BAS 474: Final Coding Exam

Isaac Sheets

Due: April 26, 2020

- Do not discuss this work with anyone. If you have any questions, you should consult your instructor.
- Your instructors will not answer questions on Sunday.
- For questions without tuning parameters given, you don't have to include tuneGrid in the train function. In this way some default parameters will be used. You can also set the parameters up by following examples from slides or assignments.

Problem 1. Predictive Model For Tip Percentage

The `TIPS` dataset in the `regclass` package contains information about tips a waiter received over a period of a few months working in one restaurant. The waiter collected information in 8 variables on 244 waiting services (observations). Your task is to build a predictive model for `TipPercentage`. The variables available in the data are:

- `TipPercentage`: a numeric vector, the tip written as a percentage (0-100) of the total bill
- `Bill`: a numeric vector, the bill amount (dollars)
- `Tip`: a numeric vector, the tip amount (dollars)
- `Gender`: a factor with levels Female Male, gender of the payer of the bill
- `Smoker`: a factor with levels No Yes, whether the party included smokers
- `Weekday`: a factor with levels Friday Saturday Sunday Thursday, day of the week
- `Time`: a factor with levels Day Night, rough time of day
- `PartySize`: a numeric vector, number of people in party

Do not use the `Tip` variable as predictor (you can remove this variable by nulling it out).

Split the data into 80% training rows with the remaining rows being the holdout (use `set.seed(474)` on the same line as the required `sample` command).

Use `set.seed(474)` everywhere randomness is infused in the training process.

Use 5-fold cross-validation to estimate the generalization error.

```
data(TIPS)
TIPS$Tip <- NULL
set.seed(474); train.rows <- sample(nrow(TIPS), 0.8*nrow(TIPS))
TRAIN <- TIPS[train.rows,];
```

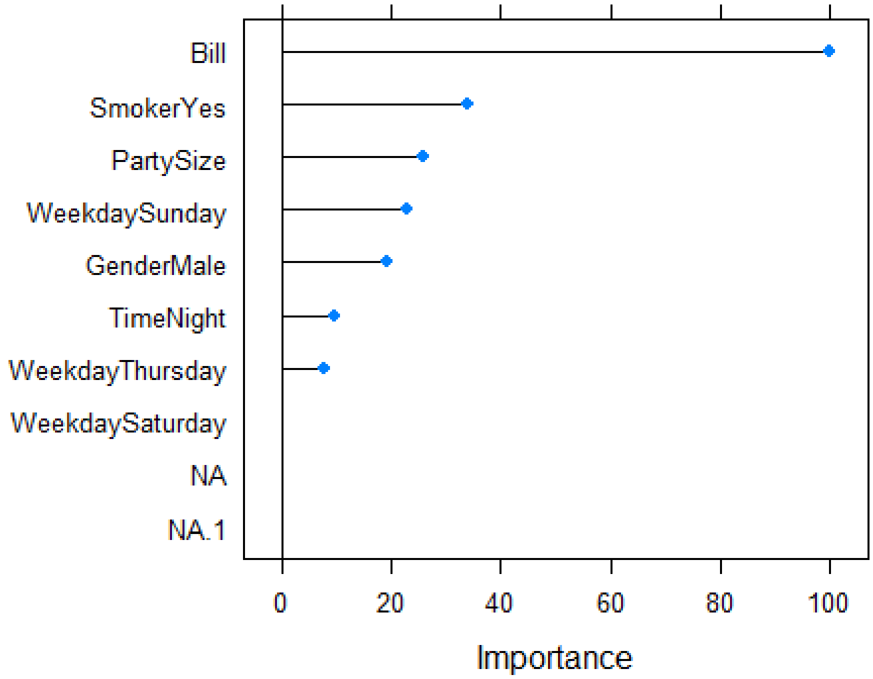
```
HOLDOUT <- TIPS[-train.rows,]
fitControl <- trainControl(method="cv", number=5)
```

(a) Build a vanilla linear regression model using the training data to predict `TipPercentage`. Report the estimated generalization RMSE and the RMSE on the holdout sample. Report the variable importance plot, and comment on which predictors appear most important for predicting `TipPercentage`.

```
set.seed(474); GLM <- train(TipPercentage~., data=TRAIN, method="glm",
                             trControl=fitControl, preProc=c("center", "scale"
))
GLM$results
##   parameter     RMSE  Rsquared      MAE   RMSESD RsquaredSD     MAESD
## 1      none 5.904779 0.1390757 4.134148 2.116873 0.07076399 0.6548951

set.seed(474); classification.glm <- predict(GLM,newdata=HOLDOUT)
RMSE(HOLDOUT$TipPercentage, classification.glm)
## [1] 4.620693

IMP <-varImp(GLM)$importance
IMP$Variable <-rownames(IMP)
IMP <- IMP[order(IMP$Overall,decreasing=TRUE),]
head(IMP)
##                    Overall      Variable
## Bill            100.000000          Bill
## SmokerYes        34.081084     SmokerYes
## PartySize        25.785500     PartySize
## WeekdaySunday    23.071185 WeekdaySunday
## GenderMale       19.193405    GenderMale
## TimeNight         9.664646     TimeNight
plot(varImp(GLM), top=10)
```



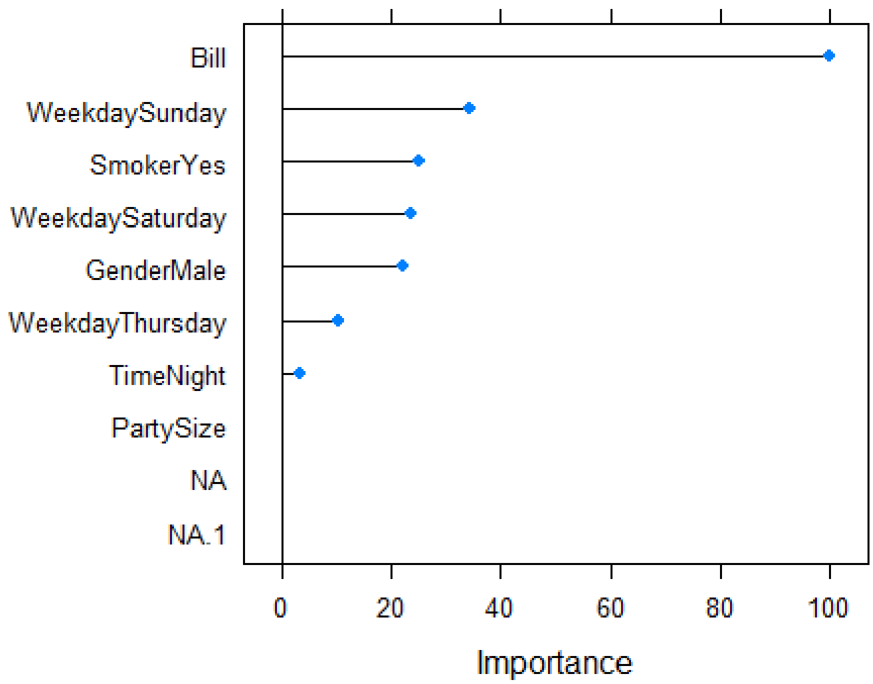
RESPONSE: The estimated generalization RMSE is 5.527699. The RMSE of the holdout sample is 5.63277. The predictors that appear to be most important for predicting TipPercentage are Bill, PartySize, and SmokersYes.

(b) Build a regularized multiple linear regression model using the training data to predict `TipPercentage`. Audition `alpha` values along the sequence 0, 0.1, 0.2, ..., 0.9, 1 and `lambda` of 10 raised to the -4, -3.5, ..., 1.5, 2 powers. Report the estimated generalization RMSE and the RMSE on the holdout sample. Report the variable importance plot, and comment on which predictors appear most important for predicting `TipPercentage`.

```
glmnetGrid <- expand.grid(alpha = seq(0,1, by=0.1), lambda = 10^seq(-4,2, by=
0.5))
set.seed(474); GLMnet <- train(TipPercentage~., data=TRAIN, method="glmnet",
tuneGrid=glmnetGrid,
                              trControl=fitControl, preProc=c("center", "scale
"))
## Warning in nominalTrainWorkflow(x = x, y = y, wts = weights, info = trainI
nfo, :
## There were missing values in resampled performance measures.
GLMnet$results[rownames(GLMnet$bestTune),]
##    alpha   lambda     RMSE  Rsquared     MAE   RMSESD RsquaredSD     MAESD
## 10     0 3.162278 5.813812 0.1323445 3.98199 2.257199 0.06256246 0.6354841

set.seed(474); classification.glmnet <- predict(GLMnet,newdata=HOLDOUT)
RMSE(HOLDOUT$TipPercentage, classification.glmnet)
## [1] 4.579815
```

```
IMP <-varImp(GLMnet)$importance
IMP$Variable <-rownames(IMP)
IMP <- IMP[order(IMP$Overall,decreasing=TRUE),]
head(IMP)
##                    Overall        Variable
## Bill             100.00000            Bill
## WeekdaySunday     34.20049   WeekdaySunday
## SmokerYes         25.07473       SmokerYes
## WeekdaySaturday   23.80801 WeekdaySaturday
## GenderMale        22.26291      GenderMale
## WeekdayThursday   10.49329 WeekdayThursday
plot(varImp(GLMnet), top=10)
```



Response: The estimated generalization RMSE is 5.46346. The RMSE of the holdout sample is 5.607496. The predictors that appear to be most important for predicting TipPercentage are Bill, PartySize, SmokersYes, WeekdaySunday, and WeekdaySaturday.

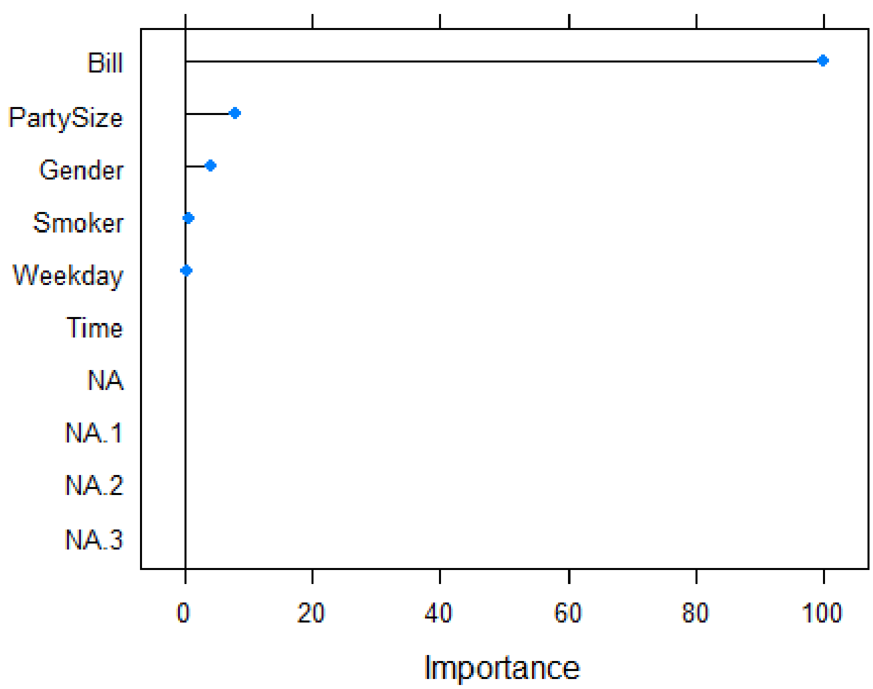
(c) Build a KNN model using your training data to predict `TipPercentage`. Audition k values of 1-30. Report the estimated generalization RMSE and the RMSE on the holdout sample. Report the variable importance plot, and comment on which predictors appear most important for predicting `TipPercentage`.

```
knnGrid <- expand.grid(k=1:30)
set.seed(474); KNN <- train(TipPercentage~.,data=TRAIN, method='knn', preProc
= c("center", "scale"),
```

```
                          trControl=fitControl, tuneGrid=knnGrid)
KNN$results[which(KNN$results$k == as.numeric(KNN$bestTune) ),]
##     k     RMSE  Rsquared     MAE   RMSESD RsquaredSD     MAESD
## 14 14 5.742774 0.1371878 4.02328 2.165275 0.07539694 0.6360195

set.seed(474); classification.knn <- predict(KNN,newdata=HOLDOUT)
RMSE(HOLDOUT$TipPercentage, classification.knn)
## [1] 4.489457

IMP <-varImp(KNN)$importance
IMP$Variable <-rownames(IMP)
IMP <- IMP[order(IMP$Overall,decreasing=TRUE),]
head(IMP)
##               Overall  Variable
## Bill      100.0000000      Bill
## PartySize   7.9935800 PartySize
## Gender      4.1750099    Gender
## Smoker      0.5760554    Smoker
## Weekday     0.4241152   Weekday
## Time        0.0000000      Time
plot(varImp(KNN), top=10)
```



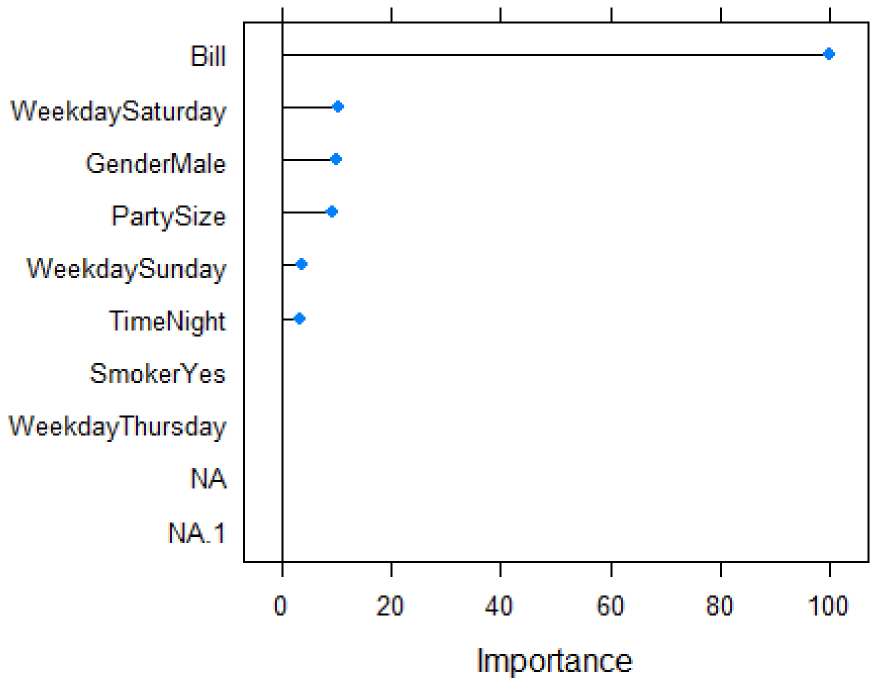
RESPONSE: The estimated generalization RMSE is 5.566491. The RMSE on the holdout sample is 5.513077. It appears that the only important predictor for predicting TipPercentage is Bill.

(d) Build a regression tree model using your training data to predict `TipPercentage`. Report the estimated generalization RMSE and the RMSE on the holdout sample. Report the variable importance plot, and comment on which predictors appear most important for predicting `TipPercentage`.

```
set.seed(474); TREE <- train(TipPercentage~.,data=TRAIN,method="rpart",trCont
rol=fitControl,
                             preProc=c("center", "scale"))
## Warning in nominalTrainWorkflow(x = x, y = y, wts = weights, info = trainI
nfo, :
## There were missing values in resampled performance measures.
TREE$results[rownames(TREE$bestTune),]
##           cp     RMSE  Rsquared      MAE   RMSESD RsquaredSD     MAESD
## 1 0.03589843 6.132361 0.1241258 4.132534 1.508779  0.1275679 0.3703102

set.seed(474); classification.tree <- predict(TREE,newdata=HOLDOUT)
RMSE(HOLDOUT$TipPercentage, classification.tree)
## [1] 4.935643

IMP <-varImp(TREE)$importance
IMP$Variable <-rownames(IMP)
IMP <- IMP[order(IMP$Overall,decreasing=TRUE),]
head(IMP)
##                    Overall        Variable
## Bill            100.000000            Bill
## WeekdaySaturday  10.529947 WeekdaySaturday
## GenderMale        9.948801      GenderMale
## PartySize         9.429961       PartySize
## WeekdaySunday     3.977997   WeekdaySunday
## TimeNight         3.441621       TimeNight
plot(varImp(TREE), top=10)
```



RESPONSE: The estimated generalization RMSE is 5.76658. The RMSE on the holdout sample is 5.75445. It appears that the only important predictor for predicting TipPercentage is Bill.

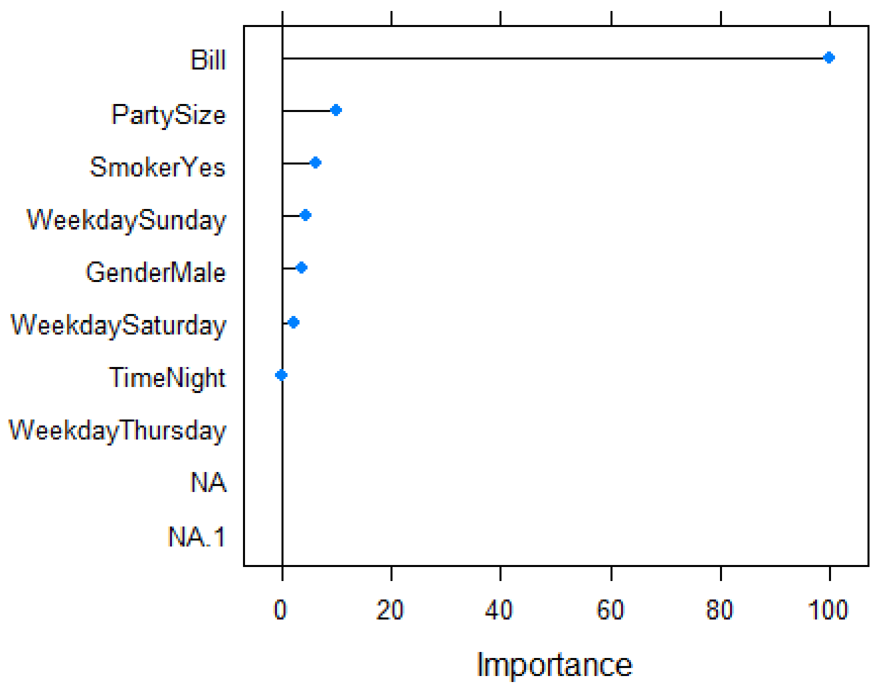
(e) Build a random forest model using your training data to predict `TipPercentage`. Audition values of `mtry` of 1-5. Report the estimated generalization RMSE and the RMSE on the holdout sample. Report the variable importance plot, and comment on which predictors appear most important for predicting `TipPercentage`.

```
forestGrid <- expand.grid(mtry=seq(5))
set.seed(474); FOREST <- train(TipPercentage~.,data=TRAIN,method="rf",tuneGri
d=forestGrid,
                               trControl=fitControl,preProc=c("center","scale
"))
FOREST$results[rownames(FOREST$bestTune),]
##   mtry     RMSE  Rsquared      MAE   RMSESD RsquaredSD     MAESD
## 2    2 5.832596 0.1560506 3.979021 1.918469  0.1393561 0.5258356

set.seed(474); classification.forest <- predict(FOREST,newdata=HOLDOUT)
RMSE(HOLDOUT$TipPercentage, classification.forest)
## [1] 4.881135

IMP <-varImp(FOREST)$importance
IMP$Variable <-rownames(IMP)
IMP <- IMP[order(IMP$Overall,decreasing=TRUE),]
head(IMP)
##                      Overall        Variable
```

```
## Bill               100.000000           Bill
## PartySize           10.003010      PartySize
## SmokerYes            6.401454      SmokerYes
## WeekdaySunday        4.477199  WeekdaySunday
## GenderMale           3.757691     GenderMale
## WeekdaySaturday      2.200660 WeekdaySaturday
plot(varImp(FOREST), top=10)
```



RESPONSE: The estimated generalization RMSE is 5.678023. The RMSE on the holdout sample is 5.894915. It appears that the only important predictor for predicting TipPercentage is Bill.

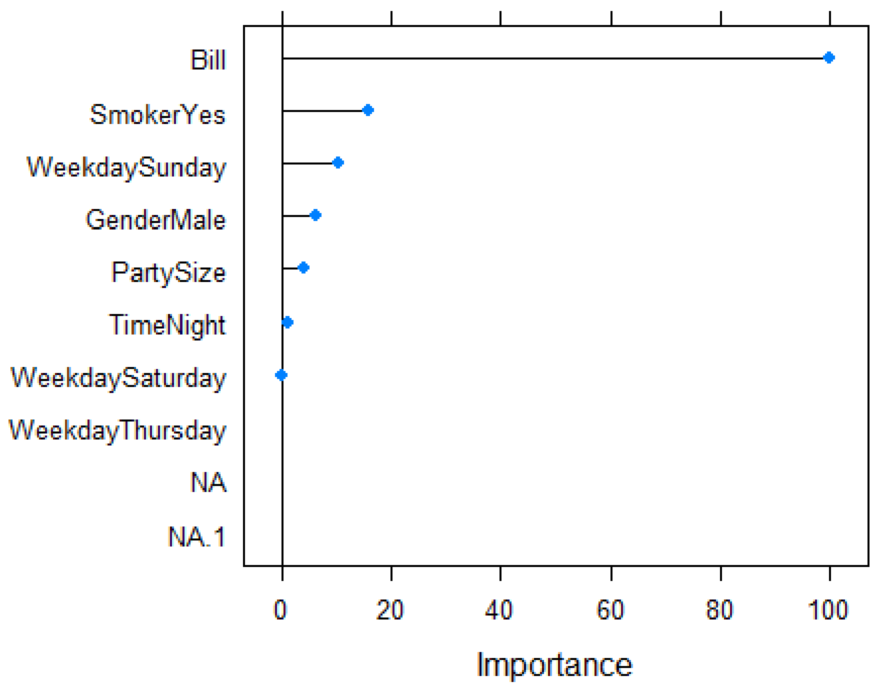
(f) Build a gradient boosted tree model using the training data to predict `TipPercentage`. Report the estimated generalization RMSE and the RMSE on the holdout sample. Report the variable importance plot, and comment on which predictors appear most important for predicting `TipPercentage`.

```
set.seed(474); GBM <- train(TipPercentage~.,data=TRAIN,method="gbm",
                            trControl=fitControl,preProc=c("center","scale"),
verbose=FALSE)
GBM$results[rownames(GBM$bestTune),]
##   shrinkage interaction.depth n.minobsinnode n.trees     RMSE  Rsquared
## 7       0.1                 3             10      50 5.709391 0.1757444
##         MAE  RMSESD RsquaredSD     MAESD
## 7 4.050114 1.99549  0.1020458 0.5408795

set.seed(474); classification.gbm <- predict(GBM,newdata=HOLDOUT)
```

```
RMSE(HOLDOUT$TipPercentage, classification.gbm)
## [1] 4.874882

IMP <-varImp(GBM)$importance
IMP$Variable <-rownames(IMP)
IMP <- IMP[order(IMP$Overall,decreasing=TRUE),]
head(IMP)
##                    Overall      Variable
## Bill            100.000000          Bill
## SmokerYes        15.790421     SmokerYes
## WeekdaySunday    10.570905 WeekdaySunday
## GenderMale        6.291911    GenderMale
## PartySize         4.035395     PartySize
## TimeNight         1.093058     TimeNight
plot(varImp(GBM), top=10)
```



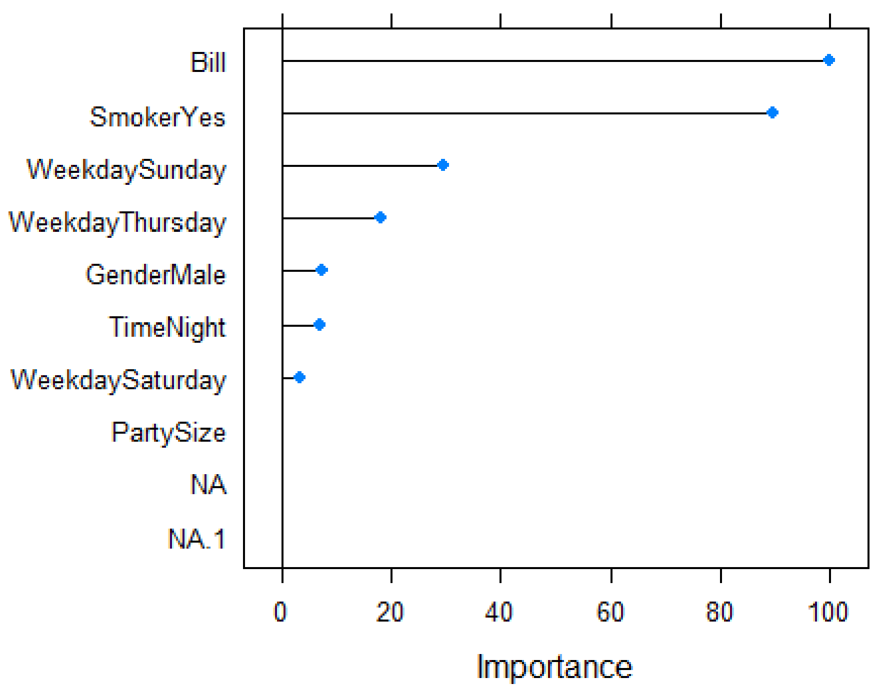
RESPONSE: The estimated generalization RMSE is 5.48649. The RMSE on the holdout sample is 5.52449. It appears that the only important predictor for predicting TipPercentage is Bill.

(g) Build a neural network model with one hidden layer using the training data to predict `TipPercentage`. Audition number of nodes (neurons) in 1-6 and decay of 10 raised to the -2, -1.5, ..., 0.5, 1 powers. Report the estimated generalization RMSE and the RMSE on the holdout sample. Report the variable importance plot, and comment on which predictors appear most important for predicting `TipPercentage`.

```
nnetGrid <- expand.grid(size=1:6,decay=c(10^seq(-2,1,by=0.5) ) )
set.seed(474); NNET <- train(TipPercentage~., data=TRAIN, hidden=1, method='n
net', trControl=fitControl,
                             tuneGrid=nnetGrid, preProc = c("center", "scale"
),
                             verbose=FALSE, trace=FALSE,linout=TRUE)
NNET$results[rownames(NNET$bestTune),]
##    size decay     RMSE   Rsquared      MAE   RMSESD RsquaredSD     MAESD
## 35    5    10 5.835846 0.1335544 4.041775 2.142476 0.06495943 0.6147713

set.seed(474); classification.nnet <- predict(NNET,newdata=HOLDOUT)
RMSE(HOLDOUT$TipPercentage, classification.nnet)
## [1] 4.602111

IMP <-varImp(NNET)$importance
IMP$Variable <-rownames(IMP)
IMP <- IMP[order(IMP$Overall,decreasing=TRUE),]
head(IMP)
##                     Overall        Variable
## Bill             100.000000            Bill
## SmokerYes         89.388789       SmokerYes
## WeekdaySunday     29.503769   WeekdaySunday
## WeekdayThursday   18.345051 WeekdayThursday
## GenderMale         7.371916      GenderMale
## TimeNight          6.960453       TimeNight
plot(varImp(NNET), top=10)
```



Response: The estimated generalization RMSE is 5.487836. The RMSE of the holdout sample is 6.078132. The predictors that appear to be most important for predicting TipPercentage are SmokerYes, Bill, and GenderMale.

(h) Is one model a compelling choice for predicting `TipPercentage` over the others? Why or why not? Show evidence to support your claim.

```
GLM$results
##   parameter     RMSE  Rsquared      MAE   RMSESD RsquaredSD     MAESD
## 1      none 5.904779 0.1390757 4.134148 2.116873 0.07076399 0.6548951
GLMnet$results[rownames(GLMnet$bestTune),]
##    alpha   lambda     RMSE  Rsquared     MAE   RMSESD RsquaredSD     MAESD
## 10     0 3.162278 5.813812 0.1323445 3.98199 2.257199 0.06256246 0.6354841
KNN$results[which(KNN$results$k == as.numeric(KNN$bestTune) ),]
##     k     RMSE  Rsquared     MAE   RMSESD RsquaredSD     MAESD
## 14 14 5.742774 0.1371878 4.02328 2.165275 0.07539694 0.6360195
TREE$results[rownames(TREE$bestTune),]
##           cp     RMSE  Rsquared      MAE   RMSESD RsquaredSD     MAESD
## 1 0.03589843 6.132361 0.1241258 4.132534 1.508779  0.1275679 0.3703102
FOREST$results[rownames(FOREST$bestTune),]
##   mtry     RMSE  Rsquared      MAE   RMSESD RsquaredSD     MAESD
## 2    2 5.832596 0.1560506 3.979021 1.918469  0.1393561 0.5258356
GBM$results[rownames(GBM$bestTune),]
##   shrinkage interaction.depth n.minobsinnode n.trees     RMSE  Rsquared
## 7       0.1                 3             10      50 5.709391 0.1757444
##         MAE  RMSESD RsquaredSD     MAESD
## 7 4.050114 1.99549  0.1020458 0.5408795
NNET$results[rownames(NNET$bestTune),]
##    size decay     RMSE  Rsquared      MAE   RMSESD RsquaredSD     MAESD
## 35    5    10 5.835846 0.1335544 4.041775 2.142476 0.06495943 0.6147713
```

No model is a clear winner here, as all RMSE's are well within 1 standard deviation of each other.

Problem 2. Predictive Model for Making a Purchase

The `PURCHASE.csv` dataset contains a small part of a customer database from a bank. Of interest is the variable `Purchase`, which tells us if a customer did or did not make a purchase at a major chain retailer in the following 30 days. Predictor variables include `Visits` (number of visits to the store in the last 90 days), `Spent` (how much the customer has spent in the last 90 days, `PercentClose` (the percentage of purchases this customer makes in general that are within 5 miles of their home address, `Closeset` and `CloseStores` which details how closest the nearest store in the chain is to the customer and how many stores of that chain are within 5 miles of home).

Your task is to build a predictive model to predict `Purchase` (Buy/No).

Split the data into 1000 training rows with the remainder being the holdout (use `set.seed(474)` on the same line as the required `sample` command).

Use `set.seed(474)` everywhere randomness is infused in the training process.

Use 5-fold cross-validation to estimate the generalization AUC (i.e. .

```
PURCHASE <- read.csv("PURCHASE.csv")
set.seed(474); train.rows <- sample(nrow(PURCHASE),1000)
PURCHASE_TRAIN <- PURCHASE[train.rows,];
PURCHASE_HOLDOUT <- PURCHASE[-train.rows,]
fitControl <- trainControl(method="cv",number=5, classProbs=TRUE, summaryFunc
tion = twoClassSummary)
# set fitControl for estimating the generalization error. Here use AUC to tra
in and compare models.
```

(a) Which class will the naive model classify everyone in data to? What is the estimated generalization accuracy of the this model? What is its accuracy in the holdout sample?

```
summary(PURCHASE_TRAIN$Purchase)
## Buy  No
## 252 748
summary(PURCHASE_HOLDOUT$Purchase)
##   Buy    No
##  6640 20083
782/(218+782)
## [1] 0.782
20049/(6674+20049)
## [1] 0.7502526
```

Response: The naive model would classify everyone in the data to "No". The estimated generalization accuracy woud be 0.782. The model's accuracy on the holdout sample would be 0.7502526.

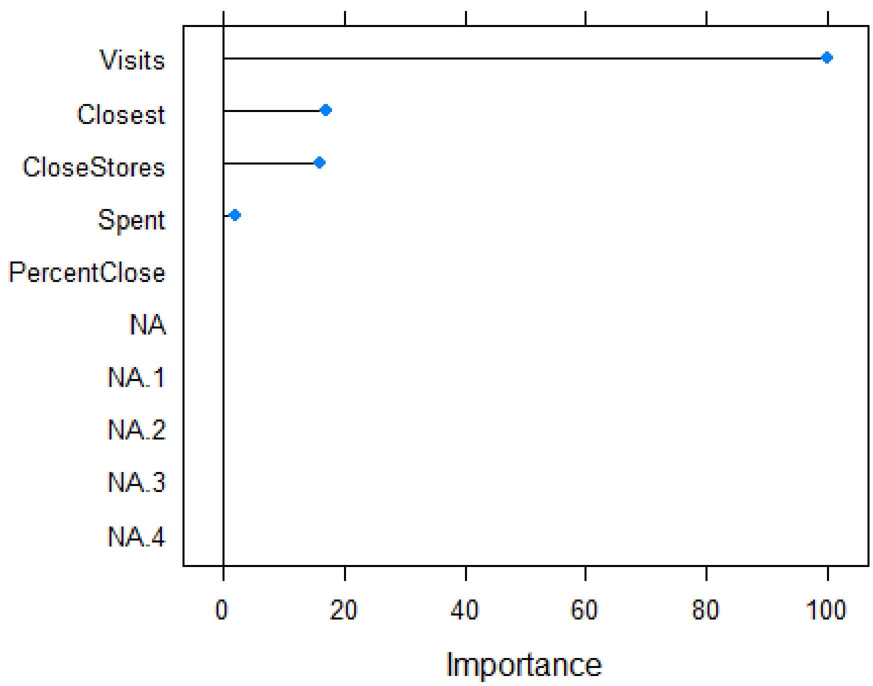
(b) Using the training data, train a logistic regression model to predict `Purchase`. Report the estimated generalization metrics of the best model, as well as its accuracy and auc on the holdout sample. Also, report the variable importance plot, and comment on which predictors appear most important for predicting `Purchase`.

```
set.seed(474); GLM2 <- train(Purchase~., data=PURCHASE_TRAIN, method="glm",
                             trControl=fitControl, preProc=c("center", "scale"
))
GLM2$results
##   parameter       ROC       Sens      Spec      ROCSD     SensSD      Spec
SD
## 1      none 0.5827802 0.01984314 0.9866488 0.08689495 0.01414377 0.0094281
75

set.seed(474); classification.glm2 <- predict(GLM2,newdata=PURCHASE_HOLDOUT)
mean(classification.glm2==PURCHASE_HOLDOUT$Purchase)
## [1] 0.7477454
set.seed(474); roc(PURCHASE_HOLDOUT$Purchase, predict(GLM2,newdata=PURCHASE_H
OLDOUT,type="prob")[,2])
## Setting levels: control = Buy, case = No
## Setting direction: controls < cases
```

```
##
## Call:
## roc.default(response = PURCHASE_HOLDOUT$Purchase, predictor = predict(GLM2
,     newdata = PURCHASE_HOLDOUT, type = "prob")[, 2])
##
## Data: predict(GLM2, newdata = PURCHASE_HOLDOUT, type = "prob")[, 2] in 664
0 controls (PURCHASE_HOLDOUT$Purchase Buy) < 20083 cases (PURCHASE_HOLDOUT$Pu
rchase No).
## Area under the curve: 0.625

IMP <-varImp(GLM2)$importance
IMP$Variable <-rownames(IMP)
IMP <- IMP[order(IMP$Overall,decreasing=TRUE),]
head(IMP)
##                  Overall    Variable
## Visits        100.000000      Visits
## Closest        17.054036     Closest
## CloseStores    16.040762  CloseStores
## Spent           1.952926       Spent
## PercentClose    0.000000 PercentClose
plot(varImp(GLM2), top=10)
```



RESPONSE: The estimated generalization AUC is 0.5917691. The AUC for the holdout sample is 0.6214, and the accuracy of the holdout sample is 0.7457247. The predictors that appear to be most important for predicting Purchase are Visits and Spent.

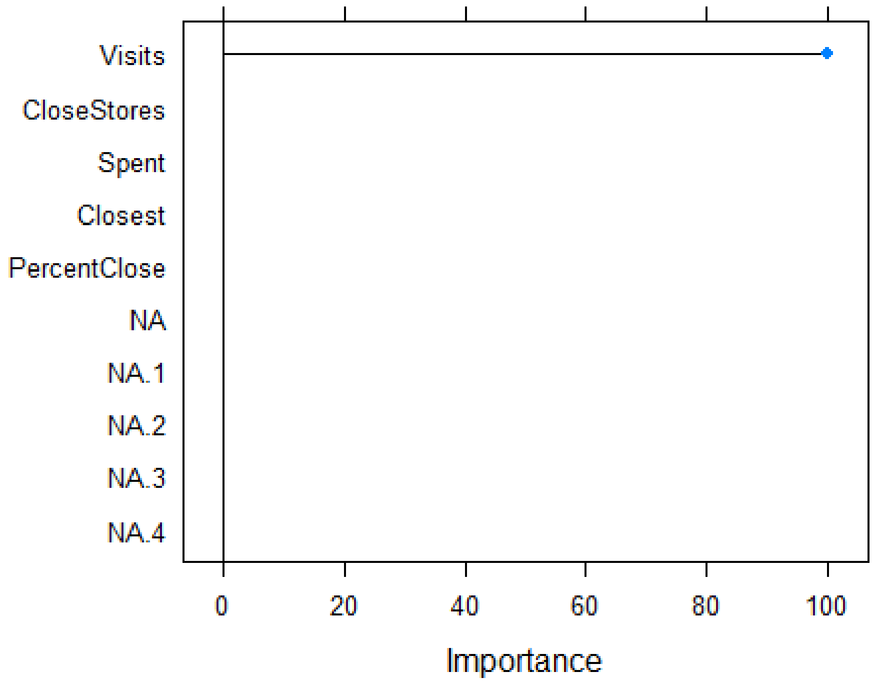
(c) Using the training data, train a regularized logistic regression model to predict `Purchase`. Audition `alpha` values along the sequence 0, 0.1, 0.2, ..., 0.9, 1 and `lambda` of 10 raised to the -3, -2.5, -2, ..., 1.5, 2 powers. Report the estimated generalization metrics for the best model, as well as its accuracy and auc on the holdout samples. Also, report the variable importance plot, and comment on which predictors appear most important for predicting `Purchase`.

```
glmnetGrid <- expand.grid(alpha = seq(0,1, by=0.1), lambda = 10^seq(-3,2, by=
0.5))
set.seed(474); GLMnet2 <- train(Purchase~., data=PURCHASE_TRAIN, method="glmn
et", tuneGrid=glmnetGrid,
                             trControl=fitControl, preProc=c("center", "scale
"))
## Warning in train.default(x, y, weights = w, ...): The metric "Accuracy" wa
s not
## in the result set. ROC will be used instead.
GLMnet2$results[rownames(GLMnet2$bestTune),]
##    alpha     lambda       ROC   Sens  Spec      ROCSD     SensSD      SpecS
D
## 92   0.8 0.03162278 0.6262559 0.008 0.996 0.07445817 0.01095445 0.00894427
2

set.seed(474); classification.glmnet2 <- predict(GLMnet2,newdata=PURCHASE_HOL
DOUT)
mean(classification.glmnet2==PURCHASE_HOLDOUT$Purchase)
## [1] 0.7505894
set.seed(474); roc(PURCHASE_HOLDOUT$Purchase, predict(GLMnet2,newdata=PURCHAS
E_HOLDOUT,type="prob")[,2])
## Setting levels: control = Buy, case = No
## Setting direction: controls < cases
##
## Call:
## roc.default(response = PURCHASE_HOLDOUT$Purchase, predictor = predict(GLMn
et2,     newdata = PURCHASE_HOLDOUT, type = "prob")[, 2])
##
## Data: predict(GLMnet2, newdata = PURCHASE_HOLDOUT, type = "prob")[, 2] in
6640 controls (PURCHASE_HOLDOUT$Purchase Buy) < 20083 cases (PURCHASE_HOLDOUT
$Purchase No).
## Area under the curve: 0.6285

IMP <-varImp(GLMnet2)$importance
IMP$Variable <-rownames(IMP)
IMP <- IMP[order(IMP$Overall,decreasing=TRUE),]
head(IMP)
##              Overall     Variable
## Visits           100       Visits
## Spent              0        Spent
## PercentClose       0 PercentClose
## Closest            0      Closest
```

```
## CloseStores         0  CloseStores
plot(varImp(GLMnet2), top=10)
```



RESPONSE: The estimated generalization AUC is 0.6252065. The AUC for the holdout sample is 0.6287, and the accuracy of the holdout sample is 0.7502526. The only predictor that appears to be important for predicting Purchase is Visits.

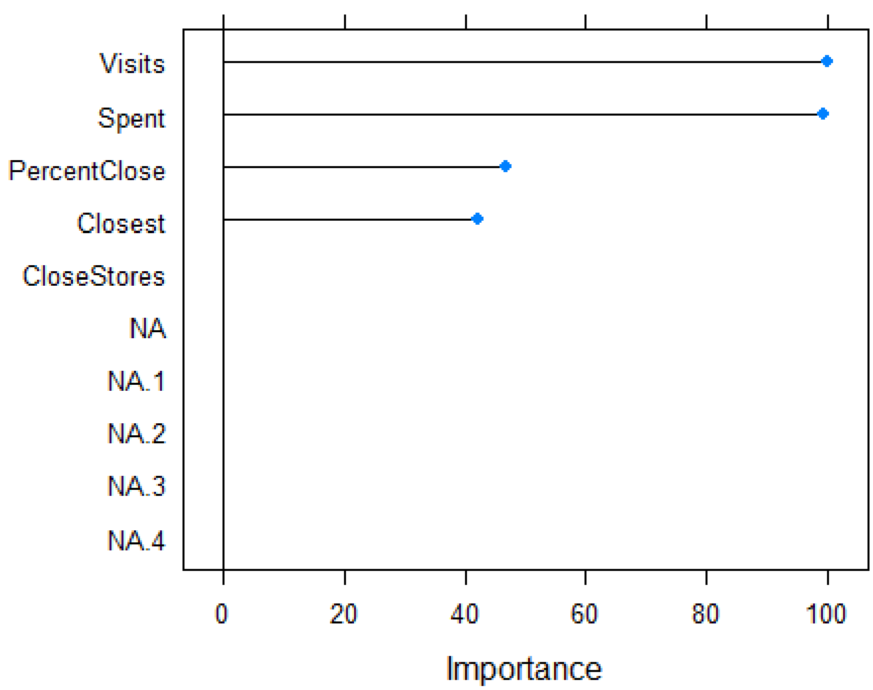
(d) Using the training data, train a classification tree model to predict `Purchase`. Report the estimated generalization metrics of the best model, as well as its accuracy and auc on the holdout sample. Also, report the variable importance plot, and comment on which predictors appear most important for predicting `Purchase`.

```
set.seed(474); TREE2 <- train(Purchase~.,data=PURCHASE_TRAIN,method="rpart",t
rControl=fitControl,
                             preProc=c("center", "scale"))
## Warning in train.default(x, y, weights = w, ...): The metric "Accuracy" wa
s not
## in the result set. ROC will be used instead.
TREE2$results[rownames(TREE2$bestTune),]
##            cp       ROC      Sens      Spec      ROCSD     SensSD     Spec
SD
## 2 0.006613757 0.5642159 0.1705882 0.8704161 0.06042947 0.04577665 0.060303
75

set.seed(474); classification.tree2 <- predict(TREE2,newdata=PURCHASE_HOLDOUT
)
mean(classification.tree2==PURCHASE_HOLDOUT$Purchase)
```

```
## [1] 0.7465105
set.seed(474); roc(PURCHASE_HOLDOUT$Purchase, predict(TREE2,newdata=PURCHASE_
HOLDOUT,type="prob")[,2])
## Setting levels: control = Buy, case = No
## Setting direction: controls < cases
##
## Call:
## roc.default(response = PURCHASE_HOLDOUT$Purchase, predictor = predict(TREE
2,     newdata = PURCHASE_HOLDOUT, type = "prob")[, 2])
##
## Data: predict(TREE2, newdata = PURCHASE_HOLDOUT, type = "prob")[, 2] in 66
40 controls (PURCHASE_HOLDOUT$Purchase Buy) < 20083 cases (PURCHASE_HOLDOUT$P
urchase No).
## Area under the curve: 0.6039

IMP <-varImp(TREE2)$importance
IMP$Variable <-rownames(IMP)
IMP <- IMP[order(IMP$Overall,decreasing=TRUE),]
head(IMP)
##                 Overall    Variable
## Visits        100.00000      Visits
## Spent          99.45258       Spent
## PercentClose   46.73082 PercentClose
## Closest        42.28232     Closest
## CloseStores     0.00000  CloseStores
plot(varImp(TREE2), top=10)
```



RESPONSE: The estimated generalization AUC is 0.6112599. The AUC for the holdout sample is 0.5391, and the accuracy of the holdout sample is 0.7344235. The predictors that appear to be most important for predicting Purchase are Closest, Spent, Visits, and PercentClose.

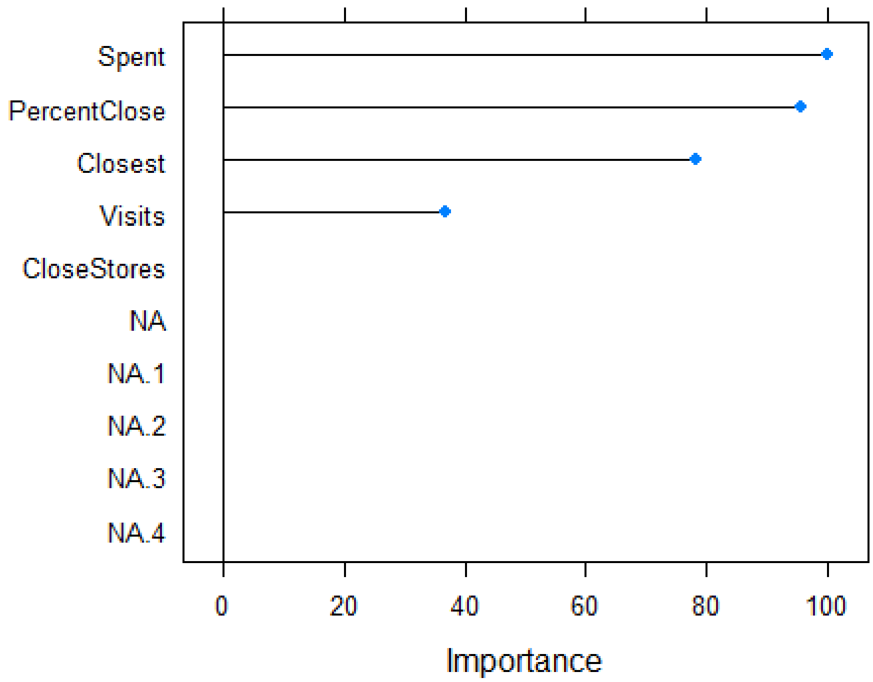
(e) Using the training data, trian a random forest model to predict `Purchase`. Audition two values of `mtry` of 1 and 5 (pure bagging). Report the estimated generalization metrics of the best model, as well as its accuracy and auc on the holdout sample. Also, report the variable importance plot, and comment on which predictors appear most important for predicting `Purchase`.

```
forestGrid <- expand.grid(mtry=c(1,5))
set.seed(474); FOREST2 <- train(Purchase~.,data=PURCHASE_TRAIN,method="rf",tu
neGrid=forestGrid,
                              trControl=fitControl,preProc=c("center","scale
"))
## Warning in train.default(x, y, weights = w, ...): The metric "Accuracy" wa
s not
## in the result set. ROC will be used instead.
FOREST2$results[rownames(FOREST2$bestTune),]
##   mtry       ROC       Sens      Spec      ROCSD     SensSD    SpecSD
## 1    1 0.6364147 0.04384314 0.9679374 0.06806732 0.03849275 0.0118973

set.seed(474); classification.forest2 <- predict(FOREST2,newdata=PURCHASE_HOL
DOUT)
mean(classification.forest2==PURCHASE_HOLDOUT$Purchase)
## [1] 0.7418329
set.seed(474); roc(PURCHASE_HOLDOUT$Purchase, predict(FOREST2,newdata=PURCHAS
E_HOLDOUT,type="prob")[,2])
## Setting levels: control = Buy, case = No
## Setting direction: controls < cases
##
## Call:
## roc.default(response = PURCHASE_HOLDOUT$Purchase, predictor = predict(FORE
ST2,     newdata = PURCHASE_HOLDOUT, type = "prob")[, 2])
##
## Data: predict(FOREST2, newdata = PURCHASE_HOLDOUT, type = "prob")[, 2] in
6640 controls (PURCHASE_HOLDOUT$Purchase Buy) < 20083 cases (PURCHASE_HOLDOUT
$Purchase No).
## Area under the curve: 0.608


IMP <-varImp(FOREST2)$importance
IMP$Variable <-rownames(IMP)
IMP <- IMP[order(IMP$Overall,decreasing=TRUE),]
head(IMP)
##                 Overall    Variable
## Spent         100.00000       Spent
## PercentClose   95.90545 PercentClose
## Closest        78.34374     Closest
```

```
## Visits        36.87581       Visits
## CloseStores    0.00000  CloseStores
plot(varImp(FOREST2), top=10)
```



RESPONSE: The estimated generalization AUC is 0.5619115. The AUC for the holdout sample is 0.6062, and the accuracy of the holdout sample is 0.7476705. The predictors that appear to be most important for predicting Purchase are Closest, Spent, Visits, and PercentClose.

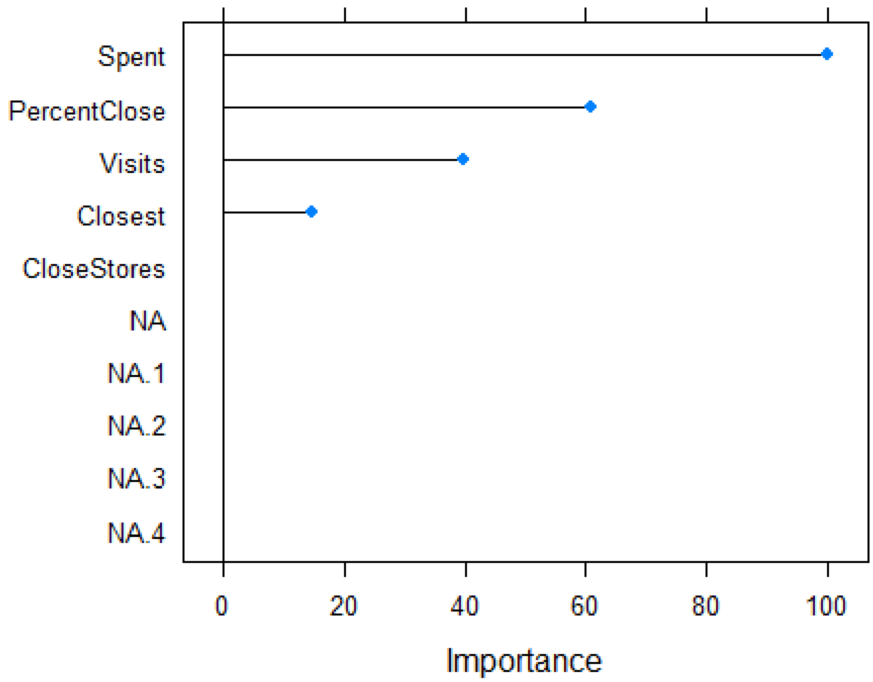
(f) Using the training data, train a gradient boosted tree model to predict `Purchase`. Report the estimated generalization metrics of the best model, as well as its accuracy and auc on the holdout sample. Also, report the variable importance plot, and comment on which predictors appear most important for predicting `Purchase`.

```
set.seed(474); GBM2 <- train(Purchase~.,data=PURCHASE_TRAIN,method="gbm",
                                    trControl=fitControl,preProc=c("center","scale"),
verbose=FALSE)
## Warning in train.default(x, y, weights = w, ...): The metric "Accuracy" wa
s not
## in the result set. ROC will be used instead.
GBM2$results[rownames(GBM2$bestTune),]
##   shrinkage interaction.depth n.minobsinnode n.trees       ROC   Sens  Spec
## 1       0.1                 1             10      50 0.6320067 0.012 0.996
##        ROCSD      SensSD      SpecSD
## 1 0.07886722 0.01788854 0.008944272

set.seed(474); classification.gbm2 <- predict(GBM2,newdata=PURCHASE_HOLDOUT)
```

```
mean(classification.gbm2==PURCHASE_HOLDOUT$Purchase)
## [1] 0.7509636
set.seed(474); roc(PURCHASE_HOLDOUT$Purchase, predict(GBM2,newdata=PURCHASE_H
OLDOUT,type="prob")[,2])
## Setting levels: control = Buy, case = No
## Setting direction: controls < cases
##
## Call:
## roc.default(response = PURCHASE_HOLDOUT$Purchase, predictor = predict(GBM2
,     newdata = PURCHASE_HOLDOUT, type = "prob")[, 2])
##
## Data: predict(GBM2, newdata = PURCHASE_HOLDOUT, type = "prob")[, 2] in 664
0 controls (PURCHASE_HOLDOUT$Purchase Buy) < 20083 cases (PURCHASE_HOLDOUT$Pu
rchase No).
## Area under the curve: 0.6168

IMP <-varImp(GBM2)$importance
IMP$Variable <-rownames(IMP)
IMP <- IMP[order(IMP$Overall,decreasing=TRUE),]
head(IMP)
##                 Overall     Variable
## Spent        100.00000        Spent
## PercentClose  60.81310 PercentClose
## Visits        39.77526       Visits
## Closest       14.67604      Closest
## CloseStores    0.00000  CloseStores
plot(varImp(GBM2), top=10)
```



RESPONSE: The estimated generalization AUC is 0.5780394. The AUC for the holdout sample is 0.6087, and the accuracy of the holdout sample is 0.749841. The predictors that appear to be most important for predicting Purchase are Spent, Visits, PercentClose, and Closest.

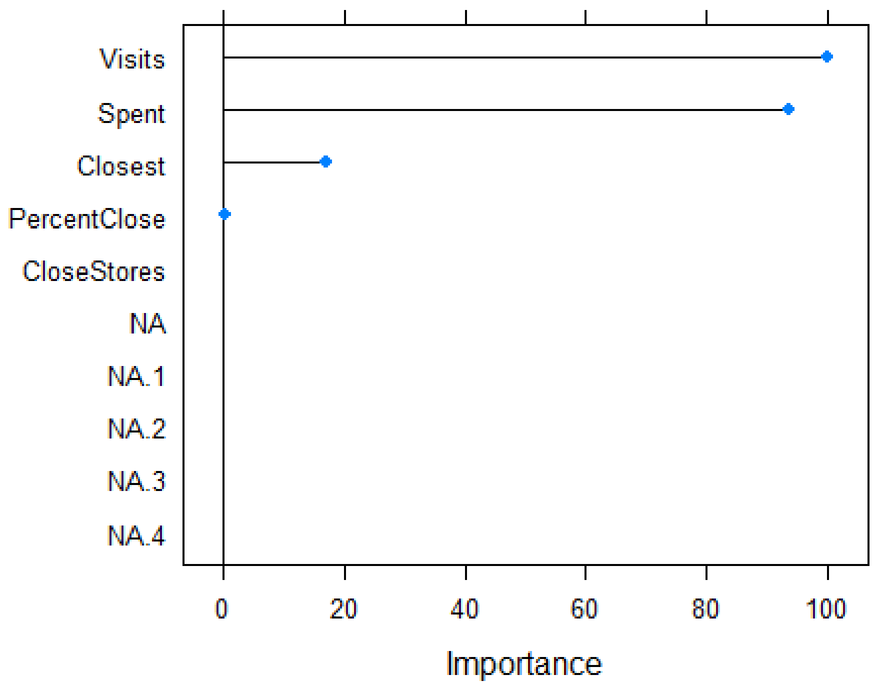
(g) Using the training data, train a support vector machine with a `radial basis kernel` to predict `Purchase`. Report the estimated generalization metrics of the best model, as well as its accuracy and auc on the holdout samples. Also, report the variable importance plot, and comment on which predictors appear most important for predicting `Purchase`.

```
set.seed(474); SVMradial <- train(Purchase~., data=PURCHASE_TRAIN, method="sv
mRadial",
                              trControl=fitControl, verbose=FALSE, preProc=c("
center", "scale"))
## Warning in train.default(x, y, weights = w, ...): The metric "Accuracy" wa
s not
## in the result set. ROC will be used instead.
SVMradial$results[rownames(SVMradial$bestTune),]
##       sigma C       ROC  Sens      Spec      ROCSD      SensSD      SpecSD
## 3 0.4511284 1 0.5718481 0.004 0.9986667 0.04128938 0.008944272 0.002981424

set.seed(474); classification.svm <- predict(SVMradial,newdata=PURCHASE_HOLDO
UT)
mean(classification.svm==PURCHASE_HOLDOUT$Purchase)
## [1] 0.7511133
set.seed(474); roc(PURCHASE_HOLDOUT$Purchase, predict(SVMradial,newdata=PURCH
```

```
ASE_HOLDOUT,type="prob")[,2])
## Setting levels: control = Buy, case = No
## Setting direction: controls < cases
##
## Call:
## roc.default(response = PURCHASE_HOLDOUT$Purchase, predictor = predict(SVMr
adial,     newdata = PURCHASE_HOLDOUT, type = "prob")[, 2])
##
## Data: predict(SVMradial, newdata = PURCHASE_HOLDOUT, type = "prob")[, 2] i
n 6640 controls (PURCHASE_HOLDOUT$Purchase Buy) < 20083 cases (PURCHASE_HOLDO
UT$Purchase No).
## Area under the curve: 0.5445

plot(varImp(SVMradial), top=10)
```



RESPONSE: The estimated generalization AUC is 0.5066999. The AUC for the holdout sample is 0.52, and the accuracy of the holdout sample is 0.7502526. The predictors that appear to be most important for predicting Purchase are Visits and Spent.

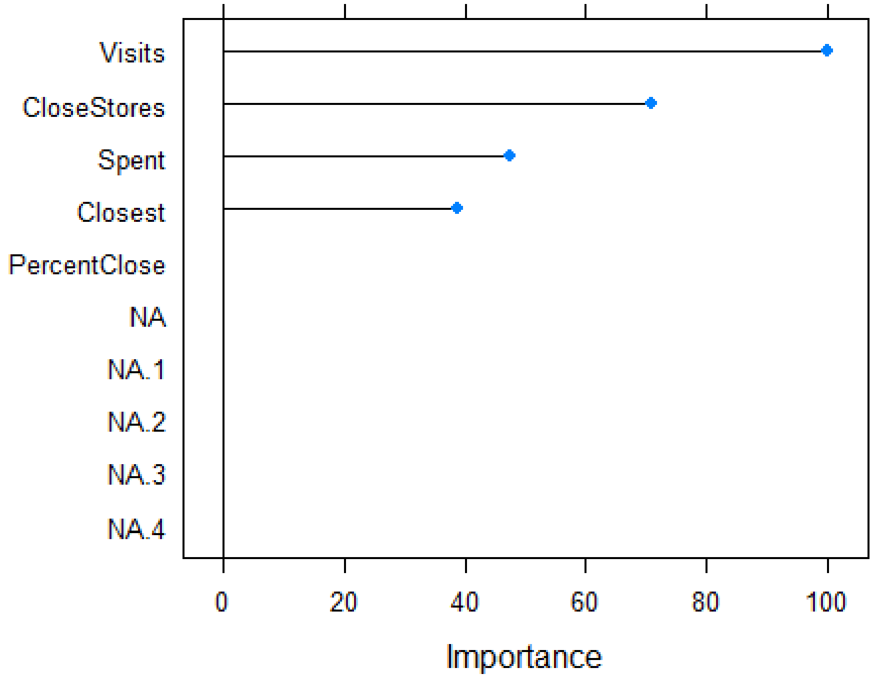
(h) Using the training data, train a neural network model with one hidden layer to predict `Purchase`. Audition number of nodes in 1-6 and decay of 10 raised to the -2, -1.5, ..., 0.5, 1 powers. Report the estimated generalization metrics of the best model, as well as the accuracy and auc on the holdout sample. Also, report the variable importance plot, and comment on which predictors appear most important for predicting `Purchase`.

```
nnetGrid <- expand.grid(size=1:6,decay=c(10^seq(-2,1,by=0.5) ) )
set.seed(474); NNET2 <- train(Purchase~., data=PURCHASE_TRAIN, hidden=1, meth
```

```
od='nnet', trControl=fitControl,
                             tuneGrid=nnetGrid, preProc = c("center", "scale"
),
                             verbose=FALSE, trace=FALSE,linout=FALSE)
## Warning in train.default(x, y, weights = w, ...): The metric "Accuracy" wa
s not
## in the result set. ROC will be used instead.
NNET2$results[rownames(NNET2$bestTune),]
##    size      decay       ROC       Sens      Spec      ROCSD     SensSD
## 16    3 0.03162278 0.6182185 0.02792157 0.9799821 0.04474965 0.02283855
##      SpecSD
## 16 0.016988

set.seed(474); classification.nnet2 <- predict(NNET2,newdata=PURCHASE_HOLDOUT
)
mean(classification.nnet2==PURCHASE_HOLDOUT$Purchase)
## [1] 0.7456498
set.seed(474); roc(PURCHASE_HOLDOUT$Purchase, predict(NNET2,newdata=PURCHASE_
HOLDOUT,type="prob")[,2])
## Setting levels: control = Buy, case = No
## Setting direction: controls < cases
##
## Call:
## roc.default(response = PURCHASE_HOLDOUT$Purchase, predictor = predict(NNET
2,     newdata = PURCHASE_HOLDOUT, type = "prob")[, 2])
##
## Data: predict(NNET2, newdata = PURCHASE_HOLDOUT, type = "prob")[, 2] in 66
40 controls (PURCHASE_HOLDOUT$Purchase Buy) < 20083 cases (PURCHASE_HOLDOUT$P
urchase No).
## Area under the curve: 0.6098

IMP <-varImp(NNET2)$importance
IMP$Variable <-rownames(IMP)
IMP <- IMP[order(IMP$Overall,decreasing=TRUE),]
head(IMP)
##                Overall    Variable
## Visits       100.00000      Visits
## CloseStores   71.03920 CloseStores
## Spent         47.53500       Spent
## Closest       38.73402     Closest
## PercentClose   0.00000 PercentClose
plot(varImp(NNET2), top=10)
```



RESPONSE: The estimated generalization AUC is 0.6160569. The AUC for the holdout sample is 0.6233, and the accuracy of the holdout sample is 0.7502526. The only predictor that appears to be important for predicting Purchase is Visits.

(i) Is one model a compelling choice for predicting `Purchase` over the others? Why or why not? Show evidence to support your claim.

```
GLM2$results
##   parameter       ROC       Sens      Spec      ROCSD     SensSD      Spec
SD
## 1      none 0.5827802 0.01984314 0.9866488 0.08689495 0.01414377 0.0094281
75
GLMnet2$results[rownames(GLMnet2$bestTune),]
##    alpha     lambda       ROC  Sens  Spec      ROCSD     SensSD      SpecS
D
## 92   0.8 0.03162278 0.6262559 0.008 0.996 0.07445817 0.01095445 0.00894427
2
TREE2$results[rownames(TREE2$bestTune),]
##            cp       ROC      Sens      Spec      ROCSD     SensSD      Spec
SD
## 2 0.006613757 0.5642159 0.1705882 0.8704161 0.06042947 0.04577665 0.060303
75
FOREST2$results[rownames(FOREST2$bestTune),]
##   mtry       ROC       Sens      Spec      ROCSD     SensSD    SpecSD
## 1    1 0.6364147 0.04384314 0.9679374 0.06806732 0.03849275 0.0118973
GBM2$results[rownames(GBM2$bestTune),]
##   shrinkage interaction.depth n.minobsinnode n.trees       ROC  Sens  Spec
```

```
## 1       0.1                 1             10      50 0.6320067 0.012 0.996
##        ROCSD     SensSD      SpecSD
## 1 0.07886722 0.01788854 0.008944272
SVMradial$results[rownames(SVMradial$bestTune),]
##       sigma C       ROC  Sens      Spec      ROCSD      SensSD      SpecSD
## 3 0.4511284 1 0.5718481 0.004 0.9986667 0.04128938 0.008944272 0.002981424
NNET2$results[rownames(NNET2$bestTune),]
##    size      decay       ROC       Sens      Spec      ROCSD     SensSD
## 16    3 0.03162278 0.6182185 0.02792157 0.9799821 0.04474965 0.02283855
##      SpecSD
## 16 0.016988
```

RESPONSE: Again, there is no clear winner here as most of the models' ROC's are within 1 standard deviation of each other. However, we can rule out the SVMradial model as its ROC is well below of 1 standard deviation of most models. Also, it should be noted that it is never a good idea to use AUC to evaluate SVM models. We can also likely rule out the random forest model as it is below of 1 standard deviation of a couple of models. That being said, I do not see a definative "best model" here.