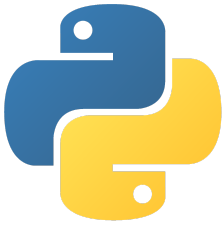
NFL Prediction Analysis

Analyzing NFL statistics from 2012-2021 for important features for wins and simulation of the 2022 NFL Season

CPT_S 315 Final Project



AbdulAziz Al-Dalaan

5/4/2023

Introduction **3**
Motivation 3
Questions 3
Challenges 4
Goals 4
Results 4
Data Mining Task **4**
Task Details 4
Input Data 5
Output Data 5
Technical Approach **7**
t-distributed stochastic neighbor embedding (t-SNE) 7
Random Forest Classifier 9
ELO Rating System 10
Evaluation Methodology **11**
Pro Football Reference (PFR) 11
NFL Data Py 12
Five Thirty Eight's NFL Predictions 12
Results and Discussion **13**
Relevant Figures 13
What worked and did not work 15
Lesson Learned **16**
Acknowledgements **16**

Introduction

This section looks to highlight the motivation behind the reason I have chosen this particular topic for my final project while also providing examples of data mining questions I personally set out to investigate along the challenges and results I have ended up with at the end of this project.

Motivation

The motivation which drove me to choose this particular project in terms of real-world applications which correlate this would be primarily sport prediction algorithms, more particularly those which are in the realm of the NFL. Many different sport blogs, and article sites will also have these “Power Rankings” which are primarily determined by real world statistics of the game itself. Furthermore this plays a huge part in the area of Sports Betting, Fantasy Football, Media, and more. On top of this one of the biggest statistics partners of the NFL is AWS, which provides the League with a set of advanced statistics known as Next Gen Stats which utilizes data such as Player, Route, Passing, and Tackle tracking capabilities. These are just some of the real-world applications which made me want to dive into the area of NFL matchup predictions. As for my personal motivation, I enjoy watching the game of football and have been watching it for around 5 years now and have grown more passionate about it more as the years go by and wanted to further my understanding of the sport more by taking on the challenges for this project.

Questions

The main question which I wanted to look for when working on this project was determining really which particular notabile statistics led a team the most to win a game. One could always say that the team that scores the most points really wins, but is that truly the case? Similarly there is a saying amongst the league called “Defense Wins Championships” and this is true. In the 2013 NFL Season the Seattle Seahawks won their first SuperBowl with one of the greatest defensive performances by a team within a season of all time against one of the greatest offensive performing teams of all time the 2013 Denver Broncos a staggering 43 to 8. Hence in a sense it is important to determine which particular statistics/features possibly matter the most to win games. Another question I hoped to answer with this project is that with these features highlighted is it possible to adjust a simple ELO rating algorithm, such as FiveThirtyEight’s in order to achieve more accurate results? These are the two main questions I hoped to answer with this project, however with these questions comes many challenges in order to solve them.

Challenges

Challenges when working through this would be the fact that there are already some great prediction algorithms out there such as FiveThirtyEight's Prediction which heavily accounts to almost every factor in the game of football with a high focus on QBR, QuarterBack Rating. Another significant challenge is that the game of football constantly changes in that new rules are made every season, players get injured, and new ones which are unpredictable in terms of quality come in every year which could lead to important features/statistics to change with every season. So to say that the prediction could be fairly accurate is a challenge which is easier said than done.

Goals

Goals I had for this project were primarily to find data mining methods in order to determine a set of highlighted features which lead a team in the NFL to win the most while also creating my own version of the ELO rating system, method for calculating the relative skill levels of players/teams in zero-sum games, with a modification based on the highlighted features which were selected from the previous data mining task.

Results

The results I have gotten from my project I believe were satisfactory in determining both which statistics/features in the NFL lead a team to the most wins as well as using these selected features to modify the ELO rating algorithm to simulate the outcome of the 2022 NFL season.

Data Mining Task

Now I will look to describe the details of the tasks which were completed within this project (i.e., input data, output data, and illustrations). I will also list more in depth into the questions and challenges I personally set out for and overcame within this project.

Task Details

To briefly describe the details of the two tasks I wish to accomplish, each of which correlate to one of the two questions previously stated, were first to find the top most features which lead an NFL team to win games among a set of statistics the most. I have used TSNE in order to analyze the data in correlation to the number of wins by each team among different datasets along with using a Random Forest classifier in order to find the top most features which lead the teams among the different datasets to wins. These features which are found will then be used (if possible), in my own personal version of the ELO rating algorithm in order to run a simulation of the 2022 NFL Season. These are the two main tasks with this project, now we will move on to going in depth into the input data.

Input Data

The input data for my entire project is a total of four csv files, the first titled “decade_team_stats” is a summary of statistics from every year from 2012-2021 which include the statistics I look to analyze and highlight which ones are of the utmost importance to win games. The next “nfl_elo " is similar to the first csv in that it covers dates from 2012-2021 however this one keeps track of ELO data of teams and QBs for every single matchup within that time frame. This same file also gives me the updated elo post matchup for every team and QB. The next is “season_data”, this file has data on all matchups and outcomes within the 2022 NFL season (the season the program aims to predict). The final file is “sos_data” and contains a team name along with their respective “Strength of Schedule” value for the 2022 Season which is a metric used to measure the difficulty level of the team which they are facing that season. Now with this data we will now take a peek at the output.

Output Data

The output data is split into both figures (which by default are not shown), as well as two data files. The figures come in 2 types across 3 different datasets, the first type is the simple scatterplot between the number of wins and a teams rank in there are a total of 9 different stats hence, 9 miniplots. The second type of figures is a TSNE scatterplot which is used to highlight particularly which stats are of the utmost importance to win out of all the statistics. The 3 datasets in question are teams which finished with a neutral record or higher, teams which managed to get to the NFL playoffs, and teams which made it to the SuperBowl between 2012-2021. Now I will provide a sample of each graph type.

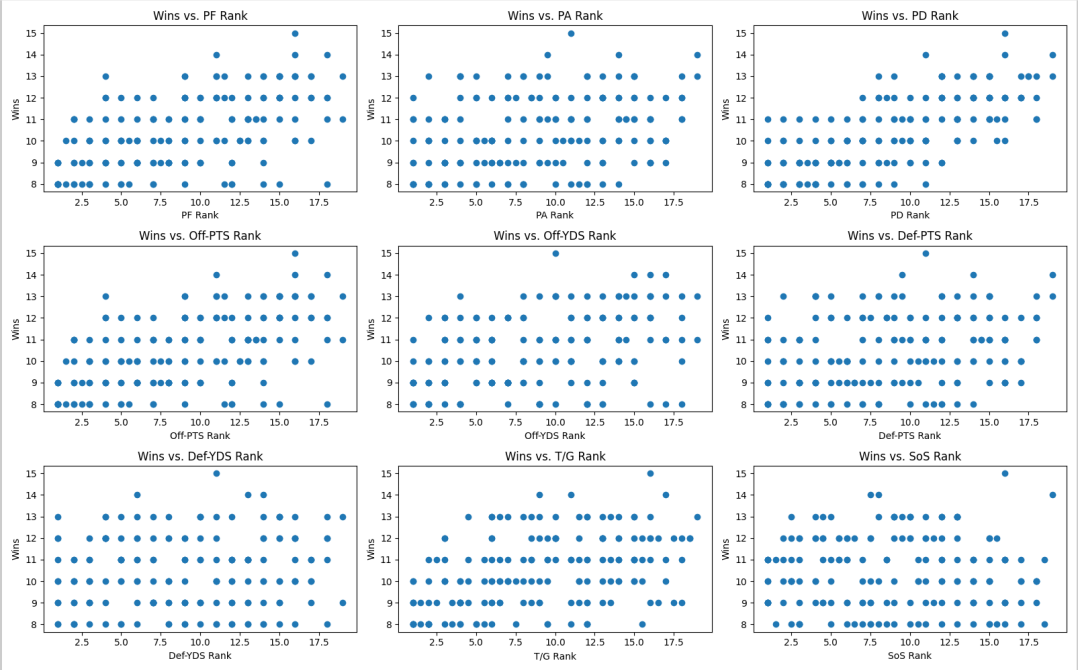


Figure 1. *Wins vs. NFL Stat Rank subplots on Neutral Record teams from 2012-2021*

This first figure is of the dataset of neutral records or higher, split into 9 mini scatterplots correlating wins to a team rank in a particular stat. The higher the datapoint is in both the X and Y axis the more important that stat is for the team winning or in other words the more points there are in the top right the more into that feature is.

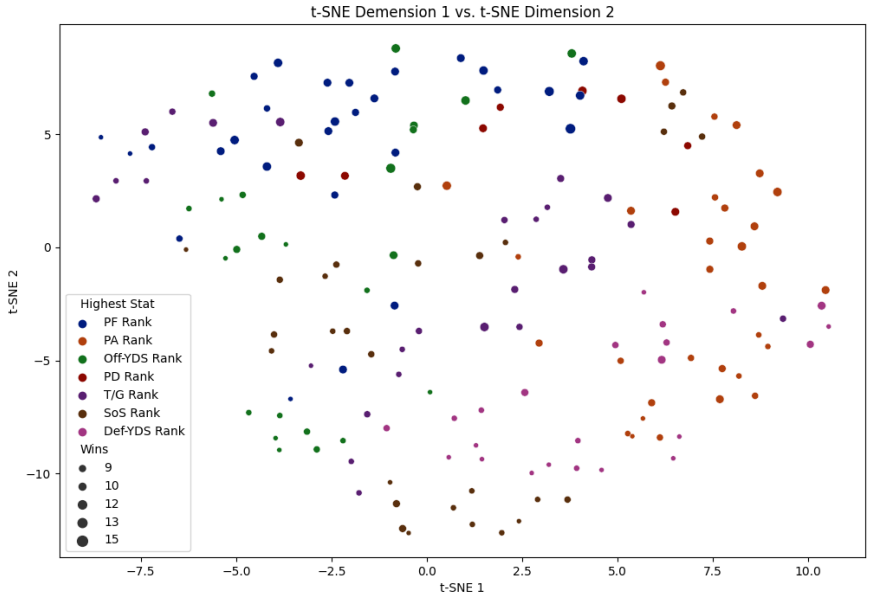


Figure 2. *t-SNE Plots along NFL Stats Ranks on Neutral Record Teams from 2012-2021*

This next figure which is still over the same dataset is of the TSNE category which utilizes all the statistics found in order to determine a depiction of where each of the features lie in terms of the number of wins. For example in the graph above we can tell that PF Rank has a high factor in terms of obtaining the utmost number of wins along with Off-YDS Rank and PD Rank. Though this graph does not show a perfect cluster of all these particular stats it still shows viable information on how they can be interpreted.

The next bit of the output comes from two text files, titled features.txt and season_results.txt. The first file, features.txt will display the important statistics/features from all three datasets which lead the team to win the most. Which will look like the following:

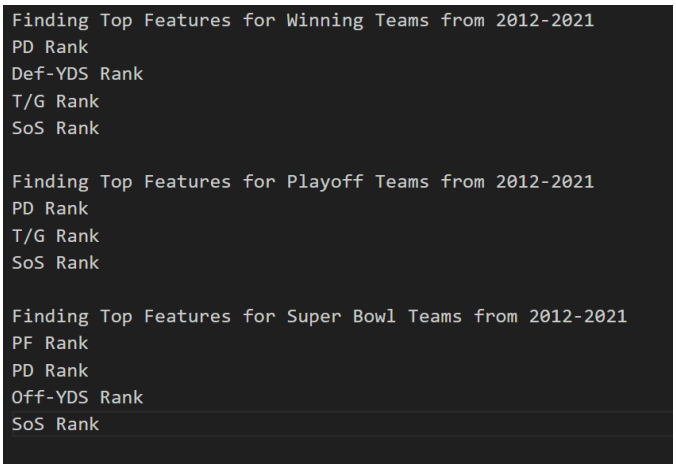


Figure 3. *Features.txt output Example*

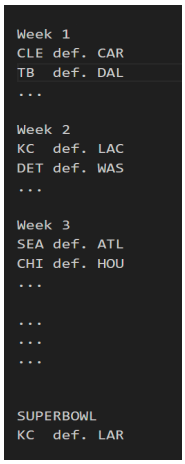


Figure 4. *Season_results.txt output example*

In the left image is the features.txt file, note that these features could differ as the classifier (Random Forest) which was used to pick out the highlight features above operates on random states which could vary through each run of the program. The final bit of output data which is on the right image is the season_reasults.txt. This file will simply show the output of each matchup throughout the 2022 NFL season from the start of Week 1 all the way to the Super Bowl through values determined via the ELO rating Algorithm. Now we will move on to discuss more in-depth on the necessary algorithms in order to obtain these outputs shown in the 4 previous figures.

Technical Approach

When incorporating the algorithms mentioned within this section, I tried to address the challenges I stated within the Introduction as much as possible. With the factor that NFL rules change with every season, player injuries, and the fact that new players come in every year, I wanted to make sure I had a decent amount of data from respective games which were not too far back as to be affected by new rules, but were also a decent amount in general in order to reflect the recentness of the 2022 Season, hence this is why I went with the data from 2012-2021 Seasons respectively. This also benefits me in that I have access to the most recent Elo ratings or teams for the respective season which will help in predicting the 2022 Season in general. Now in terms of how my predictions can be in line with for example FiveThirtyEight's it will depend on the factors I put into place when updating the elo ratings which I believe are fair in accounting for the highlighted features obtained via the output in Figure 3.. Now I will be going in-depth into each one of the three main algorithms which I utilized within this project, two of which originated from the scikit-learn library and one which I developed manually.

t-distributed stochastic neighbor embedding (t-SNE)

t-SNE is a machine learning method which is used to visualize high-dimensional data within a 2-3 dimensional space which helps in terms of making data easier to understand. It does this by measuring the similarities between data points in high dimensions and looks to lower the dimension complexity in order to have similar data points stick closer together, and have unrelated ones far apart from one another.

This method was used to obtain the output in Figure 2 in order to find the particular stats/features which correlate to the most wins. The scikit-learn library t-SNE was utilized for this project via importing TSNE from sklearn.manifold. Now the pseudo-code for the algorithm was as follows, given a data frame with NFL team stats from 2012-2021 (Which vary in size depending on the dataset) called nfl_data_frame.

1. Calculate the perplexity value of the dataframe
   This is so that the figure can account for the number of rows within the varying dataset sizes as not adjusting this or setting it to a fixed value could tamper its effectiveness. Which we will call “perplexity_val”

2. Create a t-SNE model with 2 components and set the perplexity value
   tsne = TSNE with n_components=2, random_state=42, and perplexity=perplexity_val

3. Fit and transform the t-SNE model on the data with selected features called stats:
   tsne_results = tsne.fit_transform(nfl_data_frame[stats])

4. Add the t-SNE results as new columns within the NFL dataframe (i.e., the first and second components of the results):
   nfl_data_frame[‘t-SNE 1’] = tsne_results[:, 0]
   nfl_data_frame[‘t-SNE 2’] = tsne_results[:, 1]

5. Create a new column to store the highest ranked stat in each row:
   nfl_data_frame[‘Highest Stat’] = index of the maximum value in row of the dataframe

6. Create the t-SNE scatterplot using Seaborn, a python visualization library:
   We set the plot size to be 12 by 8.
   Then we use the following line to graph the plot
   sns.scatterplot(data=nfl_data_frame, x='t-SNE 1', y='t-SNE 2', hue='Highest Stat', palette='dark', size='Wins', legend="brief")

7. Return the plot:
   return plt

This is the algorithmic approach I took in order to find the t-SNE correlations between each stat and the number of wins respectively and we will now move on to the approach I did in order to obtain the top features from each dataset.

Random Forest Classifier

Random Forest Classifier is used in classification tasks in which an input needs to be assigned to several predefined categories and/or classes. This is done by combining multiple decision trees in order to reach a single result. For this case each decision on its own will make its own prediction of how to classify the data itself, then the Random Forest will combine all them together. The benefit of this over let's say a single decision tree is that a Random Forest classifier will have a much better accuracy when classifying the data as opposed to a single decision tree classifier. Now for this section we will be using the scikit-learn library's RandomForestClassifier from sklearn.ensemble as well as a SelectFromModel from sklearn.feature_selection which will be used to select the proper features from our Random Forest Classifier. Now the pseudo-code for the algorithm was as follows, given a data frame with NFL team stats from 2012-2021 (Which vary in size depending on the dataset) called nfl_data_frame.

1. Create feature and target Dataframes (Stats represents the list of statistics which we aim to highlight from the data and 'Wins' represents the column with the number of Wins within the dataframe)
   X = nfl_data_frame[stats]
   Y = nfl_data_frame['Wins']

2. Create a Random Forest Classifier model (n_estimators, number of trees in the forest, and random state, fixed random variable for consistent output)
   clf = RandomForestClassifer with n_estimators = 100 and random state = 42

3. Fit the Random Forest Classifier with our set X and Y values
   clf.fit(X,Y)

4. Initialize a SelectFromModel instance to filter relevant features from the classifier (prefit is set to true because we already fitted the data to the classifier):
   selector = SelectFromModel(clf, prefit=True)

5. Obtain the names of the filtered columns from the selected features from selector.get_support() and output these features to features.txt.

This is the algorithmic approach I took in order to obtain the necessary features among the three datasets which lead NFL teams to the most wins which are then outputted to the file as shown in figure 3. Now we will move onto the algorithm which I implemented a version of myself, the ELO Rating System.

ELO Rating System

This method is used to calculate the relative skill between different levels of players and/or teams in order to predict the outcome of a match between them. In this system, every player is given an initial rating called a elo rating and based on the outcome of matches with others, this rating goes up if they win and down if they lose. Another factor that this system takes into consideration is that if a player beats another one with a higher rating their own rating goes up by large amount however if a player with a high rating beats one with a lower rating, their points only goes up by a small amount. There are also other factors which are taken into consideration when applying this algorithm across different sports such as football where we account for Home and Bye Week Advantages. Now within my project we not only account for a team's elo rating but also a QB's, because a QB accounts for a hefty percentage of a team's success. Now the pseudo-code for this within my project is as follows. For this modification of mine after I found the necessary highlight features, one which I found that was not taken into account was the Strength of Schedule value as typically teams which has a much tougher schedule tend to have a losing record and this in contrast to a team with a easier schedule hence an addition I made to the ELO rating system was the addition of a SOS_FACTOR variable to account for this as this was the main feature which is not typically accounted for when calculated the ELO for a NFL Season.

1. Define constants HOME_ADVANTAGE, BYE_ADVANTAGE, K_FACTOR, and SOS_FACTOR. All of these are taken into account when updating a team's and QB's elo.

2. Load the latest Team and QB ELO ratings prior to the 2022 Season from nfl_elo.csv into an elo dictionary containing the elo ratings of each team and their respective QB.

3. Initialize a dictionary (team_records) to store a team's win-loss record.

4. Create a function to calculate the expected outcome of the match based on the Elo ratings
   For my case, I called the function get_prob which takes in two Elo ratings from Team A and B, then calculates the odds of Team A beating Team B given their Elo ratings.

5. Define a function to update the Elo ratings for both teams and QBs after a matchup
   For my case, I called this function update_elo which takes in the Winning and Losing team's elo ratings and updates them respectively. These updated elos are then sent to the Elo dictionary to store. It is also important to note that each team is updated by a K-Factor amount along with the probability in which that team either wins or loses.

6. Now the next section of pseudo-code repeats for matchups within the regular season.

    i. Given a list of match ups (i.e., two teams)
    ii. Get the home and away team
    iii. Apply Home, Strength of Schedule, and Bye Week Advantages accordingly
    iv. Simulate the outcome of the match using the adjusted elo and calling get_prof and then set whichever them lost and won accordingly
    v. call update_elo with the winning and losing team and their elos
    vi. Update the Elo Dictionary
    vii. Update the Team Records
    viii. Print the match outcome to season_results.txt
7. The process of Step 6 is then repeated for the playoffs, and SuperBowl matchups

Now we have gone through the ELO rating system which I implemented and season_results.txt will contain the output of every match from the start of the Regular Season, Wildcard Round, Divisional Round, Conference Championship, and then SuperBowl. Other functions are used to calculate the matchups within the playoffs as they are determined via Division and Seed values by the NFL.

Now that we have walked through the primary algorithms which I utilized within my project I will now dive into the dataset itself and discuss challenges I ran into incorporating them as well as what I used in order to measure the accuracy of my data.

Evaluation Methodology

For the way I used to evaluate my results given from my datasets I primarily used the outcome of the 2022 Season itself, provided in season_data.csv in order to determine the accuracy of my results along with FiveThirtyEight's own predictions to determine the outcome of the season itself as the ELO rating accuracy is determined upon it though I had ran into many challenges when first incorporating the datasets. As there were some teams which changed names and/or locations throughout the course of the last decade which forced me to go and manually change them to match. Along with this the dataset I worked with came with a lot of unnecessary data which I had to filter initially through in order to figure out in particular which columns were needed and which were not. Now I will go through my three main primary sources I utilized for obtaining my data.

Pro Football Reference (PFR)

PFR is an online resource which is dedicated to provide a vast amount of statistical and historical records of not only the NFL, but also the AFL. It allows users to access both teams statistics as well as players statistics through a vast amount of perspectives such as entire history, season basis, and game to game basis. This is the resource I used to obtain my decade_team_stats.csv and sos_data.csv files. The decade_team_stats.csv contains the seasonal data of every NFL team from 2012-2021 and the sos_data.csv contains every team's strength of schedule value for the 2022 NFL season.

NFL Data Py

This is a Python package designed to easily retrieve the latest NFL data for any purpose. It provides users with a vast amount of functions in order to retrieve information regarding aspects such as play-by-play, season, schedule, draft data, and more. I used this library in order to obtain my season_data.csv information which contains information in regards to the 2022 NFL season and was used to obtain the original matchups in order to predict the 2022 Season using the ELO rating system.

Five Thirty Eight's NFL Predictions

This is an online data forecast of game outcomes and team performance specifically within the NFL Season. It uses the ELO rating system in order to determine these forecasts with a variety of variables such as team strength, home advantage, and QB rating. I used relevant ELO data provided by FiveThirtyEight in order to obtain the most recent Team and QB Elo ratings prior to the 2022 NFL Season which are stored in my nfl_elo.csv file. Five Thirty Eight's own predictions of the said season online is another metric which I used in order to determine the accuracy of my own results. Along with the actual outcome of the 2022 Season provided in season_data.csv.

Results and Discussion

Relevant Figures

Now I will display other relevant figures and data other than Figure 1-4. Which I have already explained the purpose of and then going into explain what I have discovered by obtain these graphs.

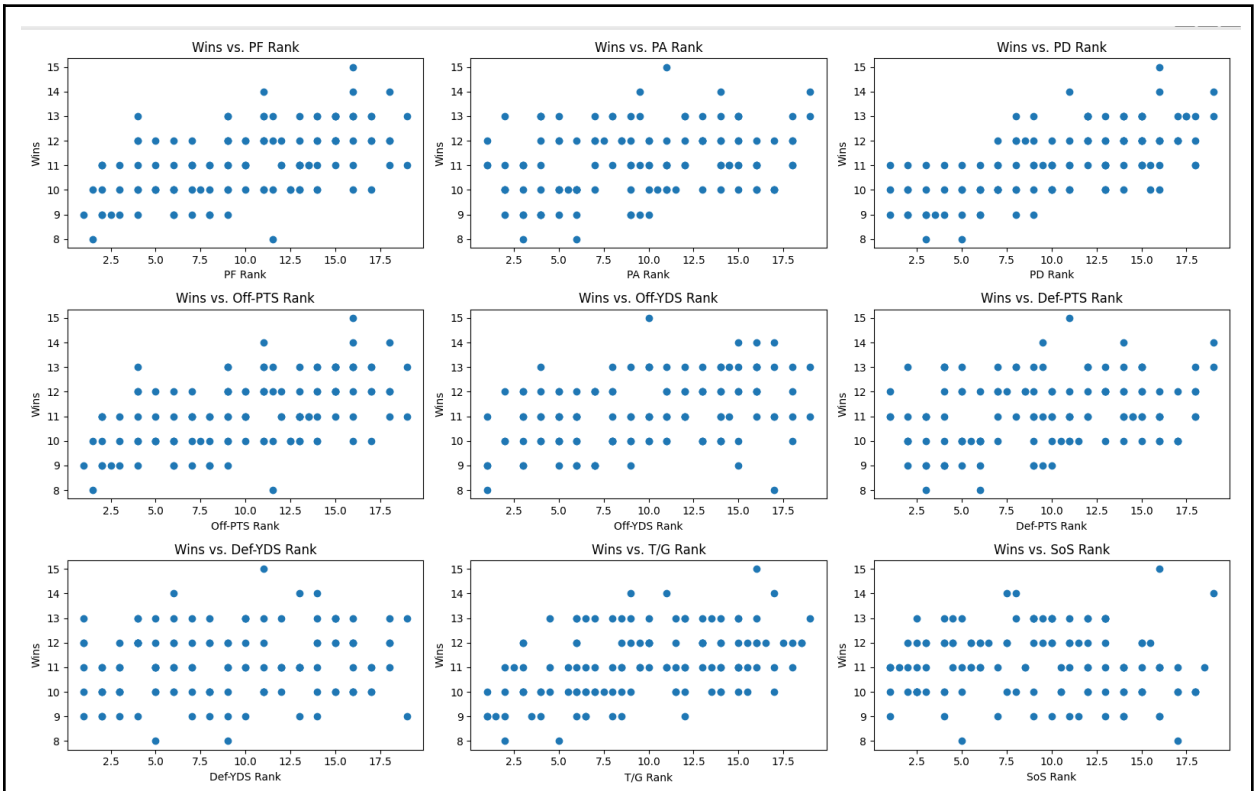


Figure 5. *Wins vs. NFL Stat Rank subplots on Playoff Teams from 2012-2021*

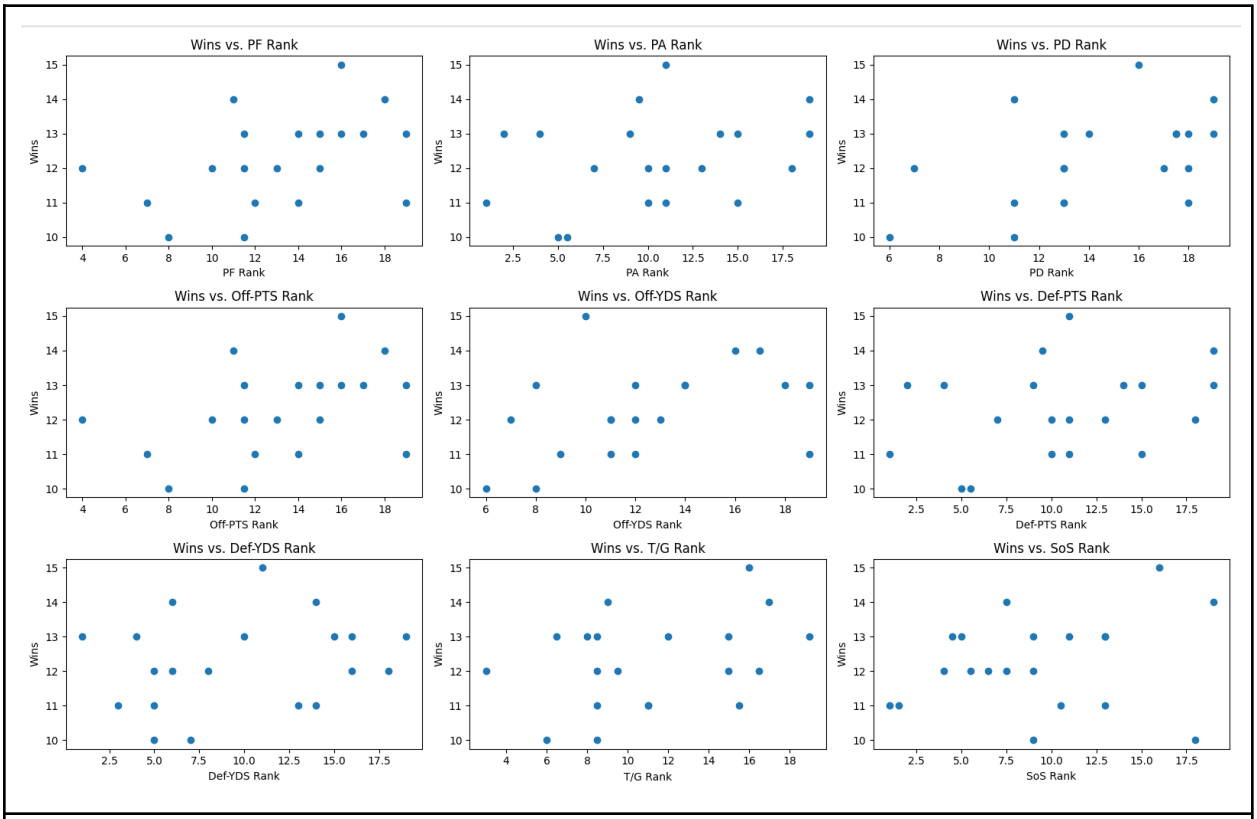


Figure 6. *Wins vs. NFL Stat Rank subplots on SuperBowl teams from 2012-2021*

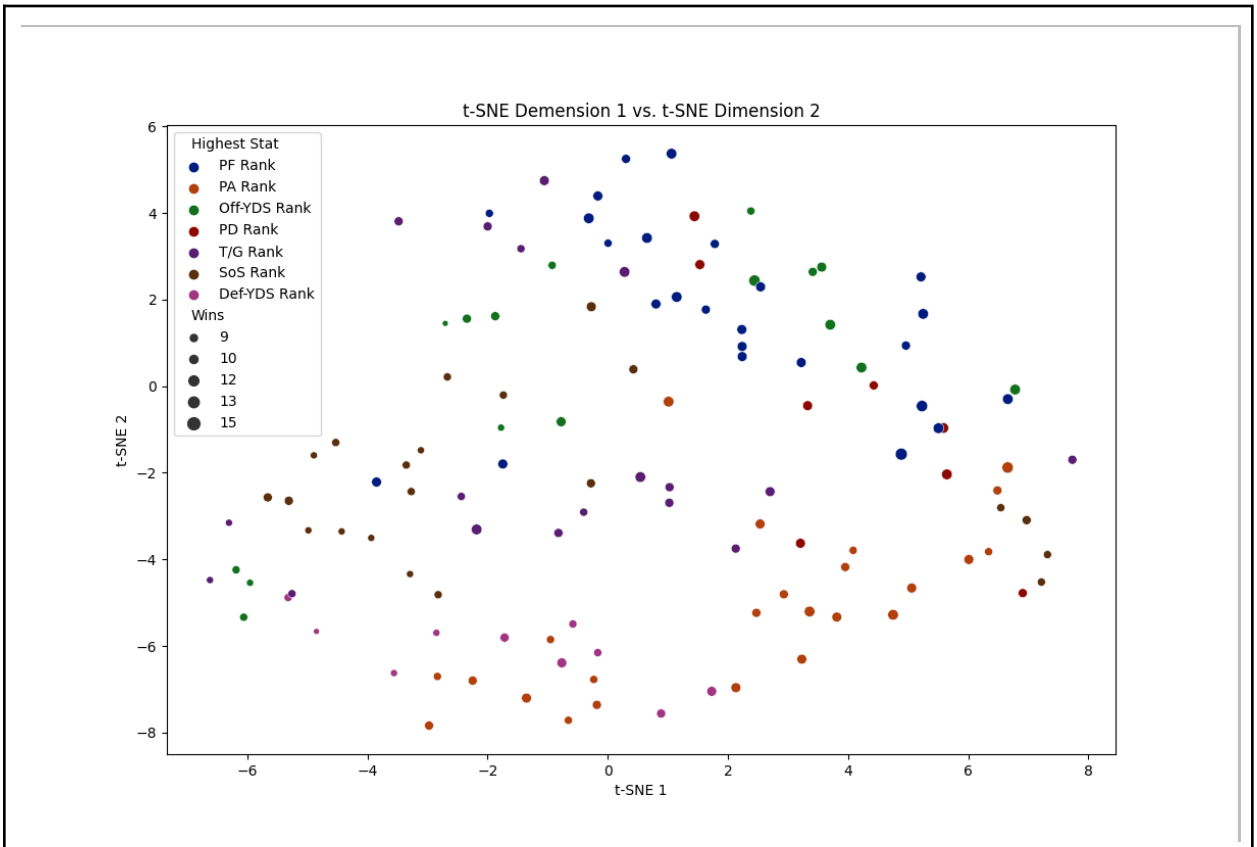


Figure 7. *t-SNE Plots along NFL Stats Ranks on Playoff Teams from 2012-2021*

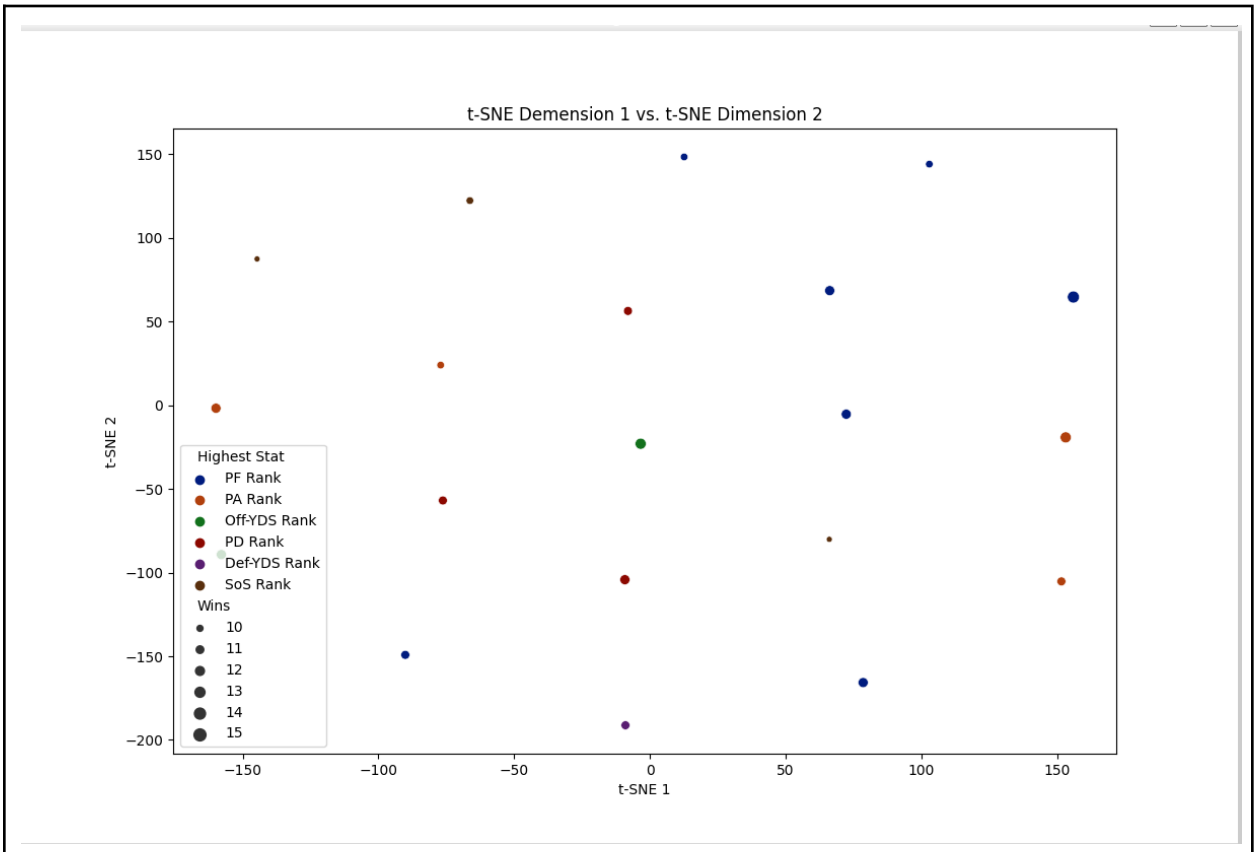


Figure 8. *t-SNE Plots along NFL Stats Ranks on SuperBowl Teams from 2012-2021*

What worked and did not work

Along with these figures are also the data provided in features.txt and season_results.txt which we have already elaborated on. These other figures are simply the same ones as shown and explained before just on different sized datasets. When first learning t-SNE I was skeptical of the fact of using it to represent simply statistical rank to win correlation data, however as shown in Figure 8. This method perhaps should not be used on a small dataset and I should have eliminated this graph entirely, as this graph in my opinion does not give any necessary information that could be simply explained with a linear model. For the data front the Random Forest Classifier I believe that this classifier was great in terms of highlighting the important features of wins in the NFL and is consistent for the most part through multiple iterations as shown in features.csv. As for the ELO rating system, this is a system determined by a vast amount of variables and randomness especially, with that under consideration I believe that my season_results.csv are consistent through different iterations of calculated the end winning team. In essence what worked is to obtain the features and the ELO rating system, because I had a firm understanding on the highlighted features obtained and how to incorporate them into the rating system but what did not was analyze these features using t-SNE as either I have misinterpreted the data in which the algorithm can take itself, perhaps the data is too simple, or this algorithm does not work well on small datasets.

Lesson Learned

From this project I have learned a couple of data mining and data analysis concepts as well as a couple of python libraries in correlation to a great passion of mine which I hope to get more into in the future. In hindsight if I were to improve this project even further, which I really hope to, I would take more time into understanding the concepts such as t-SNE itself as understanding these concepts are what took me the longest time to understand throughout the course of the entire project and perhaps I am still missing a couple pieces in order to fully grasp the concept. I would also love to improve my prediction algorithm to be even more accurate and detect and account for changes such as player injuries, new draft player ratings, and more which could help improve my own Elo rating system's accuracy to that of FiveThirtyEight's.

Acknowledgements

Note this acknowledgements section only contains sources in which datasets where retrieved as well as information in regards to the ELO rating system

Diff, C. (n.d.). *NFL-data-py*. PyPI. Retrieved May 4, 2023, from https://pypi.org/project/nfl-data-py/

List of all the pro football franchises. Pro-Football-Reference.com - Pro Football Statistics and History. (n.d.). Retrieved May 4, 2023, from https://www.pro-football-reference.com/teams/

Silver, N. (2018, September 5). *How our NFL predictions work*. FiveThirtyEight. Retrieved May 4, 2023, from https://fivethirtyeight.com/methodology/how-our-nfl-predictions-work/

Silver, N. (2023, February 13). *2022 NFL Predictions*. FiveThirtyEight. Retrieved May 4, 2023, from https://projects.fivethirtyeight.com/2022-nfl-predictions/