Amaan Rai 5-2-2023
STAT 380 Professor Slifko

Final Project Report

Front Matter:

I started the project by reading all the .csv documents and giving each .csv document a reasonable name. For example, I named the "CODGameModes.csv" document, "GameModes". Also, as I went through the project and had to install/load packages, placed them all in the *Front Matter* section.

Task 1:

I plan on answering this question by finding the count of each map winning subtracted by the number of times a map won for the sole fact it was option one and dividing that entire number by the count of each map being an option. I have to do this for both players and combine the numbers and then I will plot the data to see which maps had the highest win rate. However, I have to clean the data first by fixing the misspellings and removing cases with missing variables because those factors will alter my numbers. Lines 34-37 remove any blank cases or cases with NA values for the variable *Map1* for both player datasets. I did this because if I didn't, the blanks and the NAs would be included in my graphs. I didn't have to use the same code for *Map2* and *Choice* because after scanning the data, those variables didn't have any blank or NA values. In lines 39-108, I fix all the misspellings in the dataset. I don't individually change the cases, but I used code to check for all the different values in the *Map1*, *Map2*, and *Choice* columns and fixed the spellings for any case that had a misspelling. Next, in lines 113-116, I created two new columns called *LeftNum* and *RightNum* from the *MapVote* column. I split the numbers into both sides of the "to" and placed them into their respective variables. I also made them numeric. In lines 117-121, I created a new variable called *WinLoss* where if the number in *LeftNum* is larger than the number in *RightNum*, then Player1 won the game, if not, then Player1 either lost or drew. I also removed all the NA values. In lines 122-141, I first remove any draws. Then I create a new data frame called *WinCount* and in this data frame, I use the Player1 dataset and group the values by *Choice*, and find the count for each value. This tells me how many times each map was selected in choice in the *Player1* dataset. I also created a

dataset called *PlayerMap1OptionCount* based on the *Player1* dataset which groups the values of the *Map1* variable, finds the count of each value in the dataset, and renames the variable *Map1* to *Map*. I did the same thing for the *Map2* variable in the same dataset. I lastly created another data frame called *TotalOptionCount* which combines *the Player1Map1OptionCount* data frame and the *Player1Map2OptionCount* data frame using a full join. I created a new variable in the *TotalOptionCount* data frame called *TotalOptionCountVar* which adds both the counts from the two combined data sets to get the number of all the times each map was an option for player1. I also removed all the NAs from the *TotalOptionCount* data frame. I did the same thing for the second player's data set in lines 147-176. Next in lines 180-188 I do the same thing as I previously just did with the total options, except I did this to combine the player 1 data set with the player 2 data set for both, wins and options. These two new data sets are called *CombinedWins* and *CombinedOptionCount*. In lines 191-204, there was a problem with some map values not combining because the map wouldn't be in both players' datasets (Player2 didn't have some datasets as an option, let alone a choice). So I used a for loop and an if statement to get the numbers from the player1 dataset if there are any NA values because that dataset has all the maps. The code above iterates through to check for the NA maps and replaces the NA values with the count in the player 1 data set for both wins and options. Next in lines 209-213, I merged the two datasets, *CombinedWins* and *CombinedOptionsCount*, to create a new data frame called *merged_df*. In *merged_df*, I created a new variable called *WinPercentage* and divided *CombinedWinsVar* by *CombinedOptionsCountVar*, and multiplied the number by 100 to get a percentage. Then in lines 218-222, I used ggplot to plot a scatter plot of *WinPercentage* against *Map*. This tells me which maps have the most likely chance and the least likely chance of being selected. Essentially, the top five maps most likely to be selected are "Raid", "Crossroads Strike", "Diesel", "Nuketown '84", and "Express" in that order.

Task 2:

I first start by cleaning the data in the *GameType* variable so that there were only four different types of game modes and I did this for both the *Player1* and *Player2* datasets. I also removed any NA values in the *Player2 GameType* variable. All this was done in lines 229-251. Then I created

a new data frame called *P1_P2* which used a full_join to join both the *Player1* data frame and *Player2* data frame in lines 256 and 257. In lines 262 to 277, I created two plots; the first plot is a scatterplot that plots the variables *TotalXP* against *Score* for each value of *GameType*. This shows me that, on average, "Domination" looks like the best value game type while "Kill Confirmed" is the worst value game type, but there is very little data to make a confident assumption. The second plot I made is a box plot that plots *GameType* against *TotalXP*. This plot also shows that "Domination" is the best value game type and "Kill Confirmed" is the worst value game type, but again, we don't have a lot of data for both game types. However, for both plots, we can see that "Hard Point" and "TDM" have the greatest maxes, so players could potentially get their best value games from these two game types. In line 280 I got the five-number summary for each game type and in lines 281 and 282 I created a multiple linear regression model for *TotalXP* based on *GameType* and *Score*.

Task 3:

Research Question: Can we predict the if a player wins a game in an online match based on the player's performance, the map they are playing on, and the game mode?

The three classification methods I will be using are Random Forest, Logistic Regression, and Naive Bayes. I start by implementing the Random Forest classification method. I first create a new data set called *CODData* and then I clean and wrangle the data to make it usable for Random Forest. Lines 307 - 315 are where I implement Random Forest. Based on the predicted response, the classifier is roughly 65% accurate. Next, I implemented Logistic Regression in lines 302-326. After obtaining the summary of the model in line 327, I found that the adj. The r-Squared value is very low; making the Logistic Regression a poor classifier. Lastly, in lines 332-337, I implemented the Naive Bayes classification method. Based on the summary of the predictor, the accuracy of the classifier is about 63%. After implementing all the classifiers, Random Forest was a slightly better predictor than Naive Bayes by 2%. Logistic Regression was the most inaccurate of the three. I was also able to answer my initial research question, we are able to predict, with some degree of confidence, if a player wins a game in an online match based on the player's performance, the map they are playing on, and the game mode.