Futbol Money

Group Project for Business Analytics Spring 2023

Ethan Bondick, Louis Cheng, Francesco Staudacher, and Mackenzi Stevenson

Contents

1. Project Goal
2. Data
3. Exploratory Data Analysis
4. Feature Selection
5. Lasso Regression
6. Clustering
7. Optimization
8. GUI
9. Conclusion

**Project Goal**

The goal of our project was to build the ideal soccer team to compete in the Premier League given a set of constraints. We focused on the constraints of budget and number of players in each position. We created a GUI that could easily be used by a team manager to craft the best team possible based on their set budget and formation of positions(goalkeeper, defender, midfielder, forward). This way, someone without any background in data analytics could derive use from our project. Our definition of ‘ideal’ or ‘best’ team is based on features that we found to be significant from the exploratory data analysis we conducted and feature selection.

We broke our goal into two main subproblems. We wanted our subproblems to bring unique value to the sports analytics realm. The first subproblem was to create and implement a ranking algorithm based on a calculated performance metric that would evaluate player performance based on the context of their team. The purpose of this is to adjust performance rankings biased by the team’s win percentage. For instance, we take into account the fact that a talented player on a team with a low win percentage might have lower stats. Our second subproblem was to create and solve an optimization problem that could take in selected features and return an optimal team of players. This optimization problem takes into consideration constraints of budget and the number of players per position.

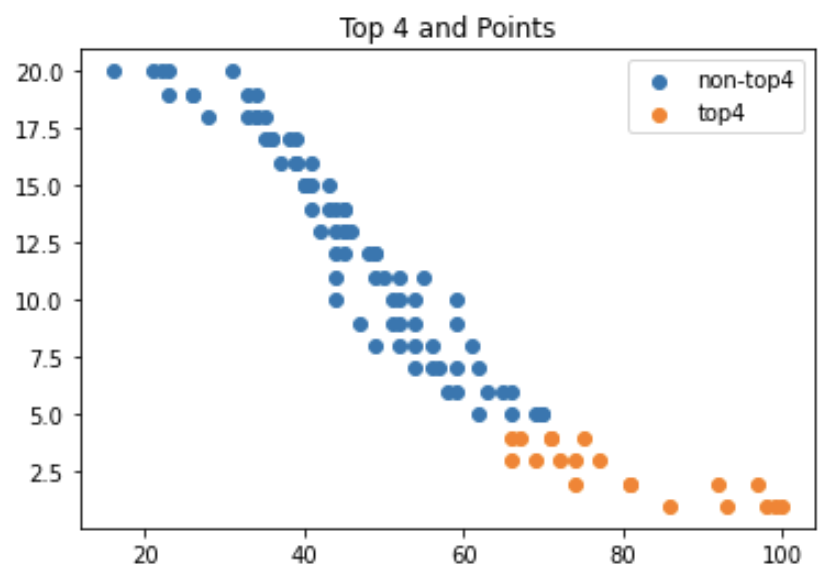
**Data**

Our data collection went through phases as we learned what we needed as we progressed through the project. Initially, we sourced our league, match, and player statistics from FootyStats (choosing 2017-2022 as our time window), an online database for soccer. As a secondary source, our team decided to scrape statistics directly from the Premier League website to fact-check and verify the results we were seeing. After careful analysis, with many of our results passing fundamental “logic,” we decided to proceed with creating an “average per game database” where we divided career-long statistics by appearances to normalize the data – this database would be used to figure out what individual statistics were most important to a team achieving their objectives.

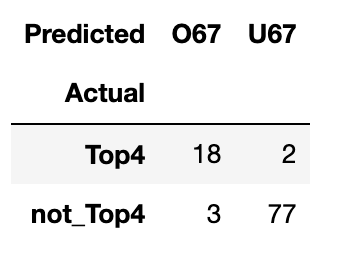
After statistical analysis, when tackling team building, it was necessary to pull player salary information for the players in our database – as the fundamental constraint for our team is budget. For each player in the league, only their most recent salary data was fetched, after which they were saved in our league-wide salary database.

**Exploratory Data Analysis (EDA)**

We conducted EDA to better understand our dataset, beginning with focusing on the team statistics. We split the dataset into two groups based on whether or not their league position was in the top four. In the Premier League, the teams who finish in the top four automatically qualify for the next season's UEFA Champions League group stages. We used EDA to predict a threshold for points needed to qualify for the championship. For context, each team plays 38 games in a season where they are awarded three points for a draw and one for a win. We formulated this problem as a classification problem where it is predicted that a team will finish in the top four (positives) if they reach at least a certain threshold of points.

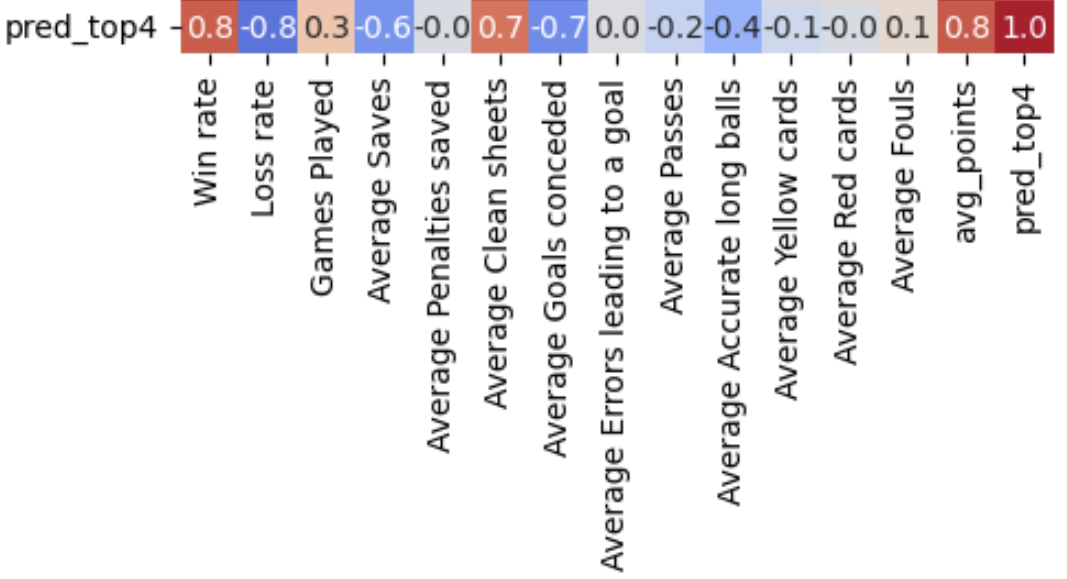


Based on the scatterplot, it is clear the threshold should be around 70 points. We used a contingency table to serve as a confusion matrix to evaluate each possible threshold value.

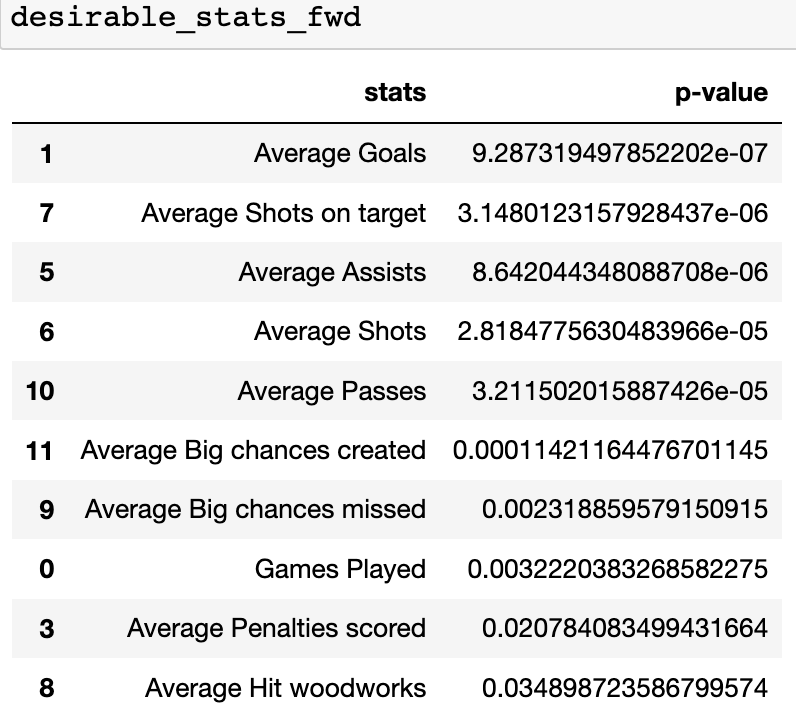
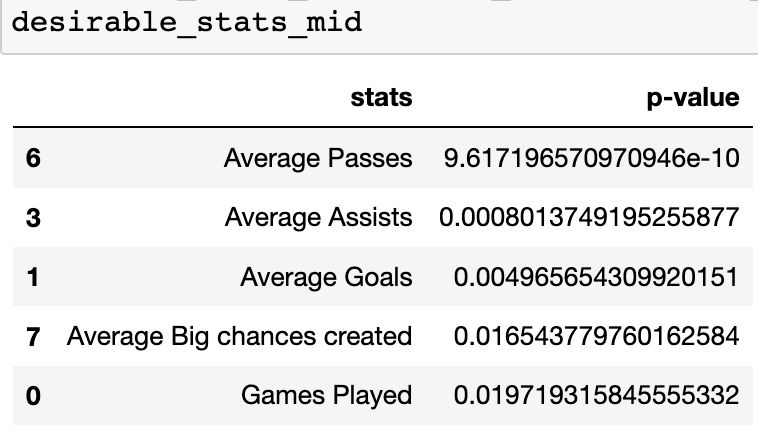
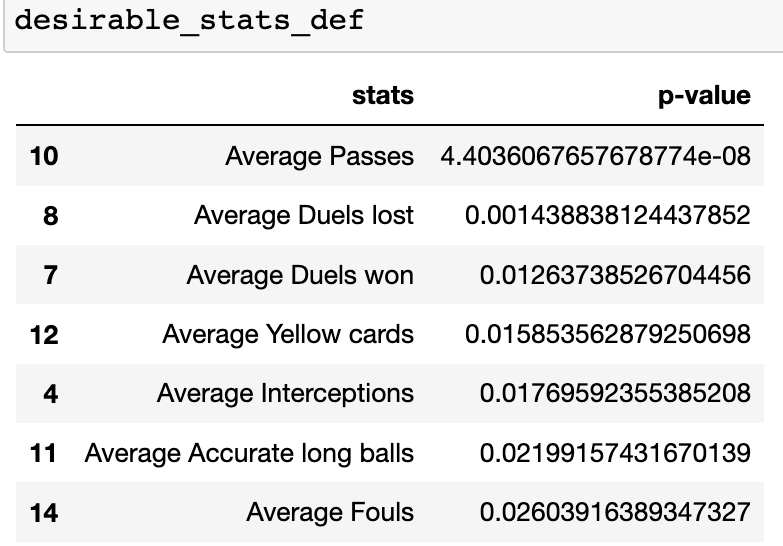
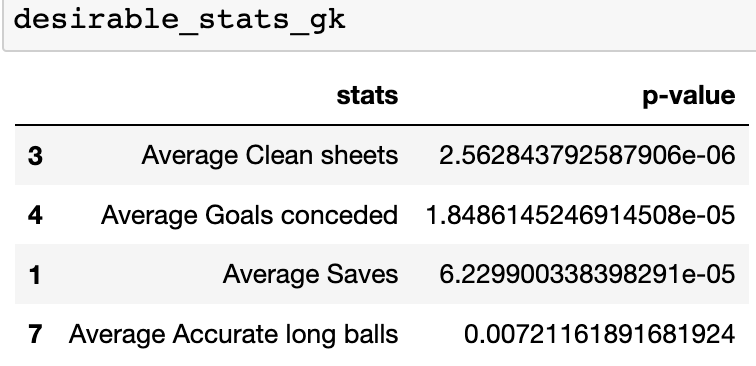


In the contingency table, false positives occur when the team scored over 67 points ("O67") but did not qualify for the Champions League ("Not Top4"), while false negatives occur when the team scored under 67 points ("U67") but still ranked top 4.The thresholds of 66 and 67 points both return an accuracy of 95%, but the threshold of 67 was chosen because it had more balance between false negatives and false positives: the confusion matrix indicates a false positive rate of 3.75% and a true positive rate of 90%. This distinction allowed us to estimate which players, based on average win/loss rates, bring enough points for their team to finish in the top four. We then used this information to perform a high level analysis of the players’ features that are most likely to help a team reach that goal, using a correlation matrix and a p-value analysis.

The correlation matrix, and more specifically its “top 4” row, is a visual starting point to easily determine whether there are attributes of players for each position that deserve a closer look. However, it is important to be aware that the results do not imply a causation relationship. For instance, the goalkeepers’ feature “average saves” has a highly negative correlation with finishing in the top four, even though saving a goal clearly does not negatively affect the chances of a team winning games. This is explained by the fact that better teams concede less shots, therefore even if their goalkeeper is highly rated they make less saves per game.



The P-value analysis we conducted was based on a t-test for the means of two independent groups, which were the players that average enough points for their team to finish in the top four and the ones who don’t. This is performed to find out which features are most likely to differentiate the two groups, i.e. the ones that have the lowest probability of having the same population mean given the data (p-value). The threshold for considering a variable significant was set to 0.05, so we highlighted the features that had p-value less than 0.05 for each position.



**Feature Selection**

During the feature selection, outcomes features such as (Win, Draws, Lose, Goals conceded, etc.) were dropped as they were features that cannot be controlled by a player. They are statistics that are an outcome of the actions the player made (eg. shots taken, tackles made, etc.). Furthermore other features that the player have no control over (eg. whether they start a game, when they get subbed out, etc.) were also dropped.

After that, we used FeatureWiz to first discard low information features, then find a subset of uncorrelated features from the features that were leftover. After the subset of uncorrelated features are found, it uses XGBoost (a “distributed gradient-boosted decision tree” model) to find the most important features for the target variable, in our case: wins. When analyzing the attackers the most important feature was “goals\_scored” and for defenders it was “clean\_sheets.”

**Lasso Regression**

Although FeatureWiz was already used to select features, we also used Lasso Regression to further shrink the number of features selected by FeatureWiz. LASSO regression was used because it provides interpretability and it has relatively high adjusted R^2 given the features it was provided by FeatureWiz. This is shown by the adjusted R^2 computed by the LazyPredictor package.

After running the LASSO Regression, the coefficients of the regressor were checked to select the most important features that have a positive effect on the target variable. From the analysis the most important features for goals scored was “shots\_on\_target\_total\_overall” and “shot\_conversion\_rate\_overall”, which had a coefficient of 2.44 and 0.76 respectively; and the most important features for clean sheets was “passes\_completed\_total\_overall” (2.74), “key\_passes\_total\_overall” (0.50), “shots\_off\_target\_total\_overall” (0.43) ,“assists\_per90\_percentile\_overall” (0.40), “min\_per\_card\_overall” (0.33), “duels\_total\_overall” (0.26).

**Clustering**

Using the information above, we dropped all the columns of the dataset, keeping only the features selected above by the LASSO regression. And using the new dataset we did clustering of players in each position (ie. Forward, Midfielder, Defender, Goalkeeper). The idea was to group players that are similar to each other given the features we selected above.

For each position, we selected an optimal number of clusters using the elbow method, and created clusters for the different positions. We then ranked each player in the clusters using a performance metric we created. The performance metric is calculated by multiplying each feature by a weightage set by the user. The initial idea was to find the best player in each cluster, given the salary constraint. We also output the cluster dataframes for later analysis - allowing the coach to look at the clusters of players after they run the optimization.

**Optimization**

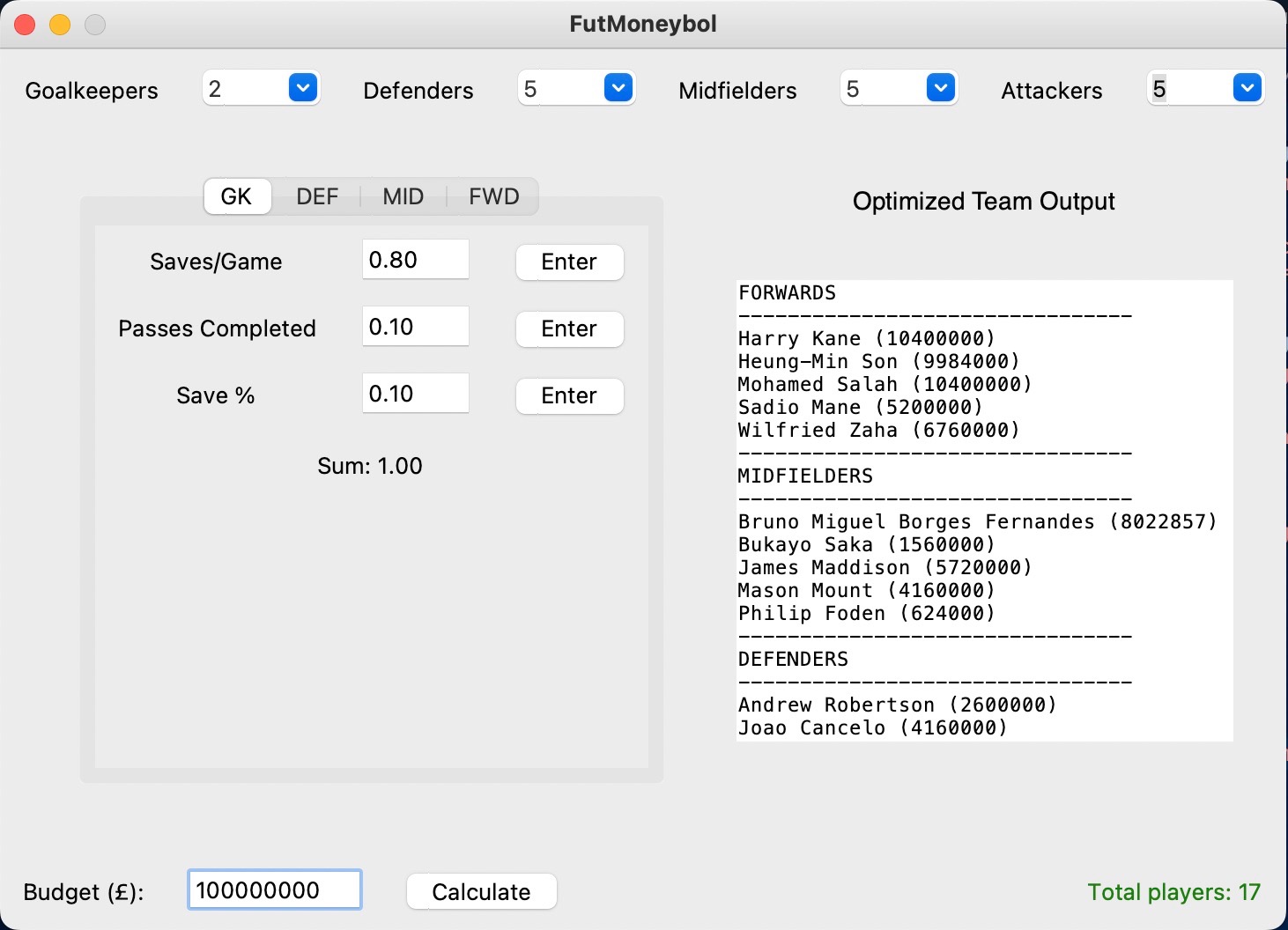
Our optimization function uses the Gurobi optimization software library. The goal of our optimization was to create a team building function that outputs a team with the highest total performance metric given a budget and a number of players per position. Our objective function maximizes the performance metric we created for the clustering algorithm for each player on the team. We added the constraints of a budget which caps the total sum of the players on the teams salaries. We also added constraints on the number of players for each position. The optimization is designed to be implemented into the GUI so that users can set these constraints based on the team they want to build. If a solution is unable to be obtained given the budget, the program tells the user to increase their budget. This function also outputs the total cost of the created team.



This is an example output of the optimization function for a team with 11 players (1 goalkeeper, 4 defenders, 4 midfielders, 2 forwards) with a budget of 50 million dollars.

**GUI**

The GUI adds significant value to our project. It allows a user to input a budget, the number of players per position, and weights for each feature that is statistically significant. These are the weights used to calculate the performance metric used in the optimization. The weights add up to one. The user then clicks the calculate button, and the optimal team for their selections is outputted. This is an example of what our GUI looks like to the user.



**Conclusion**

The value we were able to add through this project was creating a tool that managers can use to help make decisions regarding roster formation. We also were able to use unsupervised learning(clustering) to recognize players with similar stats at different price points. This is a large advantage for team managers in selecting players with a constrained budget. Essentially, we were able to implement data-driven decision making to scouting when this is often done largely through word of mouth. Of course, there are some intangible factors that our tool cannot account for, such as momentum in a season or chemistry among players on a team.