Homework5 Models in R

Makenna Meyer

Task One: Conceptual Questions

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

Cross-validation allows us to make use of the entire dataset that we have access to. It does this by creating equal sections of data (called "folds") and alternating which single fold is used for testing and what folds are grouped together for training. This reduces the chance of getting a weird model due to a weird training/testing split and gives a less variable estimate of the model since it uses more (all) of the data.

Question Two: Describe the bagged tree algorithm.

A bagged tree algorithm means that bootstrap aggregation is used in addition to the standard tree algorithm. Bootstrapping means that we are resampling either from the data (non-parametric) or from a fitted model (parametric). For each resample, we have a certain fitted tree. We can then use each tree to come up with a prediction, and then we can take the average of all of the predictions from all of our trees to create a final prediction.

Question Three: What is meant by a general linear model?

A general linear model does not necessarily assume that there is a linear relationship between the explanatory variable(s) and the response, but it assumes that there is a linear relationship between the explanatory variable(s) and the transformed response. This transformation is also referred to as the link function.

Question Four: 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?

An interaction term allows the model to account for the fact that the impact of an explanatory variable A on the response Y may be different at different levels of another explanatory variable B. Otherwise, the impacts of the two explanatory variables A and B will be independent and constant regardless of each other's values.

Question Five: Why do we split our data into a training and test set?

Splitting into a training and test set is important to maintain independence. If a model is trained and tested on the same data, it is obvious that the model will be a good fit to that data. We are actually curious about how the model performs on new data, so it is important to choose either to use the data for training or for testing but not both.

Task Two: Data Prep

Packages and data

```r
#loading required packages
library("tidyverse")
library("tidymodels")
library("caret")
library("yardstick")

#reading in the heart data
heartdata <- read_csv("https://www4.stat.ncsu.edu/~online/datasets/heart.csv")
```

Question One: Summarizing the heart data

```r
summary(heartdata) #creating a summary for all variables in the 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  
```

a. What type of variable (in R) is Heart Disease? Categorical or Quantitative?

Heart Disease is a numerical (integer) variable. It is quantitative rather than categorical, and R is assuming that all numeric values between 0 and 1 are valid. We know this because a numerical summary is presented for the Heart Disease variable.

b. Does this make sense? Why or why not.

This does not make sense because a person has either been diagnosed with heart disease or they have not. Therefore, Heart Disease should be a categorical variable rather than a numeric. Heart Disease should be a factor variable with the levels 1 (indicating the person has been diagnosed with heart disease) and 0 (indicating the person has not been diagnosed with heart disease).

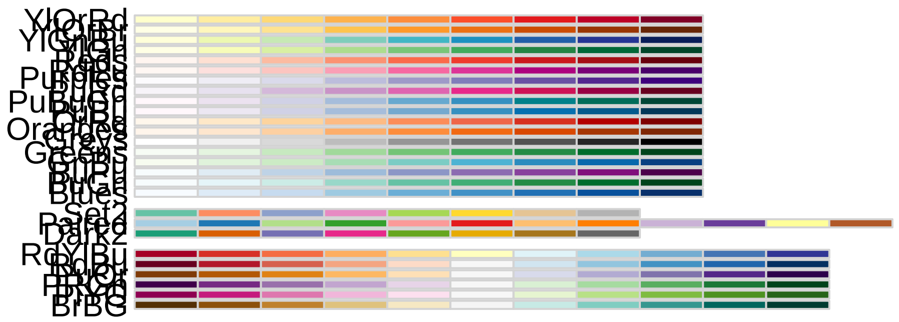
Question Two: Managing and cleaning the heart data

```r
#Creating a new heart tibble
new_heart <- heartdata |> #using the initial tibble
  mutate(new_hd = as.factor(HeartDisease)) |> #creating a new factor heart disease variable
  select(-c(ST_Slope, HeartDisease)) #removing ST_Slope and old heart disease variable
```

Task Three: EDA

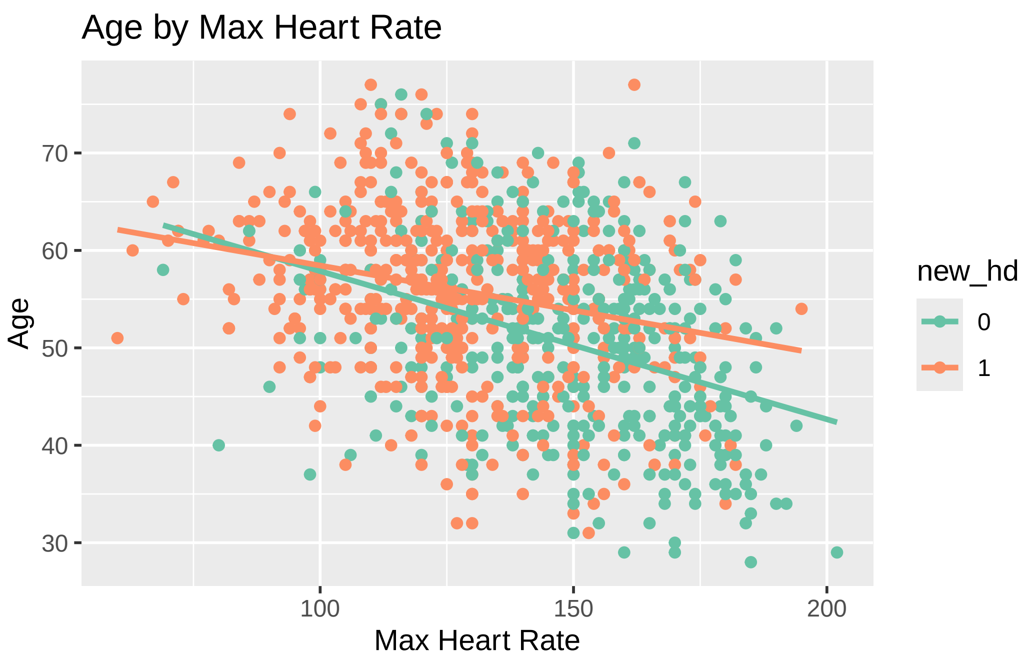
Question One: Creating a Scatter Plot Corresponding to the Heart Disease Model

```r
#Identifying potential colorblind friendly palettes from R Color Brewer
library("RColorBrewer")
display.brewer.all(colorblindFriendly = TRUE)
```



```r
g <- ggplot(new_heart, aes(x = MaxHR, y = Age, color = new_hd)) + #Plot Base
  geom_point() + #Creating a scatterplot
  geom_smooth(method=lm, se=FALSE) + #Adding lines
  labs(title = "Age by Max Heart Rate", x = "Max Heart Rate", y = "Age") + #Labels
  scale_color_brewer(palette = "Set2") #Colorblind friendly palette from R Color Brewer
g
```

```
`geom_smooth()` using formula = 'y ~ x'
```



Question Two: Based on visual evidence, do you think an interaction model or an additive model is more appropriate? Justify your answer.

Based on this plot, an interaction model is more appropriate because the two lines are not parallel. That indicates that whether or not heart disease is present impacts the effect that maximum heart rate has on our response variable age, and therefore an interaction term is necessary.

Task Four: Testing and Training

```r
set.seed(101) #setting a random seed so this process can be duplicated
heart_split <- initial_split(new_heart, prop = 0.80) #specifying a 80/20 split
heart_train <- training(heart_split) #splitting training data
heart_test <- testing(heart_split) #splitting testing data
```

Task Five: OLS and LASSO

Question One: Fitting an OLS Interation Model

```r
#fitting an interaction model using the training data
ols_mlr <- lm(Age ~ MaxHR + new_hd + MaxHR:new_hd, data = heart_train)

#displaying the model summary
summary(ols_mlr)
```

```

Call:
lm(formula = Age ~ MaxHR + new_hd + MaxHR:new_hd, data = heart_train)

Residuals:
     Min       1Q   Median       3Q      Max 
-22.7703  -5.7966   0.4516   5.7772  20.6378 

Coefficients:
              Estimate Std. Error t value Pr(>|t|)    
(Intercept)   75.58896    3.07510  24.581  < 2e-16 ***
MaxHR         -0.16992    0.02064  -8.233 8.43e-16 ***
new_hd1       -8.58502    3.83433  -2.239  0.02546 *  
MaxHR:new_hd1  0.08343    0.02716   3.072  0.00221 ** 
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 8.478 on 730 degrees of freedom
Multiple R-squared:  0.1839,	Adjusted R-squared:  0.1806 
F-statistic: 54.84 on 3 and 730 DF,  p-value: < 2.2e-16
```

Question Two: Calculating the RMSE for the OLS Interaction Model

```r
#Calculating RMSE using the testing data
RMSE_calc <- sqrt(mean((heart_test$Age - predict(ols_mlr, newdata=heart_test))^2))
RMSE_calc
```

```
[1] 9.100206
```

Question Three: Fitting a LASSO Model

```
heart_cv_folds <- vfold_cv(heart_train, 10) #creating 10 fold CV of the training data

LASSO_recipe <- recipe(Age ~ MaxHR + new_hd, data = heart_train) |>
  step_normalize(MaxHR) |> #standardizing the quantitative variable
  step_dummy(new_hd) |> #standardizing the categorical variable
  step_interact(~MaxHR:starts_with("new_hd")) #creating the interaction

LASSO_recipe
```

```
-- Recipe ----------------------------------------------------------------------



-- Inputs

Number of variables by role

outcome:   1
predictor: 2



-- Operations

* Centering and scaling for: MaxHR

* Dummy variables from: new_hd

* Interactions with: MaxHR:starts_with("new_hd")
```

Question Four: Now, set up your appropriate spec, and grid. Next, select your final model, and report the results using the tidy() function around your model name.

```r
#Creating LASSO spec
LASSO_spec <- linear_reg(penalty = tune(), mixture = 1) |>
  set_engine("glmnet")

#Creating LASSO workflow
LASSO_wkf <- workflow () |>
  add_recipe(LASSO_recipe) |>
  add_model(LASSO_spec)

#Creating LASSO grid
LASSO_grid <- LASSO_wkf |>
  tune_grid(resamples = heart_cv_folds,
            grid = grid_regular(penalty(), levels = 200))
```

```
Warning: package 'glmnet' was built under R version 4.3.3
```

```r
#Finding the lowest RMSE for all 200 LASSO models
lowest_rmse <- LASSO_grid |>
  select_best(metric = "rmse")

#Fitting the best model on the entire training dataset
LASSO_final <- LASSO_wkf |>
  finalize_workflow(lowest_rmse) |>
  fit(heart_train)

tidy(LASSO_final)
```

```
# A tibble: 4 x 3
  term                estimate penalty
  <chr>                  <dbl>   <dbl>
1 (Intercept)            52.5   0.0174
2 MaxHR                  -4.22  0.0174
3 new_hd_X1               2.75  0.0174
4 MaxHR_x_new_hd_X1       2.00  0.0174
```

Question Five: Without looking at the RMSE calculations, would you expect the RMSE calculations to be roughly the same or different? Justify your answer using output from your LASSO model.

Without looking, I would expect the RMSE values very similar because the tuning parameter penalty is just 0.017 so it is fairly close to 0.

Question Six: Now compare the RMSE between your OLS and LASSO model and show that the RMSE calculations were roughly the same.

The RMSE from the OLS Interaction Model was

```r
#Reprinting saved calculated RMSE value for the OLS model
RMSE_calc
```

```
[1] 9.100206
```

The RMSE from the LASSO model is

```r
#Calculating the RMSE of the LASSO model
LASSO_final |>
  predict(heart_test) |>
  pull() |>
  rmse_vec(truth = heart_test$Age)
```

```
[1] 9.091133
```

These values are extremely similar (off by just about 0.01)

Question Seven: Why are the RMSE calculations roughly the same if the coefficients for each model are different?

The RMSE calculation are still roughly the same because the tuning parameter value is very close to 0 so the coefficients shouldn't have changed very much.

Task Six: Logistic Regression

Question One: Proposing two different logistic models

```r
#Creating the spec for logistic models
LR_spec <- logistic_reg() |>
  set_engine("glm")
```

```r
#Creating the first logistic model using
#Resting Blood Pressure, Sex, and Cholesterol as predictors
LR1_rec <- recipe(new_hd ~ RestingBP + Cholesterol + Sex, data = heart_train) |>
  step_normalize(all_numeric()) |>
  step_dummy(Sex)

LR1_wkf <- workflow() |>
  add_recipe(LR1_rec) |>
  add_model(LR_spec)

LR1_fit <- LR1_wkf |>
  fit_resamples(heart_cv_folds, metrics = metric_set(accuracy, mn_log_loss))
```

```r
#Creating the first logistic model using
#Max heart rate, Age, and Sex as predictors
LR2_rec <- recipe(new_hd ~ MaxHR + Age + Sex, data = heart_train) |>
  step_normalize(Age, MaxHR) |>
  step_dummy(Sex)

LR2_wkf <- workflow() |>
  add_recipe(LR2_rec) |>
  add_model(LR_spec)

LR2_fit <- LR2_wkf |>
  fit_resamples(heart_cv_folds, metrics = metric_set(accuracy, mn_log_loss))
```

```r
rbind(LR1_fit |> collect_metrics(),
      LR2_fit |> collect_metrics()) |>
  mutate(Model = c("Model1", "Model1", "Model2", "Model2")) |>
  select(Model, everything())
```

```
# A tibble: 4 x 7
```

```
  Model  .metric     .estimator  mean     n std_err .config
  <chr>  <chr>       <chr>      <dbl> <int>   <dbl> <chr>
1 Model1 accuracy    binary     0.654    10  0.0167 Preprocessor1_Model1
2 Model1 mn_log_loss binary     0.616    10  0.0150 Preprocessor1_Model1
3 Model2 accuracy    binary     0.700    10  0.0134 Preprocessor1_Model1
4 Model2 mn_log_loss binary     0.567    10  0.0169 Preprocessor1_Model1
```

Based on the accuracy and log loss metrics above, Model 2 is a better model. This is because it has both a higher accuracy and a lower log loss. Model 2 predicts heart disease based off of Max heart rate, age, and sex. Model 2 only shared the sex predictor in common with Model 1 which used resting blood pressure and cholesterol as predictors. This is a significant finding because it indicates that Max HR and Age are potentially better predictors than Blood Pressure and Cholesterol.

Question Two: Lastly, check how well your chosen model does on the test set using the confusionMatrix() function.

```
LR_train_fit <- LR2_wkf |>
  fit(heart_train)

conf_mat(heart_test |>
           mutate(estimate = LR_train_fit |>
                    predict(heart_test) |>
                    pull()),
         new_hd,
         estimate)
```

```
          Truth
Prediction  0  1
         0 62 20
         1 32 70
```

This confusion matrix shows that my model correctly predicted no heart disease diagnosis in 62 out of 94 cases. My model correctly predicted a heart disease diagnosis in 70 out of 90 heart disease cases. My model seems to do about as well at predicting heart disease cases vs predicting non-heart disease cases, but is maybe slightly better at predicting heart disease cases.

Question Three: Next, identify the values of sensitivity and specificity, and interpret them in the context of the problem.

Sensitivity is probability that a test correctly detects a disease when it is truly present and is calculated by (# positive and test positive)/# total positive. In this context, sensitivity describes the probability of the model accurately detecting heart disease when a person is known to be diagnosed with heart disease. My model correctly predicted a heart disease diagnosis in 70 out of 90 heart disease cases. This is a sensitivity rate of 70/90 = 0.7778 = 77.78%. Sensitivity is the probability that a test correctly does not detect disease when it is truly absent and is calculated by (# negative and test negative)/# total negative. In this context, specificity describes the probability of the model accurately not detecting heart disease when a person is known to not be diagnosed with heart disease. This confusion matrix shows that my model correctly predicted no heart disease diagnosis in 62 out of 94 cases. This is a specificity rate of 62/94 = 0.6596 = 65.96%.