# Homework 4 Solutions

Alexander McLain

```
library(rmarkdown)
library(printr)
library(tidyverse)
library(dplyr)
library(rpart)
library(rpart.plot)
library(rminer)
library(kernlab)
library(randomForest)
library(gbm)
library(caret)
```

For this homework we're going to focus on the Indian Liver Patient Dataset. The data that we'll use here is different then what we used in our examples. You can read about the details of the predictors and outcome http://archive.ics.uci.edu/dataset/225/ilpd+indian+liver+patient+dataset.

Be sure to note that any patient whose age exceeded 89 is listed as being of age "90' ', for your functions below you will simply recode such people's ages as 92 (numeric). The complete data contains 583 observations, 416 patients diagnosed with liver disease and 167 patients without liver disease.

The data `ILPD.csv` is available on GitHub. You will use this data to fit all of your models. There is another dataset `ILPD_test.csv` which contains 125 observations that I removed (at random) from the complete dataset. This dataset is not available on the website.

You will answer the following questions based on this data. I recommend reading an understanding all questions before commencing with your analyses.

1. **(15 points)** Fit a CART to this data. Prune the tree accordingly (i.e., using CV). What appears to be the most important factors in your final model?
2. **(15 points)** Fit an SVM to the data. Choose the shape parameter of the kernal (or the two kernal parameters depending on what kernal you use) and the cost according to CV.
3. **(15 points)** Fit a random forest model to the data, appropriately using CV to tune the parameters. What three variables seem to matter the most?
4. **(15 points)** Fit a GBM model to the data, appropriately using CV to tune the parameters. Draw partial dependence plots for three variables you found to be the most important in the previous question.
5. **(15 points)** What model do you think works the best? For your best model, estimate what the value of $\frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2$ would be for a new test data set with size $N$. Give a standard error for this value and a 95% confidence interval, i.e., using $est \pm 1.96SE$.
6. **(25 points)** For this question, turn in a separate R program `lastname_firstname_Q5HW4.R` which:

- Uses the data `ILPD.csv` to train the final version of the model from question 5. That is, it should not run CV, just your final model with your final tuning parameters and the same seed (if applicable). It should use the entire `ILPD.csv` dataset.

- Reads in the data `ILPD_test.csv` and predicts for the model you felt worked the best. You will not have this dataset but I will.
- Creates a data.frame called `lastname.firstname_error` with the **mean squared error** from the test data from your best model. This data should have 3 columns in the following order:
- `name`: which is your name in the format '`lastname.firstname`',
- `best.model`: character variable with the name of the best model (`CART`, `SVM`, `RF`, or `GBM`), and
- `est.error`: the test error.

I will run your program on my computer. It must run with zero error messages and produce the desired data frame.

**Solutions**

Read in the data

```
ILPD <- read.csv("ILPD.csv")
# Recode ages greater than 89 to 92
ILPD <- ILPD %>%
  mutate(Age = ifelse(Age == 90, 92, Age)) %>%
  na.omit() %>%
  mutate(
    Selector = as.factor(Selector),
    Gender = as.factor(Gender)
  )
head(ILPD)
```

| Age | Gender | TB | DB | Alkphos | Sgpt | Sgot | TP | ALB | AG.Ratio | Selector |
|---|---|---|---|---|---|---|---|---|---|---|
| 65 | Female | 0.7 | 0.1 | 187 | 16 | 18 | 6.8 | 3.3 | 0.90 | 1 |
| 62 | Male | 10.9 | 5.5 | 699 | 64 | 100 | 7.5 | 3.2 | 0.74 | 1 |
| 58 | Male | 1.0 | 0.4 | 182 | 14 | 20 | 6.8 | 3.4 | 1.00 | 1 |
| 72 | Male | 3.9 | 2.0 | 195 | 27 | 59 | 7.3 | 2.4 | 0.40 | 1 |
| 26 | Female | 0.9 | 0.2 | 154 | 16 | 12 | 7.0 | 3.5 | 1.00 | 1 |
| 29 | Female | 0.9 | 0.3 | 202 | 14 | 11 | 6.7 | 3.6 | 1.10 | 1 |

I'm going to split the data, so that I can give an unbiased estimate of the prediction error in Question 5:

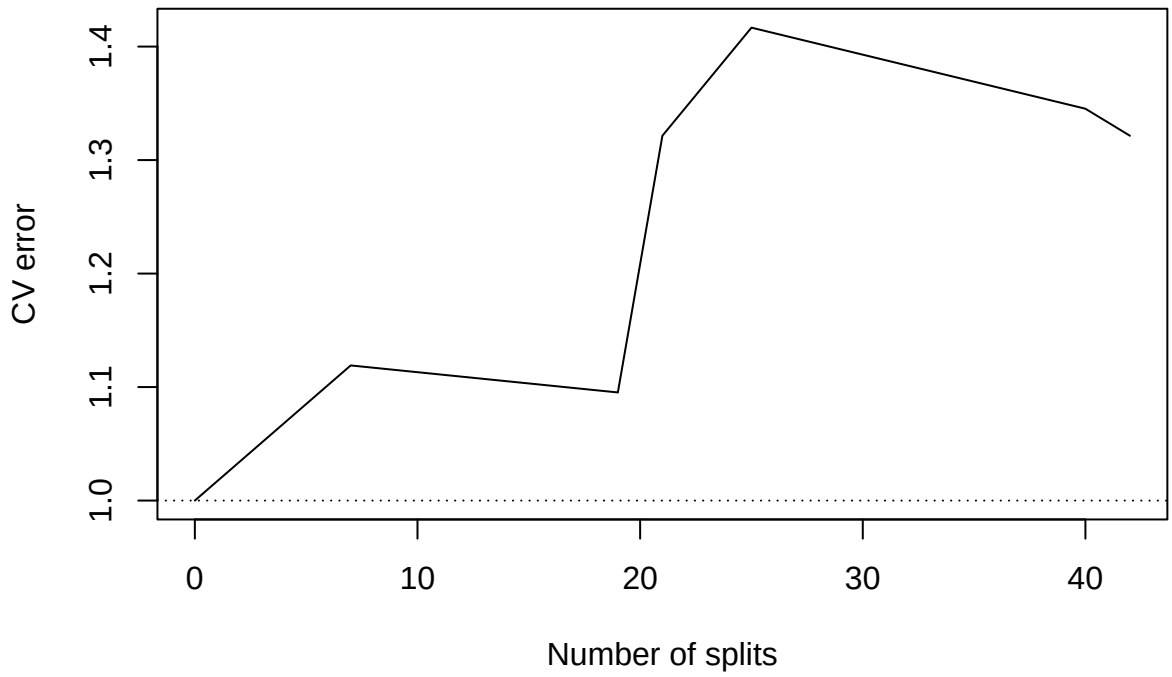
```
# Generate a random sample of row indices - stratified sampling
set.seed(12390)
n <- nrow(ILPD)
trainIndex <- sample(1:3, n, prob = c(0.6,0.2,0.2) , replace = TRUE)
table(trainIndex)
```

| 1 | 2 | 3 |
|---|---|---|
| 281 | 87 | 88 |

```
# Create training and test sets
ILPD_train <- ILPD[trainIndex == 1, ]
ILPD_vald <- ILPD[trainIndex == 2, ]
ILPD_test <- ILPD[trainIndex == 3, ]
```

1. Now let's fit a CART to the data. Here I decrease the "cp" criteria in the control so that it will fit the tree further down (then by default).

```
set.seed(456546)
fit.ilpd <- rpart(Selector ~ ., data = ILPD_train, method = "class",
                  control = list(minsplit = 5, minbucket = 1,
                                 cp = 0, xval = 100))

Fit_table.ilpd <- as.data.frame(fit.ilpd$cptable)
plot(Fit_table.ilpd$nsplit,Fit_table.ilpd$xerror,
     xlab = "Number of splits",ylab="CV error",
     type = "l", xlim=c(0,min(c(max(Fit_table.ilpd$nsplit),100))))
abline(h=1,lty=3)
```




The model with 1 splits has the lowest CV error. This indicates that the CART doesn't perform well (for me). I'm not going to use it going forward.

2. Now we'll fit an SVM to the data.

I'm going to use the `fit` function from the `rminer` package which will call `ksvm` from the `kernlab` package to fit the SVM. I'm gooing to use the `rbfdot` Radial Basis kernel, i.e., the "Gaussian" with kernel $K(u, v) = \exp(-(u-v)^2/\sigma^2)$. To fit, $\sigma^2$ and $C$ (the cost) will be varied over various values.

```
set.seed(5413213)
M.CV <- fit(
```

```
  Selector ~ .,
  data = ILPD_train,
  model = "svm",
  task = "class", #  class
  search = list(
    smethod = "grid",
    search = list(
      sigma = c(1e-10 ,1e-05, 5e-04, 1e-04, 5e-03, 1e-03, 0.01),
      C = c(1e-10 , 1e-5, 1e-4, 1e-3, 0.001, 0.01, 0.1, 1, 1.5, 2, 3)
    ),
    convex = 0,
    method = c("kfold", 10),
    metric = "CE"
  ),
  scale = "inputs"
)
```

Let's look at some summaries of the optimal values:

```
M.CV@mpar
```

```
## $kpar
## $kpar$sigma
## [1] 1e-10
##
##
## $C
## [1] 1e-10
##
## $task
## [1] "class"
```

```
M.CV@object
```

```
## Support Vector Machine object of class "ksvm"
##
## SV type: C-svc  (classification)
##  parameter : cost C = 1e-10
##
## Gaussian Radial Basis kernel function.
##  Hyperparameter : sigma =  1e-10
##
## Number of Support Vectors : 168
##
## Objective Function Value : 0
## Training error : 0.298932
```

Note that 168 out of the 281 observations (60%) are a support vector. Later, we can use `predict.fit` in the `rminer` package to predict for the validation and test data.

3. Now we'll fit an Random Forests to the data. Here, i'll estimate the test and oob loss to select the best model. I'm going to use 2000 trees, which should be plenty.

```r
num_vars <- 1:10
oob.loss <- NULL
for(m in num_vars) {
  set.seed(5413213)
  RF_Liver = randomForest(
    Selector ~ .,
    data = ILPD_train,
    ntree = 2000,
    mtry = m
  )
  yhat.oob <- RF_Liver$predicted
  oob.loss <- c(oob.loss, mean(yhat.oob != ILPD_train$Selector))
}
data.frame(num_vars, oob.loss)
```

| num_vars | oob.loss |
|---|---|
| 1 | 0.3024911 |
| 2 | 0.3274021 |
| 3 | 0.3380783 |
| 4 | 0.3309609 |
| 5 | 0.3523132 |
| 6 | 0.3487544 |
| 7 | 0.3558719 |
| 8 | 0.3558719 |
| 9 | 0.3558719 |
| 10 | 0.3487544 |

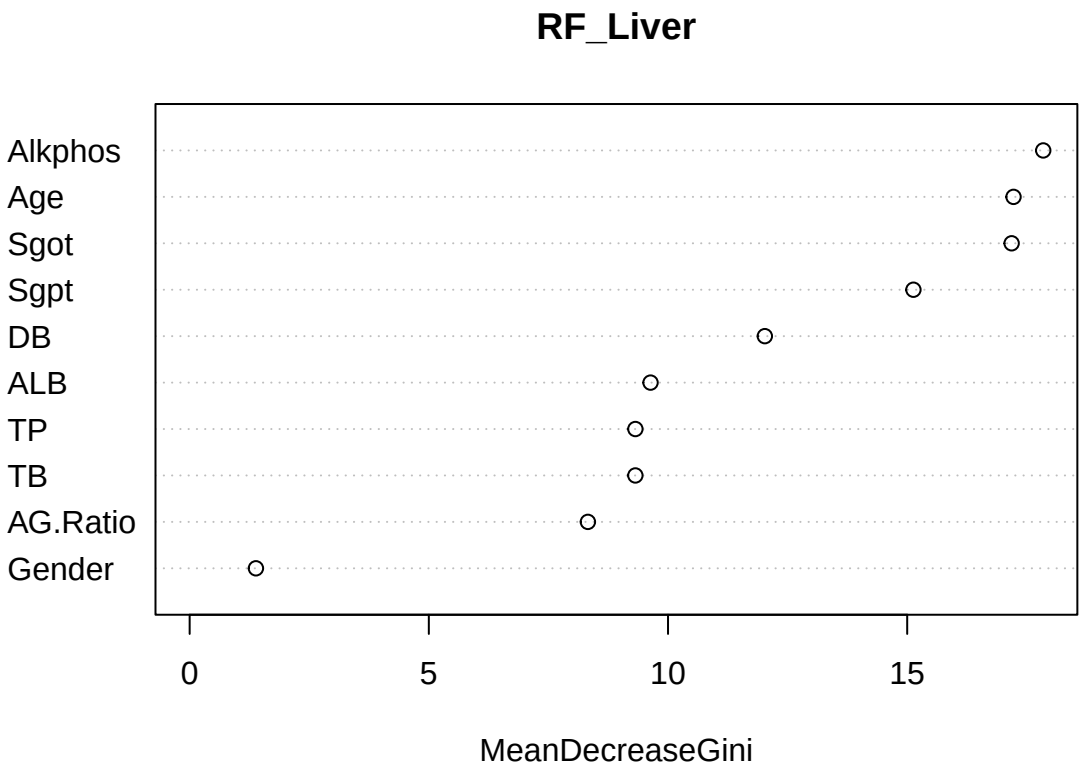
Interesting that sampling 1 variable at each split is the optimal number. We'll refit the model with the best $m$ and look at the variable importance:

```r
RF_Liver_final = randomForest(
  Selector ~ .,
  data = ILPD_train,
  ntree = 2000,
  mtry = 1,
  importance = TRUE
)

(imp <- importance(RF_Liver))
```

| | MeanDecreaseGini |
|---|---|
| Age | 17.222401 |
| Gender | 1.385186 |
| TB | 9.317460 |
| DB | 12.023891 |
| Alkphos | 17.843266 |
| Sgpt | 15.129751 |
| Sgot | 17.182995 |
| TP | 9.317718 |
| ALB | 9.634584 |

| | MeanDecreaseGini |
|---|---|
| AG.Ratio | 8.325047 |

```
varImpPlot(RF_Liver)
```




Based on the Mean Decrease in Gini, the three variables seem to matter the most are: Alkphos, Sgpt, and Age.

4. Now let's fit using GBM. To do this, we need to change the outcome to a numeric variable. As a result, i'm going to make new datasets just for gbm.

```
ILPD_train_gbm <- ILPD_train %>%
  mutate(
    Selector = as.numeric(Selector) - 1,
    Gender = as.numeric(Gender) - 1
  )
ILPD_vald_gbm <- ILPD_vald %>%
  mutate(
    Selector = as.numeric(Selector) - 1,
    Gender = as.numeric(Gender) - 1
  )
ILPD_test_gbm <- ILPD_test %>%
  mutate(
    Selector = as.numeric(Selector) - 1,
```

```
    Gender = as.numeric(Gender) - 1
  )


shrink_vec <- c(0.001, 0.01, 0.05, 0.1, 0.2, 0.3)
interaction.depth.vec <- 1:6
num_trees <- floor(seq(100,3000,length.out = 50))
cv.loss <- NULL
for(i in interaction.depth.vec){
  for(s in shrink_vec){
    set.seed(2394879)
    boost.liver = gbm(as.numeric(Selector) ~. ,data=ILPD_train_gbm,
                      distribution = "bernoulli", n.trees = 3000,
                      interaction.depth = i, shrinkage = s, cv.folds = 10)

    t_cv.loss <- boost.liver$cv.error[num_trees]
    cv.loss <- rbind(cv.loss,
                     cbind(s, i, num_trees, t_cv.loss) )
  }
}

cv.loss <- data.frame(cv.loss)
names(cv.loss) <- c("s", "Int_depth", "num_trees","cv_loss")
cv.loss[which.min(cv.loss$cv_loss),]
```

| | s | Int_depth | num_trees | cv_loss |
|---|---|---|---|---|
| 53 | 0.01 | 1 | 218 | 1.134959 |

Now, let's refit for the best model:

```
(M <- cv.loss$num_trees[which.min(cv.loss$cv_loss)] )
```

```
## [1] 218
```

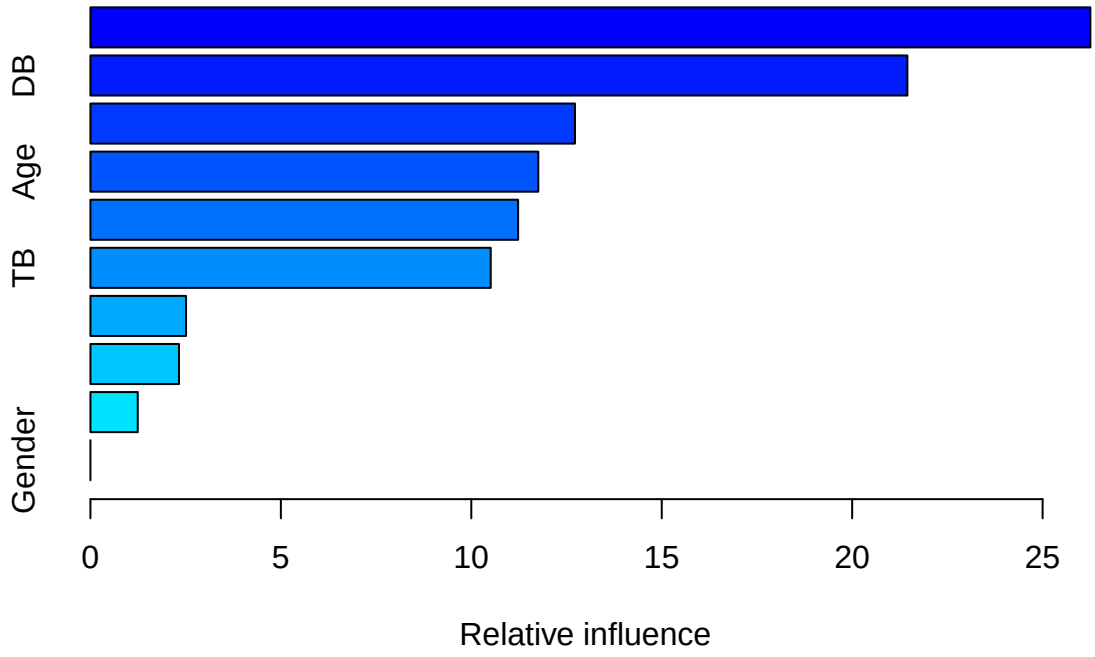
```
(s <- cv.loss$s[which.min(cv.loss$cv_loss)])
```

```
## [1] 0.01
```

```
(i <- cv.loss$Int_depth[which.min(cv.loss$cv_loss)])
```

```
## [1] 1
```

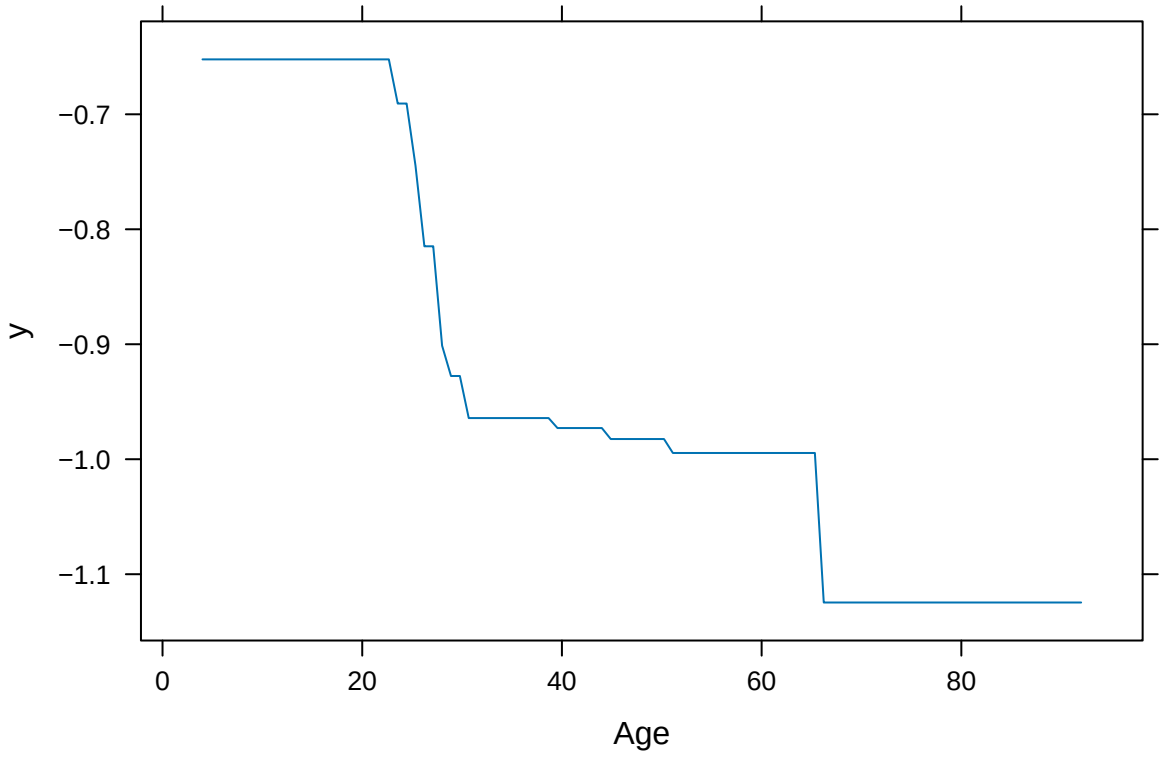
```
boost.liver=gbm(Selector ~. ,data = ILPD_train_gbm,
                distribution = "bernoulli",
                n.trees=M, shrinkage = s, interaction.depth=i)

summary(boost.liver)
```



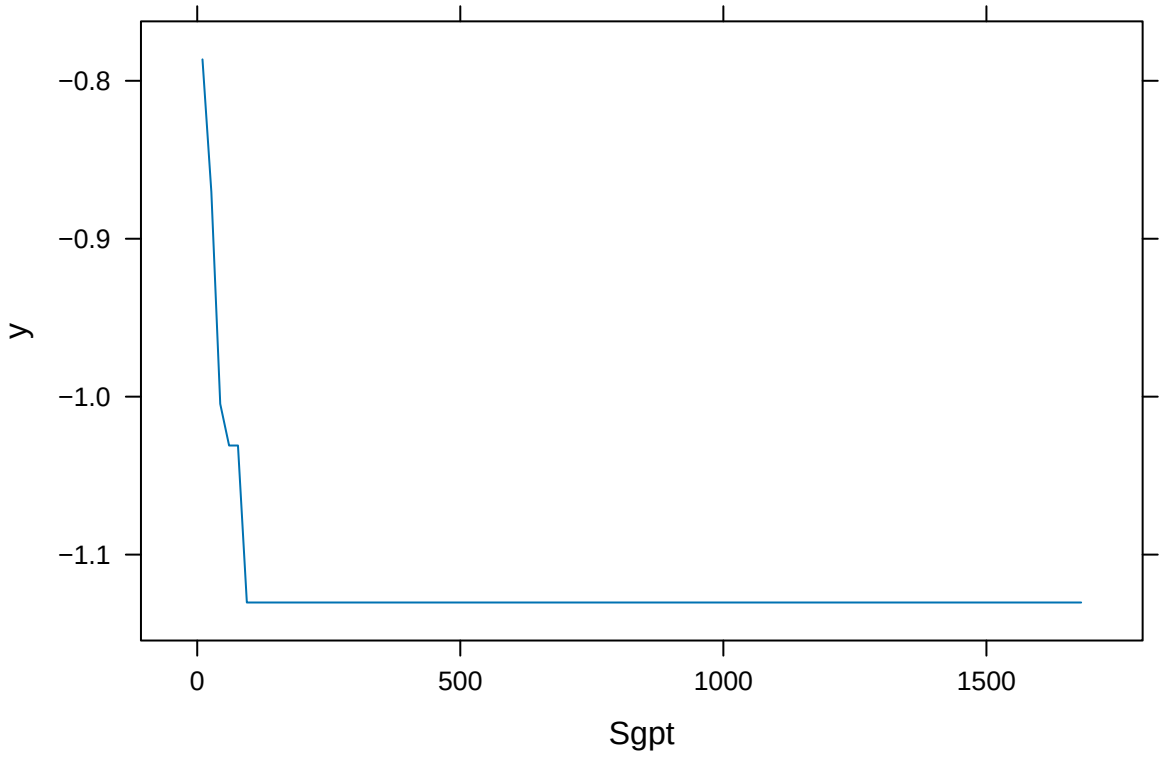

| | var | rel.inf |
|---|---|---|
| Sgot | Sgot | 26.254014 |
| DB | DB | 21.450141 |
| Alkphos | Alkphos | 12.721659 |
| Age | Age | 11.759037 |
| Sgpt | Sgpt | 11.229044 |
| TB | TB | 10.509943 |
| AG.Ratio | AG.Ratio | 2.510003 |
| ALB | ALB | 2.324413 |
| TP | TP | 1.241744 |
| Gender | Gender | 0.000000 |

Let's look at the effect of the variables I found to be most important in the previous question:

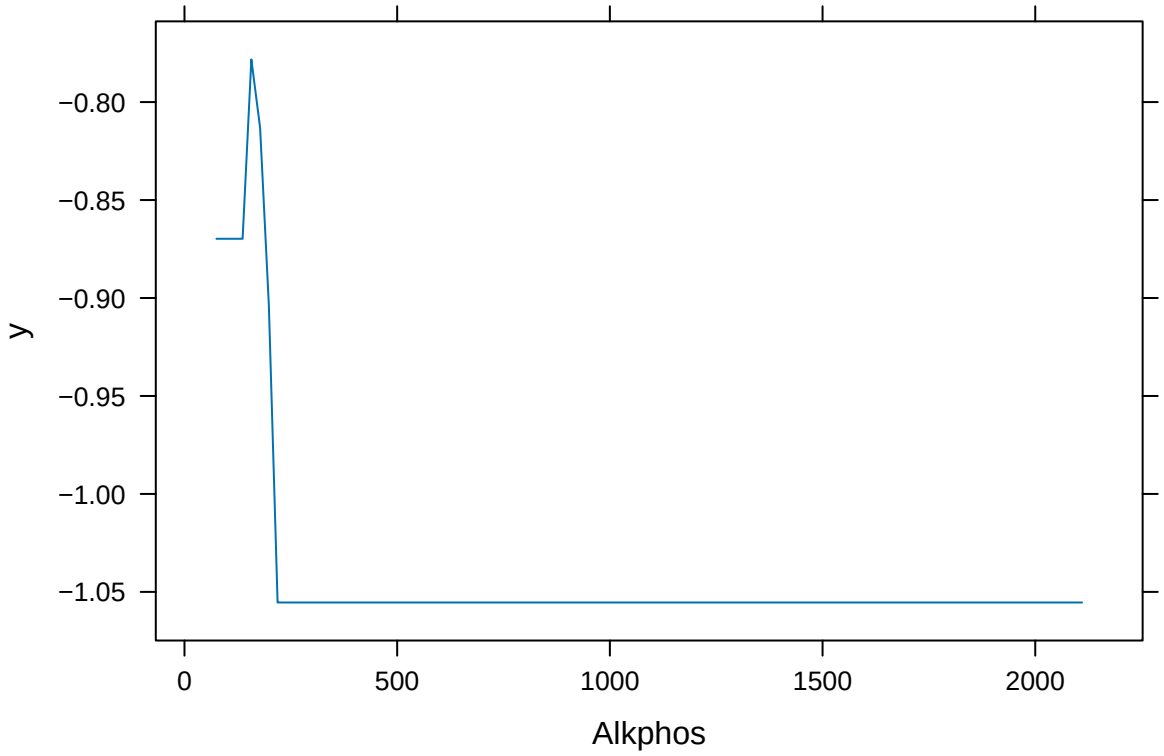
```
plot(boost.liver, i= "Age")
```




```
plot(boost.liver, i= "Sgpt")
```




```
plot(boost.liver, i= "Alkphos")
```




5. Now, let's compare the MSE of all of the models.

First, I'm going to use the validation data (which hasn't been used yet) to select the best model. Recall that I'm going to skip CART since it didn't have a useful model.

```
### svm
svm_vald_yhat <- predict(M.CV, newdata = ILPD_vald)
table(svm_vald_yhat, ILPD_vald$Selector)
```

| svm_vald_yhat/ | 1 | 2 |
|---|---|---|
| 1 | 64 | 23 |
| 2 | 0 | 0 |

```
mse_svm <- mean((as.numeric(svm_vald_yhat) - as.numeric(ILPD_vald$Selector))^2)

#### RF
RF_vald_yhat = predict(RF_Liver, newdata = ILPD_vald)
table(RF_vald_yhat, ILPD_vald$Selector)
```

| RF_vald_yhat/ | 1 | 2 |
|---|---|---|
| 1 | 57 | 16 |

| RF_vald_yhat/ | 1 | 2 |
|---|---|---|
| 2 | 7 | 7 |

```
mse_RF <- mean((as.numeric(RF_vald_yhat) - as.numeric(ILPD_vald$Selector))^2)


### GBM
GBM_vald_yhat = predict(boost.liver, newdata=ILPD_vald_gbm, n.trees = M, type = "response")

mse_gbm_numer <- mean((as.numeric(GBM_vald_yhat) - as.numeric(ILPD_vald_gbm$Selector))^2)

GBM_vald_yhat[GBM_vald_yhat >= 0.5] <- 1
GBM_vald_yhat[GBM_vald_yhat < 0.5] <- 0

mse_gbm_class <- mean((as.numeric(GBM_vald_yhat) - as.numeric(ILPD_vald_gbm$Selector))^2)

table(GBM_vald_yhat, ILPD_vald$Selector)
```

| GBM_vald_yhat/ | 1 | 2 |
|---|---|---|
| 0 | 64 | 18 |
| 1 | 0 | 5 |

```
c(mse_svm, mse_RF, mse_gbm_numer, mse_gbm_class)
```

```
## [1] 0.2643678 0.2643678 0.1498010 0.2068966
```

From this we can see that the $GBM$ model works the best (defining the predictions as hard or soft). The only reason to show both is to demonstrate the the finding is consistent with either type of prediction, however, it does show that we can expect the soft prediction to be closer to the truth (on average). Going forward, i'm going to use the numeric (soft) version of the predictions

Now, I'm going to estimate the error of the GBM model we'll use the test dataset (which hasn't been used yet). Further, I'm going to use the test dataset to estimate the standard deviation of our MSE estimate using CV.

To do this I will:

- split the test data into $K$ folds,
- estimate the MSE for each of the $K$ folds, and
- estimate the standard deviation of the $K$ MSE values.

My final MSE estimate will be the mean of the $K$ MSE values, and the standard error with be the standard deviation of the $K$ MSE values divided by $\sqrt{(K)}$.

```
# split the test data into $K$ folds
set.seed(29387)
K <- 20
n_test <- nrow(ILPD_test_gbm)
test_samp <- sample(
  rep(1:K,ceiling(n_test/K))[1:n_test],
```

```
  n_test,
  replace = FALSE)
mse_K <- NULL
#Get the predicted values
GBM_test_yhat = predict(boost.liver, newdata=ILPD_test_gbm, n.trees = M, type = "response")
# estimate the MSE for each of the K folds
for(k in 1:K){
  mse_K <- c(mse_K,
             mean((GBM_test_yhat[test_samp == k] - ILPD_test_gbm$Selector[test_samp == k])^2)
  )
}

# Below is the mean, standard deviation, standard error, and 95% CI
(mse_est <- mean(mse_K))
```

```
## [1] 0.1865418
```

```
(sd_mse <- sd(mse_K))
```

```
## [1] 0.06710891
```

```
(se_mse <- sd_mse/sqrt(K))
```

```
## [1] 0.01500601
```

```
(CI_95 <- c(mse_est - 1.96*se_mse, mse_est + 1.96*se_mse))
```

```
## [1] 0.1571301 0.2159536
```

This interval is designed to capture the **true** average MSE 95% of the time. However, below what we'll estimate is the average from a sample size of 125 observations. To get a PI for this we would (optimally) want to have our $K$-folds above have 125 observations each. Then, we could use $mse_{est} \pm 1.95 sd_{mse}$ (sd not se). For this situation we'll use the interval for the **true** average MSE.

6. Below is some code to to do the final prediction with my best model:

I'm going to refit with the full `ILPD` dataset then predict for the new data.

```
library(tidyverse)
library(gbm)
# import data ------------------------------------------------------------------
ILPD <- read.csv("ILPD.csv")
# Recode ages greater than 89 to 92
ILPD <- ILPD %>%
  mutate(
    Age = ifelse(Age == 90, 92, Age),
    Selector = as.factor(Selector),
    Gender = as.factor(Gender)
  ) %>%
  na.omit() %>%
```

```
  mutate(
    Selector = as.numeric(Selector) - 1,
    Gender = as.numeric(Gender) - 1
  )

# build GBM model ----------------------------------------------------

M <- 218
s <- 0.01
i <- 1
set.seed(2394879)
gbm.liver=gbm(Selector ~. ,data=ILPD,
              distribution = "bernoulli",
              n.trees=M, shrinkage = s, interaction.depth=i)

# read in test dataset -----------------------------------------------
ILPD_test <- read.csv("ILPD_test.csv")
ILPD_test <- ILPD_test %>%
  mutate(
    Age = ifelse(Age == 90, 92, Age),
    Selector = as.factor(Selector),
    Gender = as.factor(Gender)
  ) %>%
  na.omit() %>%
  mutate(
    Selector = as.numeric(Selector) - 1,
    Gender = as.numeric(Gender) - 1
  )


# predict
GBM_test_yhat = predict(gbm.liver, newdata=ILPD_test, n.trees = M, type = "response")

# Make the final dataset:
mse_gbm <- mean((GBM_test_yhat - ILPD_test$Selector)^2)
mclain.alex_error <- data.frame("mclain.alex_error", "gbm", round(mse_gbm, 3) )
names(mclain.alex_error) <- c("name", "best.model", " est.error")
mclain.alex_error
```

| name | best.model | est.error |
|---|---|---|
| mclain.alex_error | gbm | 0.161 |

We can see that the value 0.161 is within the interval created above $(0.157, 0.216)$.

Just for fun, let's see what the misclassification rate and confusion matrix for the hard classification prediction would look like:

```
GBM_test_yhat[GBM_test_yhat >= 0.5] <- 1
GBM_test_yhat[GBM_test_yhat < 0.5] <- 0

(mse_gbm_class <- mean((GBM_test_yhat - ILPD_test$Selector)^2))
```

```
## [1] 0.2520325
```

```
table(GBM_test_yhat, ILPD_test$Selector)
```

| GBM_test_yhat/ | 0 | 1 |
|---|---|---|
| 0 | 90 | 31 |
| 1 | 0 | 2 |

So, $92/125 = 0.736$ is the accuracy.