HW5: Programming for Model Comparison

Mike Maccia

Task 1: Conceptual Questions

- What is the purpose of using cross-validation when fitting a random forest model?

Cross-validation is used when fitting a random forest model in an effect to prevent overfitting. Since random forest models are ensembles of decision trees, where each decision tree model should be created on different versions of the training set. We still want to only train our model using training data (preventing the model from seeing the test data), so using cross-validation allows us to create multiple folds of the data so each tree can be created on different versions of the training data.

These individual trees are combined into a random forest.

- Describe the bagged tree algorithm.

The bagged tree algorithm is a method of using bootstrapping to get multiple samples to fit on the same model. The results of these models would then be averaged together across multiple trees to create the final model. This allows for decreased variance when compared to an individual tree fit model.

Bootstrapping is a method of using resampling of data to create many samples. You would treat each sample as the population and this would then be used to fit the regression tree. The individual sample "y"s are then combined together to create the final model/prediction. In resampling of the data, normally it is done with replacement, so the same observation could be pulled into each individual sample multiple times.

- What is meant by a general linear model?

General linear models (GLM) are a form of modeling in which the response variable could be binary (success or failure, yes/no, etc). The GLM models are considered more flexible since they allow for this flexibility of binary variable. One example of a GLM is logistic regression in which the predicted probabilities of the response variable are a log function and would fall between 0 and 1. When graphing, this takes on a sigmoid shape. Even though the response variable is binary, continuous variables are able to be part of the prediction formula.

- When fitting a multiple linear regression model, what does adding an interaction term do? That is, what does it allow the model to do differently as compared to when it is not included in the model?

Adding an interaction effect allows for a more flexible surface when modeling a multiple linear regression. As opposed to a line of linear regression (plotted as x vs y), a multiple linear regression model would be plotted on more axes (in 3-dimensions). The model would instead be plotted as a "plane" and could take on a flat or saddle-like shape. An example of an interaction could be seen using this equation:

$$E[Y \mid x_1, x_2] = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \beta_3 (x_1 \cdot x_2)$$

In this instance the effect of x1 on Y differs based on the value of x2.

- Why do we split our data into a training and test set?

When creating predictive models, our goal is for our model to predict well for observations the model has never seen before. If we train our model on all of our data, there is a risk of overfitting and the model would not perform well on data it has not seen before.

In order to prevent overfitting, we create a training and test set by randomly splitting the data. Normally the data is split into either 80/20 or 70/30 training / test sets. The training data set is what we use to train the model. After, we can then predict using the test set and use a metric to determine how well our model worked.

Task 2: Data Prep

Loading packages and data

First the needed packages

```
suppressPackageStartupMessages(library(tidyverse))
suppressPackageStartupMessages(library(tidymodels))
suppressPackageStartupMessages(library(caret))
suppressPackageStartupMessages(library(yardstick))
```

Now let's load in the data. Looks like we will be looking at heart disease (`HeartDisease = 1` for patients with heart disease and `0` for patients who do not) along with different measurements of someone's health.

```
heart <- read.csv("heart.csv", header = TRUE)
heart_data <- as_tibble(heart) #make the data set a tibble
```

Let's first look at the dataset using the `summary()` function

```
summary(heart_data)
```

```
      Age            Sex            ChestPainType        RestingBP
 Min.   :28.00   Length:918         Length:918         Min.   :  0.0
 1st Qu.:47.00   Class :character   Class :character   1st Qu.:120.0
 Median :54.00   Mode  :character   Mode  :character   Median :130.0
 Mean   :53.51                                         Mean   :132.4
 3rd Qu.:60.00                                         3rd Qu.:140.0
 Max.   :77.00                                         Max.   :200.0
  Cholesterol      FastingBS       RestingECG            MaxHR
 Min.   :  0.0   Min.   :0.0000   Length:918         Min.   : 60.0
 1st Qu.:173.2   1st Qu.:0.0000   Class :character   1st Qu.:120.0
 Median :223.0   Median :0.0000   Mode  :character   Median :138.0
 Mean   :198.8   Mean   :0.2331                      Mean   :136.8
 3rd Qu.:267.0   3rd Qu.:0.0000                      3rd Qu.:156.0
 Max.   :603.0   Max.   :1.0000                      Max.   :202.0
 ExerciseAngina        Oldpeak          ST_Slope          HeartDisease
 Length:918         Min.   :-2.6000   Length:918         Min.   :0.0000
 Class :character   1st Qu.: 0.0000   Class :character   1st Qu.:0.0000
 Mode  :character   Median : 0.6000   Mode  :character   Median :1.0000
                    Mean   : 0.8874                      Mean   :0.5534
                    3rd Qu.: 1.5000                      3rd Qu.:1.0000
                    Max.   : 6.2000                      Max.   :1.0000
```

What type of varible (in R) is Heart Disease? Categorical or Quantitative?

The heart disease variable is quantitative in R.

Does this make sense? Why or why not?

No it does not make sense that heart disease is quantitative, since it really is not a numeric value. We only want to determine if someone has heart disease or not. It is current stored as an integer, but it is really a binary value, where 1 = success (or patient has heart disease) while 0 = failure (or the patient does not have heart disease). We do not want it to be considered numeric since the mean of the responses does not necessarily make sense when modeling in the same way we would model using say the mean heart rate value.

Adjusting the data a bit

We will adjust the `heartdisease` variable to make it binary and format is as a double. This should help with future modeling

We will create a final tibble for this step and call it `new_hear` where the old `heartdisease` variable and also `ST_slope` will be removed.

```
new_heart <- heart_data |>
  mutate(heart_disease_dbl = as.double(HeartDisease)) |> #make heart disease a double
  select(-HeartDisease, -ST_Slope) |> #remove original heartdisease and ST_slope
  relocate(heart_disease_dbl) #move new heart disease variable to front

new_heart
```

```
# A tibble: 918 x 11
   heart_disease_dbl   Age Sex   ChestPainType RestingBP Cholesterol FastingBS
               <dbl> <int> <chr> <chr>              <int>       <int>     <int>
 1                 0    40 M     ATA                  140         289         0
 2                 1    49 F     NAP                  160         180         0
 3                 0    37 M     ATA                  130         283         0
 4                 1    48 F     ASY                  138         214         0
 5                 0    54 M     NAP                  150         195         0
 6                 0    39 M     NAP                  120         339         0
 7                 0    45 F     ATA                  130         237         0
 8                 0    54 M     ATA                  110         208         0
 9                 1    37 M     ASY                  140         207         0
10                 0    48 F     ATA                  120         284         0
# i 908 more rows
# i 4 more variables: RestingECG <chr>, MaxHR <int>, ExerciseAngina <chr>,
#   Oldpeak <dbl>
```