Homework 09

STAT 430, Fall 2017

Due: Monday, November 20, 11:59 PM

Please see the homework instructions document for detailed instructions and some grading notes. Failure to follow instructions will result in point reductions.

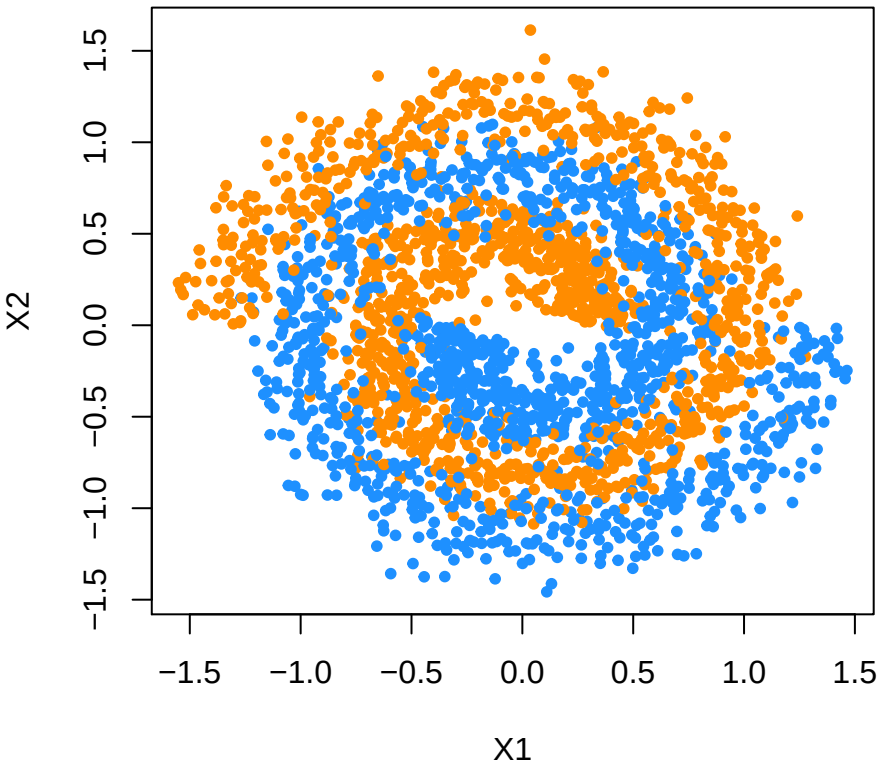
Please note the altered due date.

Exercise 1 (Computation Time)

[**8 points**] For this exercise we will create data via simulation, then assess how well certain methods perform. Use the code below to create a train and test dataset.

```
library(mlbench)
set.seed(42)
sim_trn = mlbench.spirals(n = 2500, cycles = 1.5, sd = 0.125)
sim_trn = data.frame(sim_trn$x, class = as.factor(sim_trn$classes))
sim_tst = mlbench.spirals(n = 10000, cycles = 1.5, sd = 0.125)
sim_tst = data.frame(sim_tst$x, class = as.factor(sim_tst$classes))
```

The training data is plotted below, with colors indicating the `class` variable, which is the response.



Before proceeding further, set a seed equal to your UIN.

```
uin = 123456789
set.seed(uin)
```

We'll use the following to define 5-fold cross-validation for use with `train()` from `caret`.

```
library(caret)
cv_5 = trainControl(method = "cv", number = 5)
```

We now tune two models with `train()`. First, a logistic regression using `glm`. (This actually isn't "tuned" as there are not parameters to be tuned, but we use `train()` to perform cross-validation.) Second we tune a single decision tree using `rpart`.

We store the results in `sim_glm_cv` and `sim_tree_cv` respectively, but we also wrap both function calls with `system.time()` in order to record how long the tuning process takes for each method.

```
glm_cv_time = system.time({
  sim_glm_cv  = train(
    class ~ .,
    data = sim_trn,
    trControl = cv_5,
    method = "glm")
})

tree_cv_time = system.time({
  sim_tree_cv = train(
    class ~ .,
    data = sim_trn,
    trControl = cv_5,
    method = "rpart")
})
```

We see that both methods are tuned via cross-validation in a similar amount of time.

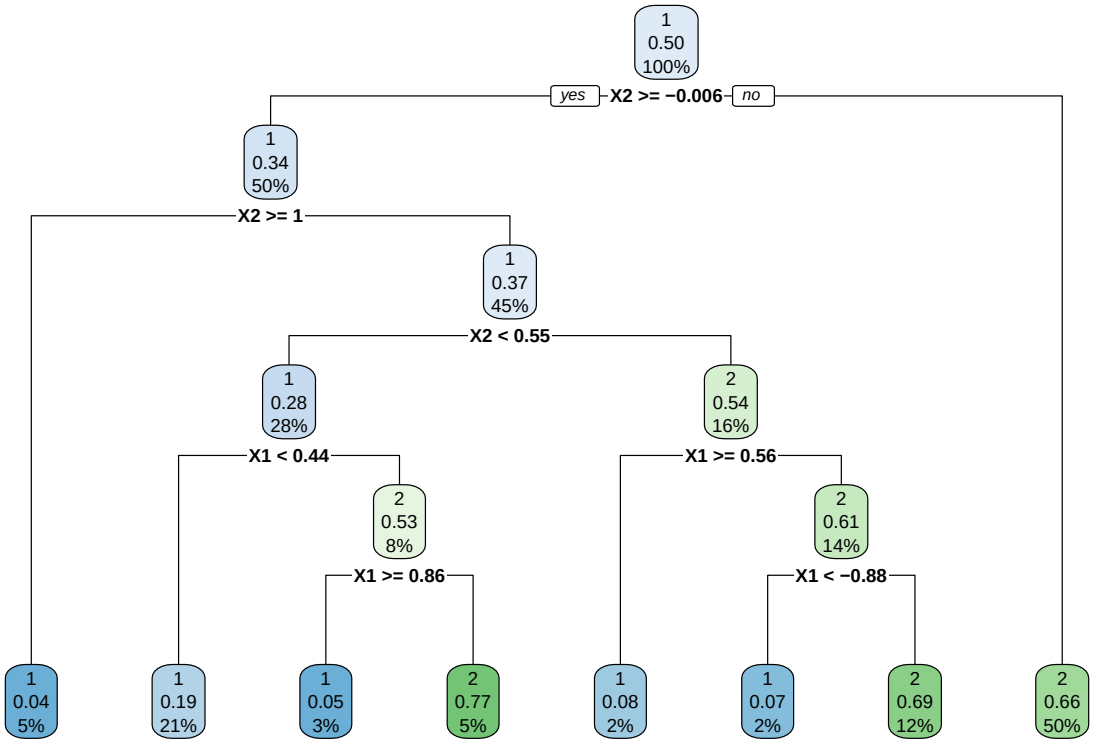
```
glm_cv_time["elapsed"]
```

```
## elapsed
##   0.963
```

```
tree_cv_time["elapsed"]
```

```
## elapsed
##   0.666
```

```
library(rpart.plot)
rpart.plot(sim_tree_cv$finalModel)
```



Repeat the above analysis using a random forest, twice. The first time use 5-fold cross-validation. (This is how we had been using random forests before we understood random forests.) The second time, tune the model using OOB samples. We only have two predictors here, so, for both, use the following tuning grid.

```
rf_grid = expand.grid(mtry = c(1, 2))
```

Create a table summarizing the results of these four models. (Logistic with CV, Tree with CV, RF with OOB, RF with CV). Report:

- Chosen value of tuning parameter (If applicable)
- Elapsed tuning time
- Resampled (CV or OOB) Accuracy
- Test Accuracy

Exercise 2 (Predicting Baseball Salaries)

[7 points] For this question we will predict the `Salary` of `Hitters`. (`Hitters` is also the name of the dataset.) We first remove the missing data:

```
library(ISLR)
Hitters = na.omit(Hitters)
```

After changing `uin` to your UIN, use the following code to test-train split the data.

```
uin = 123456789
set.seed(uin)
hit_idx = createDataPartition(Hitters$Salary, p = 0.6, list = FALSE)
hit_trn = Hitters[hit_idx,]
hit_tst = Hitters[-hit_idx,]
```

Do the following:

- Tune a boosted tree model using the following tuning grid and 5-fold cross-validation.

```
gbm_grid = expand.grid(interaction.depth = c(1, 2),
                       n.trees = c(500, 1000, 1500),
                       shrinkage = c(0.001, 0.01, 0.1),
                       n.minobsinnode = 10)
```

- Tune a random forest using OOB resampling and **all** possible values of `mtry`.

Create a table summarizing the results of three models:

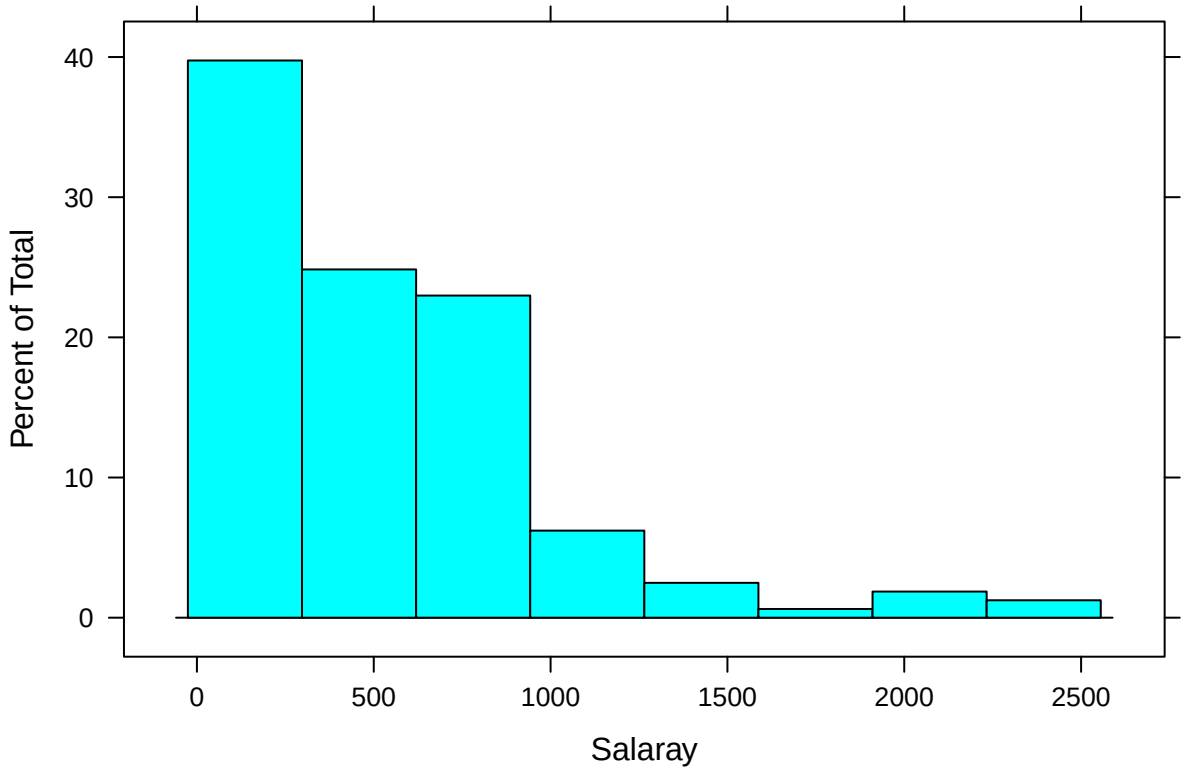
- Tuned boosted tree model
- Tuned random forest model
- Bagged tree model

For each, report:

- Resampled RMSE
- Test RMSE

Exercise 3 (Transforming the Response)

[**5 points**] Continue with the data from Exercise 2. The book, ISL, suggests log transforming the response, `Salary`, before fitting a random forest. Is this necessary? Re-tune a random forest as you did in Exercise 2, except with a log transformed response. Report test RMSE for both the untransformed and transformed model on the original scale of the response variable.



Exercise 4 (Concept Checks)

[**1 point each**] Answer the following questions based on your results from the three exercises.

Timing

(a) Compare the time taken to tune each model. Is the difference between the OOB and CV result for the random forest similar to what you would have expected?

(b) Compare the tuned value of `mtry` for each of the random forests tuned. Do they choose the same model?

(c) Compare the test accuracy of each of the four procedures considered. Briefly explain these results.

Salary

(d) Report the tuned value of `mtry` for the random forest.

(e) Create a plot that shows the tuning results for the tuning of the boosted tree model.

(f) Create a plot of the variable importance for the tuned random forest.

(g) Create a plot of the variable importance for the tuned boosted tree model.

(h) According to the random forest, what are the three most important predictors?

(i) According to the boosted model, what are the three most important predictors?

Transformation

(j) Based on these results, do you think the transformation was necessary?