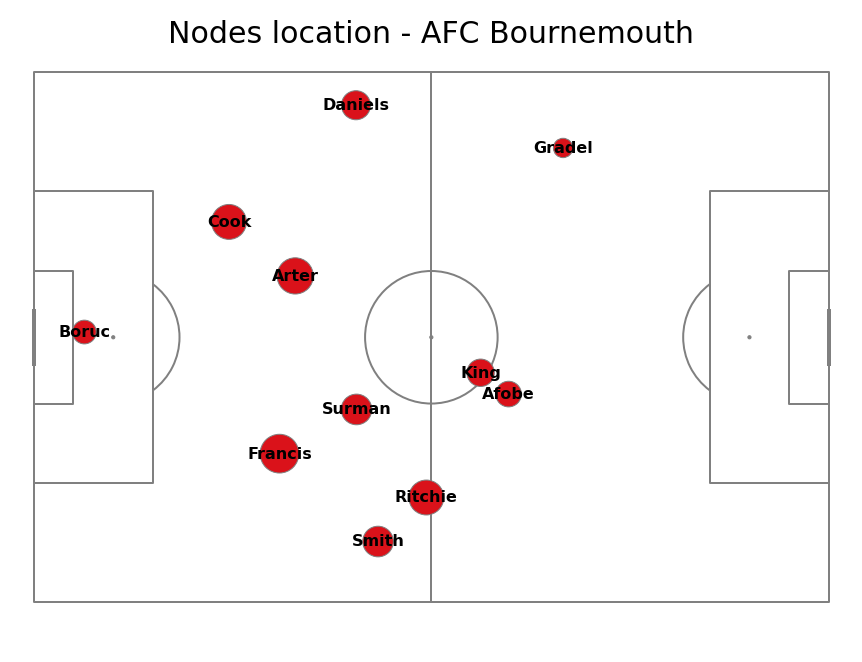
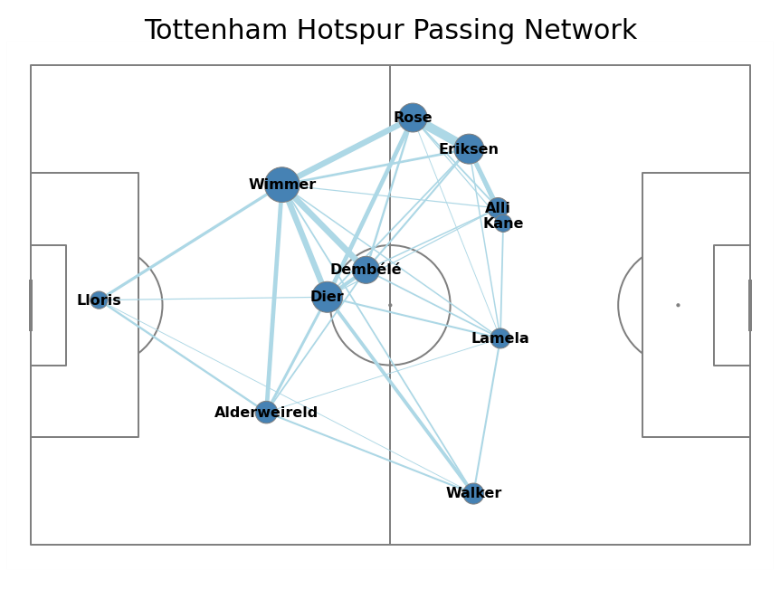
**Connecting the Dots: The Hidden Links Between Passing Networks and Soccer Match Outcomes**

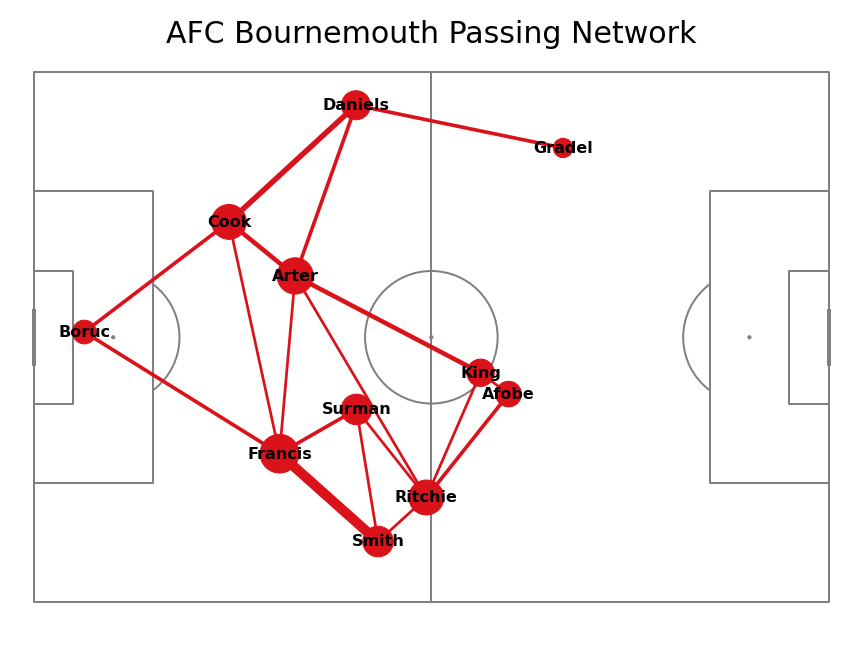
**Acknowledgements**

In the research of this paper, I was advised by Dr. Nicolas Fraiman. This research was performed as a part of the NSF Networks: Foundations in Probability, Optimization and Data Science Research Training Group with the UNC Statistics Department. I want to thank both Professor Fraiman as well as the UNC stats department on helping me complete this paper.

**Introduction**

The goal of this research primarily focused on classifying soccer match outcomes through each game’s respective passing networks. Firstly however, what is a passing network. A passing network is a directed weighted graph where each node symbolizes a player, and each edge symbolizes a pass between each player with the weights for each edge being made up of total pass counts. In each network, the node location is defined by the average X and average Y coordinate. For example, if player one receives the ball once at (1,2) and a second time at (1,4) then his average location would be (1,3). These networks can differ significantly from game to game. For example, look at the network below from AFC Bournemouth vs. Tottenham Hotspur. In this game, Tottenham Hotspur won 5-1 and there are certain characteristics that the "winning" passing networks share. This includes things such as the outside defensive players having average X and Y coordinates that are close to the striker. In the Tottenham Hotspur network, Rose and Walker both have an average X coordinate that is only slightly less than Tottenham striker Harry Kane. In terms of the losing passing network, it also shares certain characteristics. These include most of the nodes on the left side of the pitch as well as one of the attacking players being disconnected from the rest of the network.

A football field with blue circles

Description automatically generated

To classify these networks, I created two separate models. The initial model employs a K-Nearest-Neighbors model and incorporates centrality measures along with specific attributes as graph embeddings. The second model is also a K-Nearest-Neighbor model but utilizes a distance function based on the distances between players in each position and the corresponding passing weights.

**Data Collection**

Passing Network Construction

The data was collected through the usage of the StatsBomb API using live event play-by-play data from the entire 2015-2016 season of the English Premier League. Using the StatsBomb API, the data frame was filtered based off all passing events and the X and Y coordinates were extracted for the passing and receiving player. Next, the total number of passes for each player pair was then constructed to find the total edge weights.

Due to the difficulty of tracking substitute players within the passing network, each network is constructed up until the point a substitution is made and then no more data is collected within that passing network.

**Methods**

K-Nearest Neighbors Model Using Centrality Measures

In the centrality measure graph embeddings model, a data frame was constructed for each passing network consisting of the centrality measures for each team within a game with an additional score variable denoting the current state of the game. For example, the game will start off tied with a score of 0-0. Then, if the home team scores and makes the score 1-0 a new set of passing networks will be constructed with the home team network being denoted by a score variable of winning and the away team being denoted by a score variable of losing. A new set of networks is only constructed when a lead change occurs in the network.

After constructing each passing network, if there are players that never received the ball that set of networks are removed from the dataset due to the inability of being able to compute certain metrics within each network.

Next, I transformed each network into a directed graph object using the NetworkX library to compute various measures within the data frame. Initially, I computed the average betweenness centrality of the graph using the formula:   which calculates the betweenness centrality of each node. Next, I found the average of the overall network by evaluating . Next, I computed the maxium eigenvector centrality of the network by evaluating where A is the adjacency matrix of the graph and then computed to give the maximum eigenvector centrality. Then, I computed the maximum Katz Centrality through  as well as the page rank. Next, I computed the average clustering coefficient which can be defined as  . I also computed the average weight of each edge was then found through evaluating After that I computed the average closeness centrality by evaluating . Lastly, I computed the average in, out, and total degrees for each node. This procedure was done for both home and away passing networks.

After constructing the data frame for both the home away teams feature selection was performed using best subset selection which resulted in predictor variables of eigenvector centrality, page rank, average clustering, and the average edge weight. Then a 70/30 training test split was used when constructing the K-Nearest-Neighbors Model where Euclidean distance was used as the distance function. After Cross-Validation the optimal value of K was 12.

K-Nearest Neighbors Model with Linear Assignment

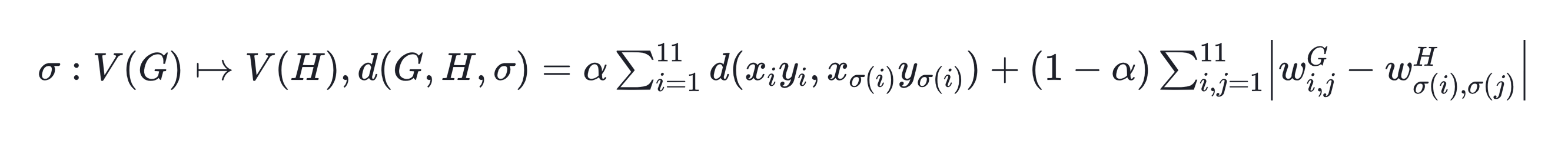
In the KNN model with linear assignment employed a similar data cleaning process for the networks. This model focus was on capturing the dissimilarity between two networks distances and passing networks through applying linear assignment. Instead of constructing a data frame for each individual network, I opted for a two-dimensional array for efficiency. In this model, my goal was to capture the entire network rather than abstracting it into centrality measures, as was done in the previous model.

The data was stored in a two-dimensional list, where the first value of the list was a dictionary encapsulating the passing network of the home team, and the second value was a dictionary encapsulating the passing network of the away team. Each dictionary contains a key representing the player, with its values being the average X and Y coordinates of the player, the score, and a nested dictionary. This nested dictionary has a tuple as the key, denoting the passing and receiving player, and the total passes for each combination of that tuple as its value. As for the score variable, this was denoted slightly differently. Each passing network was constructed with a binary variable of one for winning and zero for losing. Unlike the previous model, this network is every event in which a team scores. For example, if the game ends 1-0 where the home team scores in the 50th minute then the passing network constructed will be from the 1st to 50th minute and the home team will be denoted by a score of 1 and the away team will be denoted by a score of 0. The 50th minute to the end of the game, however, will be discarded due to there being no scoring events.

After constructing each passing network with its respective score, I added a dummy player for networks that were missing players from the data frame. Due to the data being unavailable for the average X and Y coordinates of these players, I placed them in the middle of the pitch and their pass counts were obviously zero.

To capture the dissimilarities between each network I constructed a cost matrix for the average X and Y coordinates in each passing network. This cost matrix was calculated by finding the Euclidean distance between the X and Y of each player for the home and away team. After the cost matrix is constructed, linear assignment is performed to assign each player in Graph G to Graph V in order to group each player by position. The distance is then computed using the sum of distances between each player and their assigned player. The same matrix is computed using passing counts instead of distances with the same linear assignment and then is summed up to compute the weights. The weights were then scaled by a factor of 1.5.

This can be described by the equation below:



to calculate the total distance for one network used in the KNN. After doing this, I calculated the distance between each graph in each set of networks, added them together, sorted them by the k smallest to find the k nearest graphs.

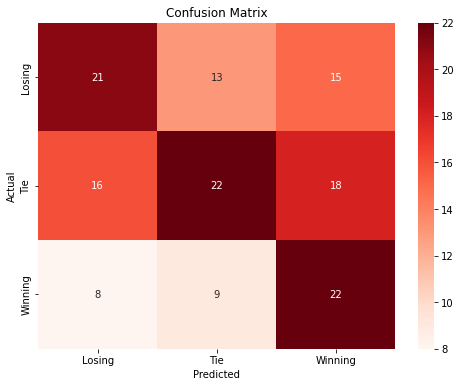
After 5-fold cross validation, a value of 7 was chosen as the value of k.

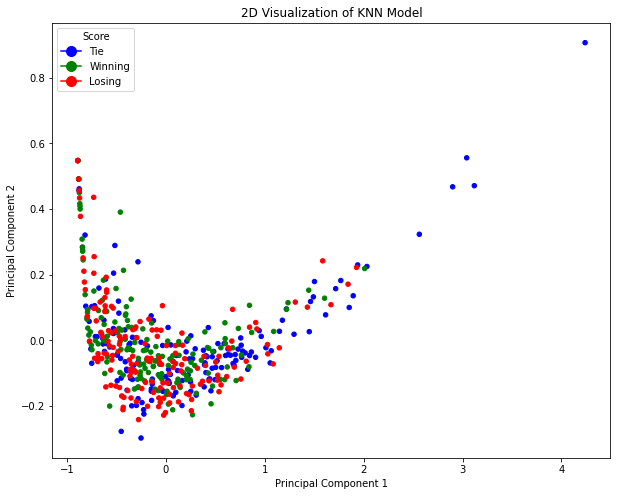
Several other variations of this model were attempted yet failed to yield any significant results. Some of these processes included omitting passing networks without all 22 players, analyzing one network at a time, as well as only considering either the player location or passing edge weights between players in the network.

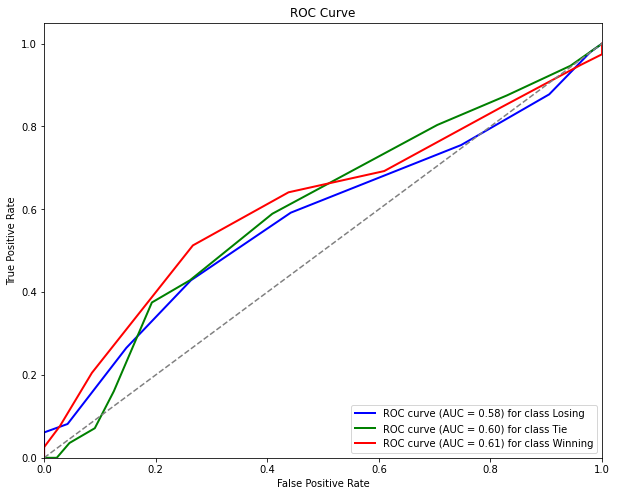
**Results**

K-Nearest Neighbors Model Using Centrality Measures

In the KNN with centrality measures I was able to predict the multinomial model consisting of win, loss, or ties with an accuracy rate of 45 percent after 5-fold cross validation. The binomial model however was significantly less successful with an accuracy rate of 58% after 5-fold cross validation.

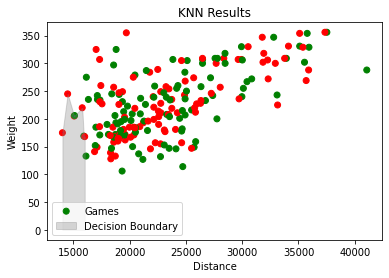


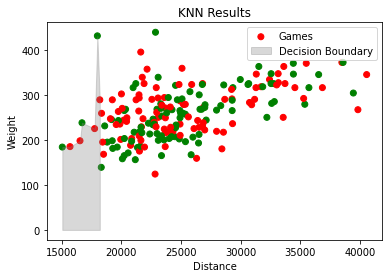
To help visualize these results I transformed the data into two principal components for visualization which gave us the respective ROC curves.

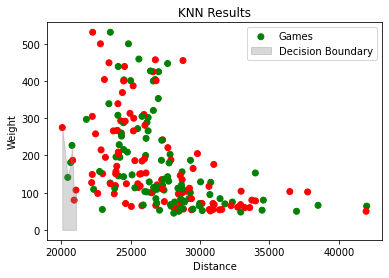


K-Nearest-Neighbor Model with Linear Assignment

After evaluating the model using 5-fold cross validation the results came out to a 60.66% accuracy rate. While this was lower than expected it was still better than the centrality measures mode. To visualize these results, the figures below represent the decision boundary with the five closest networks.







**Conclusion and Limitations**

The overarching goal of this research was to explore how the study of passing networks could be used to understand a soccer match and decide the match outcomes. To solve this problem two models were constructed consisting of a model using centrality measures and another model constructed using linear assignment.

After trying different combinations of features and models all these combinations failed to eclipse 70% accuracy consistently yet still showed some promising results. Through this research it is led to believe that the location and passing edge weights are only one part of the picture in winning matches. One reason for this could be due to number of goals scored on counter attacks. For example, one team might dominate in possession and control the game positionally, yet another team could score quickly on the counterattack. Another factor in these results could be due to the league where the data was acquired. Perhaps a slower paced league such as Serie A where more passes are completed would yield more promising results. There is certainly still room to be done on this topic and I hope to explore passing network related research in the future.

**References**

Aric A. Hagberg, Daniel A. Schult and Pieter J. Swart, [“Exploring network structure, dynamics, and function using NetworkX”](http://conference.scipy.org/proceedings/SciPy2008/paper_2/), in [Proceedings of the 7th Python in Science Conference (SciPy2008)](http://conference.scipy.org/proceedings/SciPy2008/index.html), Gäel Varoquaux, Travis Vaught, and Jarrod Millman (Eds), (Pasadena, CA USA), pp. 11–15, Aug 2008

Sahasrabudhe, Amod, and Joris Bekkers. "A Graph Neural Network deep-dive into successful counterattacks."

“Statsbombpy.” *PyPI*, pypi.org/project/statsbombpy/. Accessed 22 Jan. 2024.