Problem Set 9

Classification

[YOUR NAME]

Due Date: 2024-03-29

Getting Set Up

Open `RStudio` and create a new RMarkDown file (`.Rmd`) by going to `File -> New File -> R Markdown...` . Accept defaults and save this file as `[LAST NAME]_ps9.Rmd` to your `code` folder.

Copy and paste the contents of this `.Rmd` file into your `[LAST NAME]_ps9.Rmd` file. Then change the `author: [Your Name]` to your name.

We will be using the `fn_cleaned_final.Rds` file from the course github page (https://github.com/jbisbee1/DS1000_S2024/raw/main/data/fn_cleaned_final.rds).

All of the following questions should be answered in this `.Rmd` file. There are code chunks with incomplete code that need to be filled in.

This problem set is worth 8 total points, plus two extra credit points. The point values for each question are indicated in brackets below. To receive full credit, you must have the correct code. In addition, some questions ask you to provide a written response in addition to the code.

You are free to rely on whatever resources you need to complete this problem set, including lecture notes, lecture presentations, Google, your classmates…you name it. However, the final submission must be complete by you. There are no group assignments. To submit, compiled the completed problem set and upload the PDF file to Brightspace on Friday by midnight. Also note that the TAs and professors will not respond to Campuswire posts after 5PM on Friday, so don't wait until the last minute to get started!

Good luck!

*Copy the link to ChatGPT you used here: _____________________

Question 0

Require `tidyverse` and load the `fn_cleaned_final.Rds` (https://github.com/jbisbee1/DS1000_S2024/raw/main/data/fn_cleaned_final.rds) data to an object called `fn` .

```
# INSERT CODE HERE
```

Question 1 [2 points]

Let's consider two possible \(X\) variables which might help us predict whether a player wins a Fortnite match: `revives` and `eliminations` . `revives` counts the total number of times a player is brought back to life by a teammate. `eliminations` is a measure of how many times the player killed an opponent. Which variable do you think is more helpful for predicting whether a player wins a game of Fortnite? Why?

> Write response here.

Question 2 [2 points]

Look at the data and provide univariate and multivariate visualizations of both variables. Make sure to think carefully about what types of variables these are, and justify your visualization choices accordingly!

```
# Look to determine variable types
# INSERT CODE HERE

# Univariate #1
# INSERT CODE HERE

# Univariate #1
# INSERT CODE HERE

# Multivariate (many different options will work)
# INSERT CODE HERE
```

Question 3 [2 points]

Let’s test your intuition. Starting with the full data, calculate the AUC for both models. Then, using 100 cross validation with a logit model and a 60-40 split, calculate the AUC for the model which uses the variable you think is best, compared to the model you think is the worst. Pay attention to the things you need to change to use a logit model! Is your assumption from Q1 supported in the data?

```r
# Require the tidymodels package
# Running logit model #1
# INSERT CODE HERE

# Running logit model #2
# INSERT CODE HERE

# Calculate the AUC #1
# INSERT CODE HERE

# Calculate the AUC #1
# INSERT CODE HERE


# Calculate cross validation
set.seed(123)
cvRes <- NULL
for(i in 1:100) {
  # Cross validation prep
  # INSERT CODE HERE

  # Training models
  # INSERT CODE HERE

  # Predicting models
  # INSERT CODE HERE

  # Evaluating models
  # INSERT CODE HERE

  # Binding data
  # INSERT CODE HERE
}

# Calculate overall mean AUC
# INSERT CODE HERE

# Visualize distribution of AUC by variable (optional)
# INSERT CODE HERE

# Calculate Proportion of time the "best" model is better than the "worst" (optional)
# INSERT CODE HERE
```

Write answer here

Question 4 [2 points]

Now let's run a kitchen sink model using a random forest (make sure to install and require the `ranger` package). Use the following \(X\) variables: - `hits` - `assists` - `accuracy` - `head_shots` - `damage_to_players` - `eliminations` - `revives` - `distance_traveled` - `materials_gathered` - `mental_state` - `startTime` - `gameIdSession`

Run it on the full data and use `importance = 'permutation'` to see which variables the random forest thinks are most important. Visualize these results with a barplot. Where do the variables you thought would be best and worst appear?

```
# Require ranger
# INSERT CODE HERE

# Run RF model with permutation-based importance calculation
model_rf  <- ranger(..., # Insert regression equation here
                    ..., # Insert data here
                    ...) # Set importance calculation here
```

```
## Error in ranger(..., ..., ...): could not find function "ranger"
```

```
# Visualize variable importance results
# First, create a toplot object
toplot <- data.frame(vimp = ..., # Get variable importance values from model_rf
                     vars = names(...)) # Get variable importance value names from model
_rf
```

```
## Error in eval(expr, envir, enclos): '...' used in an incorrect context
```

```
# Second, visualize the results (make sure to reorder the variables in order of importan
ce)
# INSERT CODE HERE
```

> Write response here.

Extra Credit [2 Points]

What is the overall AUC from your random forest model? Then use 100 cross validation with a 60-40 split to get a better measure of the true model performance. Is the AUC from running the model on the full data different from the cross validation answer? Calculate the proportion of cross validation splits where the AUC is worse than the value calculated on the full data. Do you think there is evidence of overfitting?

```
# INSERT CODE HERE
```

> Write response here.