslash l})$$

then the partial dependence of $F(\mathbf{x})$ on $\mathbf{z}_l$ is $F_l(\mathbf{z}_l)$ up to an additive constant. This is the dependence of $F(\mathbf{x})$ on $\mathbf{z}_l$ accounting for the effect of the other variables $\mathbf{z}_{\backslash l}$. Show that this need not be the case for $E[F(\mathbf{x}) \mid \mathbf{z}_l]$ which is the dependence of $F(\mathbf{x})$ on $\mathbf{z}_l$ ignoring the other variables $\mathbf{z}_{\backslash l}$. Under what conditions would the two be the same?

Question 6

Binary classification: Spam Email. The data set for this problem is spam_stats315B.csv, with documentation files spam_stats315B_info.txt and spam_stats315B_names,txt. The data set is a collection of 4601 emails of which 1813 were considered spam, i.e. unsolicited commercial email. The data set consists of 58 attributes of which 57 are continuous predictors and one is a class label that indicates whether the email was considered spam (1) or not (0). Among the 57 predictor attributes are: percentage of the word "free" in the email, percentage of exclamation marks in the email, etc. See file spam_stats315B_names.txt for the full list of attributes. The goal is, of course, to predict whether or not an email is "spam". This data set is used for illustration in the tutorial Boosting with R Programming. The data set spam_stats315B_train.csv represents a subsample of these emails randomly selected from spam_stats315B.csv to be used for training. The file spam_stats315B_test.csv contains the remaining emails to be used for evaluating results.

(a) Based on the training data, fit a gbm model for predicting whether or not an email is "spam", following the example in the tutorial. What is your estimate of the misclassification rate? Of all the spam emails of the test set what percentage was misclassified, and of all the non-spam emails in the test set what percentage was misclassified?

```r
rm(list = ls())
data_path <- paste(getwd(),'/data',sep='')
setwd(data_path)

#data labels
rflabs<-c("make", "address", "all", "3d", "our", "over", "remove",
  "internet","order", "mail", "receive", "will",
  "people", "report", "addresses","free", "business",
  "email", "you", "credit", "your", "font","000","money",
  "hp", "hpl", "george", "650", "lab", "labs",
  "telnet", "857", "data", "415", "85", "technology", "1999",
  "parts","pm", "direct", "cs", "meeting", "original", "project",
  "re","edu", "table", "conference", ";", "(", "[", "!", "$", "#",
  "CAPAVE", "CAPMAX", "CAPTOT","type")

#load the data
train <- read.csv(file="spam_stats315B_train.csv", header=FALSE, sep=",")
test <- read.csv(file="spam_stats315B_test.csv", header=FALSE, sep=",")
colnames(train)<-rflabs
colnames(test)<-rflabs

#randomize order of data
set.seed(131)
x <- train[sample(nrow(train)),]

#fit model on training data
set.seed(444) # random for bag.fraction
gbm0 <- gbm(type~., data=x, train.fraction=1,
            interaction.depth=4, shrinkage=.05,
            n.trees=2500, bag.fraction=0.5, cv.folds=5,
            distribution="bernoulli", verbose=T)

#estimate of misclassification rate
```

```r
y_hat_train <- predict(gbm0, x, type="response", n.trees=300)
y_hat_train[y_hat_train > 0.5] <- 1
y_hat_train[y_hat_train <= 0.5] <- 0
y <- x$type
correct <- y == y_hat_train
pct_correct <- sum(correct)/length(correct)

#What is your estimate of the misclassification rate?
1-pct_correct

#misclassification rate test
y_hat_test <- predict(gbm0, test, type="response", n.trees=300)
y_hat_test[y_hat_test > 0.5] <- 1
y_hat_test[y_hat_test <= 0.5] <- 0
y <- test$type
correct <- y == y_hat_test

#Of all the spam emails of the test set what percentage was misclassified?
spam_correct <- correct[y==1]
pct_spam_correct <- sum(spam_correct)/length(spam_correct)
1-pct_spam_correct

#Of all the non-spam emails in the test set what percentage was misclassified?
nonspam_correct <- correct[y==0]
pct_nonspam_correct <- sum(nonspam_correct)/length(nonspam_correct)
1-pct_nonspam_correct
```

(b) Your classifier in part (a) can be used as a spam filter. One of the possible disadvantages of such a spam filter is that it might filter out too many good (non-spam) emails. Therefore, a better spam filter might be the one that penalizes misclassifying non-spam emails more heavily than the spam ones. Suppose that you want to build a spam filter that "throws out" no more that 0.3% of the good (non-spam) emails. You have to find and use a cost matrix that penalizes misclassifying "good" emails as"spam" more than misclassifying "spam" emails as "good" by the method of trial and error. Once you have constructed your final spam filter with the property described above, answer the following questions:

```r
w <- x$type
w[w == 1] <- 9
w[w == 0] <- 1

gbm1 <- gbm(type~., data=x , train.fraction=1,
            interaction.depth=4, shrinkage=.05,
            n.trees=2500, bag.fraction=0.5, cv.folds=5,
            distribution="bernoulli", verbose=T,
            weights = w)

#misclassification rate on training set
y_hat_train <- predict(gbm1, x, type="response", n.trees=300)
y_hat_train[y_hat_train > 0.5] <- 1
y_hat_train[y_hat_train <= 0.5] <- 0
y <- x$type
correct_train <- y == y_hat_train
```

```r
#Of all the spam emails of the test set what percentage was misclassified?
spam_correct_train <- correct_train[y==1]
pct_spam_correct_train <- sum(spam_correct_train)/length(spam_correct_train)
1-pct_spam_correct_train

#Of all the non-spam emails in the test set what percentage was misclassified?
nonspam_correct_train <- correct_train[y==0]
pct_nonspam_correct_train <- sum(nonspam_correct_train)/length(nonspam_correct_train)
1-pct_nonspam_correct_train
```

(i) What is the overall misclassification error of your final filter and what is the percentage of good emails and spam emails that were misclassified respectively?

```r
#misclassification rate on training set
y_hat_test <- predict(gbm1, test, type="response", n.trees=300)
y_hat_test[y_hat_test > 0.5] <- 1
y_hat_test[y_hat_test <= 0.5] <- 0
y <- test$type
correct_test <- y == y_hat_test
spam_correct_test <- correct_test[y==1]
nonspam_correct_test <- correct_test[y==0]

#Overall misclassification:
pct_correct_train <- sum(correct_train)/length(correct_train)
pct_correct_test <- sum(correct_test)/length(correct_test)
1-pct_correct_train
1-pct_correct_test

#Spam misclassification:
pct_spam_correct_test <- sum(spam_correct_test)/length(spam_correct_test)
1-pct_spam_correct_train
1-pct_spam_correct_test

#Nonspam misclassification:
pct_nonspam_correct_test <- sum(nonspam_correct_test)/length(nonspam_correct_test)
1-pct_nonspam_correct_train
1-pct_nonspam_correct_test
```

(ii) What are the important variables in discriminating good emails from spam for your spam filter?

```r
top5 <-summary(gbm1,main="RELATIVE INFLUENCE OF ALL PREDICTORS")$var[1:5]
```

(iii) Using the interpreting tools provided by gbm, describe the dependence of the response on the most important attributes.

```r
best.iter <- gbm.perf(gbm1,method="OOB")#Best iteration by OOB
top5 <- gsub("`","",top5)
top5_indexes <- match(top5, rflabs)
```

```r
for(i in top5_indexes){
  plot(gbm1,i,best.iter)
}
```

Question 7

Regression: California Housing. The data set calif_stats315B.csv consists of aggregated data from 20,640 California census blocks (from the 1990 census). The goal is to predict the median house value in each neighborhood from the others described in calif_stats315B.txt. Fit a gbm model to the data and write a short report that should include at least

(a) The prediction accuracy of gbm on the data set.

```r
rm(list = ls())
data_path <- paste(getwd(),'/data',sep='')
setwd(data_path)

#data labls
labels <-c(
  "house_value",
  "median_income",
  "housing_median_age",
  "average_no_rooms",
  "average_no_bedrooms",
  "population",
  "average_occupancy",
  "latitude",
  "longitude"
)

#load the data
data <- read.csv(file="calif_stats315B.csv", header=FALSE, sep=",")
colnames(data) <- labels

#fit model on training data
set.seed(444) # random for bag.fraction
model <- gbm(house_value~., data=data, train.fraction=1,
             interaction.depth=4, shrinkage=.05,
             n.trees=2500, bag.fraction=0.5, cv.folds=5,
             distribution="gaussian", verbose=T)

y <- data$house_value
y_hat <- predict(model, data, type="response", n.trees=300)
MSE <- sum((y_hat - y)^2)/length(y)
```

(b) Identification of the most important variables.

```r
top4 <-summary(model,main="RELATIVE INFLUENCE OF ALL PREDICTORS")$var[1:4]
```

(c) Comments on the dependence of the response on the most important variables (you may want to consider partial dependence plots (plot) on single and pairs of variables, etc.).

```r
best.iter <- gbm.perf(model,method="OOB")#Best iteration by OOB
top4 <- gsub("`","",top4)
top4_indexes <- match(top4, labels) - 1 #-1 because first label is response var

#main effects
for(i in top4_indexes){
  plot(model,i,best.iter)
}

#longitude, latitude interaction effects
plot(model,c(8,7),best.iter)
```

Question 8

Regression: Marketing data. The data set age_stats315B.csv was already used in Homework 1. Review age_stats315B.txt for the information about order of attributes etc.

(a) Fit a gbm model for predicting age form the other demographic attributes and compare the accuracy with the accuracy of your best single tree from Homework 1.

```r
rm(list = ls())
data_path <- paste(getwd(),'/data',sep='')
setwd(data_path)

#Read and type data
age_data <- read.csv('age_stats315B.csv')
factor_columns <- c(
  'Occup',
  'TypeHome',
  'sex',
  'MarStat',
  'DualInc',
  'HouseStat',
  'Ethnic',
  'Lang'
)
age_data[factor_columns] <- lapply(age_data[factor_columns], as.factor)

#fit a GBM model
set.seed(444) # random for bag.fraction
model_gbm <- gbm(age~., data=age_data, train.fraction=1,
            interaction.depth=4, shrinkage=.05,
            n.trees=2500, bag.fraction=0.5, cv.folds=5,
            distribution="gaussian", verbose=T)

#fit a tree
model_tree <- rpart(age ~ ., data = age_data, method = "anova",
                    control=rpart.control(minbucket = 10,
                                            xval = 10,
                                            maxsurrogate = 5,
                                            usesurrogate = 2,
                                            cp=0.0001))

# Find the minimum cross-validation error + one SD
min_error_window <- min(model_tree$cptable[,"xerror"] + model_tree$cptable[,"xstd"])

# Find the simplest model with xerror within the min_error_window
best_cp <- first(model_tree$cptable[which(model_tree$cptable[,"xerror"] < min_error_window),"CP"])
best_single_tree <- prune(model_tree, cp = best_cp)

#Make predictions
y_hat_gbm <- predict(model_gbm, age_data, type="response", n.trees=300)
y_hat_tree <- predict(best_single_tree, age_data)

#Compare errors
y <- age_data$age
MSE_gbm <- sum((y_hat_gbm - y)^2)/length(y)
MSE_tree <- sum((y_hat_tree - y)^2)/length(y)

MSE_gbm
MSE_tree
```

(b) Identify the most important variables.

```
top4 <-summary(model_gbm,main="RELATIVE INFLUENCE OF ALL PREDICTORS")$var[1:4]
top4
```

Question 9

Multiclass classification: marketing data. The data set occup_stats315B.csv comes from the same marketing database used in Homework 1. The description of the attributes can be found in occup_stats315B.txt. The goal in this problem is to fit a gbm model to predict the type of occupation from the 13 other demographic variables.

(a) Report the test set misclassification error for gbm on the data set, and also the misclassification error for each class.

```
rm(list = ls())
data_path <- paste(getwd(),'/data',sep='')
setwd(data_path)

#Read and type data
labels <-c(
  "Occup",
  "TypeHome",
  "sex",
  "MarStat",
  "age",
```

```r
  "Edu",
  "Income",
  "LiveBA",
  "DualInc",
  "Persons",
  "Under18",
  "HouseStat",
  "Ethnic",
  "Lang"
)
factor_columns <- c(
  'TypeHome',
  'sex',
  'MarStat',
  'Occup',
  'LiveBA',
  'DualInc',
  'HouseStat',
  'Ethnic',
  'Lang'
)

occup_labels <- c(
  "Professional/Managerial",
  "Sales Worker",
  "Factory Worker/Laborer/Driver",
  "Clerical/Service Worker",
  "Homemaker",
  "Student, HS or College",
  "Military",
  "Retired",
  "Unemployed"
)

#load the data
house_data <- read.csv('occup_stats315B.csv', header=FALSE, sep=",")
colnames(house_data) <- labels
house_data[factor_columns] <- lapply(house_data[factor_columns], as.factor)

#fit GBM model for occupation
set.seed(444) # random for bag.fraction
model <- gbm(Occup ~ ., data=house_data, train.fraction=1,
            interaction.depth=4, shrinkage=.05,
            n.trees=2500, bag.fraction=0.5, cv.folds=5,
            distribution="multinomial", verbose=T)

#determine misclassification rate
y_hat <- predict(model, house_data, type="response", n.trees=300)
y_hat_max_p <- apply(y_hat,1,which.max)
y <- house_data$Occup
correct <- y == y_hat_max_p

#overall
```

```r
pct_correct <- sum(correct)/length(correct)
1-pct_correct

for(occupation in levels(house_data$Occup)){
  occupation_i <- strtoi(occupation)
  print(occup_labels[occupation_i])
  correct_in_class <- correct[y==occupation_i]
  pct_correct_in_class <- sum(correct_in_class)/length(correct_in_class)
  print(1-pct_correct_in_class)
}
```

(b) Identify the most important variables.

```r
top4 <-summary(model,main="RELATIVE INFLUENCE OF ALL PREDICTORS")$var[1:4]
top4
```