**DATA WORK:**

The first step I took was checking the summary statistics for all the 2020 and 2019 IV’s. I noticed that quite a few of the columns seem to contain position specific data (passyds, passint, etc.) and I made note of these to either remove as to reduce possible noise or to implement into a PCA. More specifically, I want to run a PCA on passyds20, passtd20, passint20, passyds19, passtd19, passint19 to represent QB stats and a PCA on rushyds20, rushtd20, rush1st20 (and their 2019 counterparts). Additionally, I tried my best to separate some of the remaining columns into defensive stats and offensive stats in which I could do the same process of either running a PCA or removing some IVs. Additionally, I really wanted to simplify my model because of the sheer amount of columns and I decided to get rid of salary (too many missing values and doesn’t seem very indicative because I presume it is based off of position), owners/pick(I don’t believe that picking a player correlates to how well they perform without adding unnecessary noise), projected21 (I don’t believe that ‘experts’ can outperform a model, even though they may have some ‘esoteric’ knowledge about players, I consider this noise in terms of prediction) as well as all 2021 data.

My next step was to remove all NA and see if this had any effect of the statistics for the columns I wanted to keep. So, I displayed the results of before and after dropping NA from the Mean and STF in each of the columns and it seems like they were roughly the same so I argue that dropping NA does not impact the patterns within the data; thus, I will not be imputing. Additionally, it seems odd to impute on many of these columns because I would imagine player data is highly varied and compared to another the previous assignment, for example, imputing on something like the climate of a country is more reasonable since such data could be derived from geographically similar countries; on the other hand, players could have many more underlying factors contributing to performance. The following graphs display my results. A graph of numbers and columns

Description automatically generatedA graph of numbers and columns

Description automatically generated

**FIRST REGRESSION:**

Now that I have established that I will be dropping NA across all rows, I want to run a PCA on the previously mentioned variables. First, I’d like to develop some methodology to deal with the fact that each different football position differs on the number of points they score and how they reach those points (position specific IV’s). My 2 main thoughts were to either create a dummy variable for each position (one hot encoding) and see where that gets me or to create a separate model for each position. I decided to start with creating dummy variables for each position and running the results on a lasso regression and decision tree. I created a new csv dataset which contains each of the dummy variables that only contains rows which remained after dropping NA; this is the dataset that I use for the following regressions. Keeping in mind that I get rid of the IV previously mentioned, I also decided to throw away any of the variables I determined could be run in a PCA except for (passyds20) since I wanted to get rid of as much noise as possible while maintaining the pattern found in these columns. In the future, I will go back and run a PCA before my regression to see if that makes any difference. However, after running my regression with the position binary variables, it seems my MSE has actually increased by 2,000 which I did not anticipate. The lasso regression with k-fold ended up sending all variables except QB (binary var for quarterback position), points19, points20, and fumbles20. Although I imagine that these variables are in fact good predictors since it makes sense that points in previous years correlate to points in a future year, my MSE is still in the thousands, so I have to revisit the issue of positions.

A screenshot of a computer program

Description automatically generated

**MORE DATA WORK:**

I realized that although dropping NA gave me similar data statistics, I only have around 90 rows of data left with approx. 30 IV’s which is a huge issue; I cannot expect to predict accurately with such few rows of data and so many IV’s. So, I now am looking to create a new dataset that drops na on certain rows that contain na on columns that I believe should not be imputed on and impute using KNN for the rest of the rows. This method is a blend between imputing and dropping NA and I will check the dataset after all processing to see if it still has the necessary summary statistics in line (mean, std). Also, I felt like I was initially rather overzealous in dropping IV’s for the sake of reducing noise so retained a few variables that I initially counted out from my first dataset (namely projected21); the point of leaving them in this dataset is to first see if they contribute to the model before I leave them out.

Drop altogether: pick21, salary21, gp20, owners21, returntd20, returnyd20, pt20, pick20, salary20, gp19, owners20, returntd19, returnyd19, pick19, salary19, name, team, actual19, actual20

Explanation: I dropped gp20 and gp19 because I assume players of similar performance played roughly the same number of games (performance matters because I am using KNN which will group players with their neighbors to impute). I dropped returntd and returnyd because they have extremely high variance and I figured it would do more harm than good to my model. Also, I dropped actual20 and actual19 because they are simply functions of points20 and point19

(I already explained my reasoning for dropping pick, owner, salary)

Impute: project21, (passyds20, passtd20, passint20), rushyds20, rushtd20, rush1st20, recepts20, recyds20, rectd20, rec1st20, fumble20, project20, passyds19, passtd19, passint19, rushyds19, rushtd19, rush1st19, recepts19, recyds19, rectd19, rec1st19, fumble19

Explanation: Going against what my previous statement that player data should not be imputed on, I am now reasoning that since I am using KNN, my imputation technique will group players by performance, and hence position, while imputing and it is reasonable to assume that similar performing players in the same position will continue to perform similarly. Thus, I impute on all player data to try and get more rows in my data set than when I simply dropped na.

Dropna: points19, points20

Explanation: I decided against imputing on points19 and points20 because these vars are essentially what are being predicted (except in another year) and my imputed player data should be able to predict points rather than having to impute the number of points in previous years as well.

Explanation pt2: After dropping NA on these cols, I was only left with 160 rows and I felt like that was still a very small number, so, I ended up imputing on both of these vars for the sake of having more data; in the end, I did not use dropna at all.

**2ND REGRESSION/DECTREE:**

After rerunning my lasso regression, I managed to decrease my MSE by around 2,000 and increase the R2 to .46 which is a decent improvement. Unfortunately, my model sent most of the binary position IV’s to 0, which is something I will have to continue to work with. My decision tree also reflected similar results. Results in the following images: A screenshot of a computer

Description automatically generated

A diagram of a diagram

Description automatically generated

**PCA:**

At this point, I want to know if the reason my MSE is so high is because of the choices I made on dropping certain IV’s or the one-hot encoding I used on player positions. So, I want to run a PCA on 3 groups of variables to and then run the same model using the one-hot encoding to test whether there is a difference between the results from dropping IVs and my new PCA results. If my MSE is still incredibly high, I will assume that my main issue stems from player positions. I will run 3 separate PCA’s on (passyds20, passtd20, passint20, passyds19, passtd19, passint19) because they all are exclusive to the QB, (rushyds20, rushtd20, rush1st20), (recyds20, rectd20, rec1st20) (along with 2019 versions of each IV). The reason for a PCA on the latter IV’s is pretty self-explanatory, however, I was wondering how I should treat the fact that each IV has data for both 2019 and 2020 since I don’t necessarily expect a player to drastically improve within 1 year and its possible that variables from 2020 to 2019 are colinear. First, I ran the PCA on both years with the following results: (interestingly, variables from 2019 tended to account for the most variance across PCAs, maybe 2020 is skewed from COVID)

Pass PCA: A graph with numbers and symbols

Description automatically generated

A graph of a graph

Description automatically generated with medium confidence

Rush PCA: A graph with numbers and symbols

Description automatically generated

A graph of a bar graph

Description automatically generated with medium confidence

Rec PCA:

A graph with numbers and dots

Description automatically generated

A graph of a graph

Description automatically generated with medium confidence

(each PCA calculated 3 principle components and in all of them, PC1 typically accounted for around .9 of the variation for each PCA I ran but I didn’t graph a representation)

**3RD REGRESSION/DECTREE:**

Now that I have a 3 PCAs, I will drop all columns associated with each PCA in the dataset and replace them with each PCA; then run regression and dec tree. Results:

DECTREE:

A diagram of a tree

Description automatically generatedA screenshot of a computer

Description automatically generated

LASSO:

A screenshot of a computer

Description automatically generated

A graph with blue dots

Description automatically generated

It seems like pass\_latent was selected, which is good news, but my performance does not improve.

**CONCLUSIONS:**

Overall, I am confirming the null and that my efforts were not able to create a model that is accurately able to predict player performance in 2021. Aside from the fact that my dataset has relatively few rows compared to the numbers of columns, there are a number of ways this model could have went wrong considering the steps I took. With that being said, I think that my final lasso model (with kfold) is the closest since the lasso is harsher on feature selection than decision tree and therefore can produce more accurate results on datasets with a large number of IVs such as this one. This is because lasso easily sends IVs to zero; this technique tends to work well on datasets with many IVs. However, the decision tree is a greedy algorithm which makes decisions at each split; this aspect makes them much more reliant on training data and thus will have a harder time predicting at a large scale. (this could be fixed by an ensemble method).

Additionally, I had to make certain sacrifices after I imputed for the sake of keeping as many rows in the data set as I can. For instance, I did not necessarily want to impute on columns like points20 and points19 because doing so with KNN would assume I have some method to predict player points that did not actually play in those seasons, which is essentially what my model is trying to do in the first place. Also, when I created columns for player positions the columns were not used by the lasso or decision tree and this could have added more dimensions to the data with no benefit to the model.

Also, it makes sense that the PCA did not necessarily improve performance that much because the grouped/summarized variables might not be very correlated with each other (they seem to spread in the PCA plot). The year 2020 variables are somehow not well correlated with the 2019, with 2019 being better predictors as shown from the previous attempts.

Still, PCA did not decrease performance, and reduced the number of variables, which is an advantage.

To potentially fix some of these issues before I ran the models on the test data; I could have run a separate model for each position to possibly find a more tailored approach. Additionally, running an ensemble decision tree would have allowed me to get a good idea of the roof of possibilities regarding final performance of my models. Also, I was wondering if doing a multicollinearity analysis first and then selecting variables for PCA would work better,

despite the obvious grouping of them based on game action categories (passing, receiving, rushing).

['points20', 'points19', 'pass\_latent', 'TE']

As for the final variables picked by the Lasso, it makes sense that points20 and points19 were picked considering it is intuitive that previous points are good indicators of future points. In addition, pass\_latent also makes sense because it essentially indicates whether a player is a QB or not and makes the necessary predictions. Finally, TE also makes sense to be picked because the TE players have extremely high variance, and the model knows to differentiate among other positions and TE.