Jonathan Langley, Yong Liu, Sujeet Yeramareddy
Professor Fraenkel
DSC 180B
4 February 2022

Project Checkpoint Report

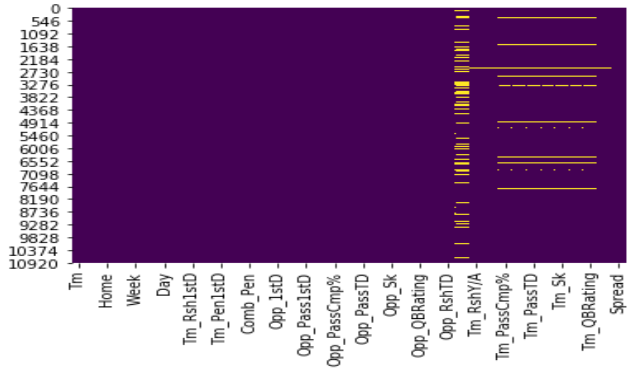
For our 180B project, my group decided to use our domain methodology of postpi (post prediction inference) and apply it to sports analysis based on NFL games. We are designing a model that can predict the outcome of a football game, such as which team will win and what the margin of their victory will be, and then correcting the statistical inference for selected key features. The main goals of our investigation is discerning which features most strongly determine the victor of a football game, and subsequently which features provide the most significant means of inferring the margin of that victory. For example, does the home field advantage give a 50% higher chance of winning by 7 points? Is the comparative offense to defense rating the most critical factor in securing a win? Does or does not weather play a statistically significant part in influencing margin of victory? These are just some of the questions we have brought up and seek to answer during the course of our research, and by conducting this project we are attempting to revolutionize the way NFL analytics are conducted via a more accurate statistical method of inference, postpi.

To begin a data science project, one requires data. Surprisingly so, the process of data collection was quite a bit more grueling than we had hoped. NFL data sets themselves and websites dedicated to NFL data are abundant yet almost none of them were helpful. We began our data search by hoping we would easily find one or two websites with all of the information we wanted neatly wrapped in uniform csv files just waiting to be merged together, much to our disappointment this was not the case at all. One website might have all of the offensive ratings for NFL teams from 2000-2021 yet lack any of the defensive ratings (which are just as important in our model), and another website might have the complimentary defensive ratings but only from 1993-2010, and also set up in a completely unique format so as to make the prospect of data merging nightmarish at best. We had hoped that maybe such a scenario would only be true for a couple of the features we wanted, but as it were to get the minimum features we needed with sufficient data points to allow for any reasonable machine learning model to be accurate we were looking at having to merge 10+ websites' csv files all formatted uniquely and encompassing different year ranges. Our stroke of luck came when Sujeet discovered the website https://www.pro-football-reference.com/?__hstc=213859787.31604216d3485faf4f8fc24da71f1b

88.1643944231760.1643944231760.1643944231760.1&__hssc=213859787.2.1643944231760&__hsfp=12124649

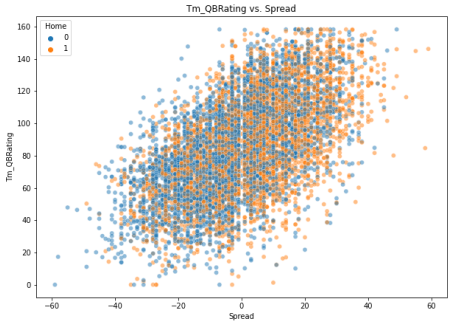
Which contains all of the features we needed, for all of the years that we needed, and all formatted in a very similar way. The catch was that we had to pay $8 a month each to get access to the csv files for the target features, and that we only had access to 100 rows of data at a time per feature (you had to click next at the bottom of the page to go on to rows 200-299, etc), and considering that there were 11,264 NFL games played from 2000 to 2021, and further considering we had 10 target features, this took quite a bit of tedious manual labor.

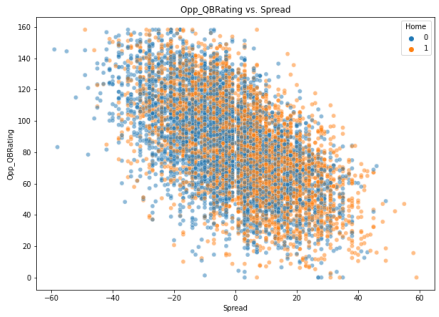
After we divvied up which of us would tackle which features and we pushed our combined data to the Github repository, we set out on our next step of exploratory data analysis. Merging the datasets together wasn't as challenging as it could have been simply because all of our data came from the same place, and all of the features shared at least one (but usually more) columns. The combined dataset consisted of 64 total columns ranging from the year the game took place in, to the number of passes completed in the game, to percent of passes completed in the game. Many of the columns were redundant, like having the number of passes attempted, the number of passes completed, and then the percent of passes completed (making the previous two unnecessary). In several instances columns were included for the winning teams but not for the opponents (an analysis on the importance of rushing yards must include how many yards both the winner *and* loser ran for). Additionally, several columns were present that were objectively not helpful and could be removed to help declutter the dataframe, like the "year" column while also having the "date" column, which included year. With a refined dataframe in hand we moved on to dealing with null values. A heatmap of null values allowed us to visually identify that 9 columns had 100-301 null values, with a tenth (temperature) having almost 2500 missing values.



As 300 data points out of 11,000 is highly insignificant, we opted to drop the null values for the 9 columns. For temperature, we imputed the 2500 null values based on the mean of the previous 7 temperatures (or from the following 7 temperatures for the first 7 values). With our completed dataset with null values removed we had a final count of 11,164 data points.

We generated a blend of scatterplots based on the remaining features in order to do an initial visual evaluation of how the covariates relate to game spread. Most variables don't have a linear relationship with the spread, which is why we aim to develop a neural network that can capture nonlinearities in the data. However, the QB Rating is one variable where a positive linear relationship can be determined with Spread, as shown in the scatterplots here. We can see that the better a QB of a specific team performs, the spread tends to favor that team. Observing this, we performed a simple linear regression to predict the Spread based on how each teams' QB performed, while including the home/away variable to normalize the comparison. This simple model suggests that with a standard deviation increase in QB performance, it can translate to approximately 8 game points, while explaining approximately 61% of the variance in spread.





My team's next step going forward will be implementing the rest of postpi. This will entail building a (potentially MLP) neural network prediction model with weights based on the varying influence of the features we have found to be most strongly correlated to game spread, making a linear/logistic regression model to observe the relationship between outcomes predicted on the test set versus the test set's actual outcomes, and conduct bootstrap based correction to find a corrected beta coefficient to use in our inference model (also likely linear/logistic regression).

Through testing ,we finalized our N-N (MLPregressor) model with 4 hidden layer and with the size of (32,64,64,128); The maximum epochs(how many times each data point will be use) of the model setting is 200; The validation set is 0.2 fraction of training data. According to the training loss curve , the training seems to stop around 80 epochs.

To test the robustness of our MLP model , we try different initial value of weight and bias by changing the parameter of Random_state in scikit learn MLP regressor:

Initialization	2	12	46	78	123	290	453	544	999	9999
Training loss(MSE)	16.643	16.475	16.770	16.389	16.275	16.181	16.292	16.483	16.427	16.626
Test MAE	4.421	4.375	4.398	4.354	4.388	4.373	4.396	4.369	4.403	4.389
prediction for 2022 super bowl (real-spread:3)	2.358	4.886	9.394	1.952	6.945	4.464	3.007	3.218	3.170	4.975

After trying out different initial values of weight and bias we can conclude that our model is robust and since the training error and test error did not vary too much. In the real world , our averaged prediction among these 10 different initializations is 4.437 which is very close to the real value 3. It is also surprising that the prediction on real games is extremely accurate when we choose initialization from a range(400-1000).