UCSD ML Bootcamp Capstone Proposal

Juan M. Tellez
January, 2020

Introduction

The goal is to create a project that will demonstrate ML acquired skills, and that will ideally be of some use.

Fantasy Premier League Recommendation Engine

The Fantasy Premier League is an online game that uses the results of the Premier League professional football (Soccer) in the UK. The general idea is that a person selects a virtual team from all the players in the league and modifies that team over the season to achieve a maximum amount of points. Points are obtained for having players that play a full game, that score, that have clean-sheets, that have assists, and possibly other variables. Every year millions of people play this game as a way of following the comings and goings of the league. Similar fantasy games exist for NFL, NBA and other leagues.

The purpose of this ML project will be to develop a model that recommends changes to a team over the season. There are 7 moves that are allowed over the season. Three of those moves can be done on a weekly basis, and four others can only be done once a season. The engine will be trained to predict what players will perform better than others over a period of 7-15 weeks, by including variables such as opponent team strength, home/away, the relative success of the team the player is in, and any other variables we can find that have an effect on performance. We understand that this won't be sufficient to pick optimal players, but we expect to layer either heuristics or additional ML models on top of player point predictions.

https://fantasy.premierleague.com/help/rules

Selecting the Initial Team

The initial team selection is done in September, and can probably benefit from heuristic algorithms as well as ML. There are already several projects in the community of data scientists addicted to FPL that have attempted to write code that selects the ideal team. Other data scientists have merely created notebooks that demonstrate which variables are correlated to successful teams, but then have selected the teams manually. For now we have decided to leave this problem for a future time, and to potentially simply achieve this with heuristics, a knapsack algorithm, or some other variation. Our goal is to optimize the weekly moves and to achieve ultimately high scoring teams by making optimal moves over the season.

Weekly Moves

- Transfers - Each team is allowed one free transfer per week. A transfer involves selling one player and buying another. Here the ML needs to select the player least likely to produce from the roster, and to replace it with a player most likely to produce. Noting that you only have 1 free transfer per week, the selection should not be short sighted. It needs to look forward to the schedule. Transfers may not exceed the maximum budget allowance for the team. Each player has an associated cost which changes during the season and the budget of the team is capped.
- Bench - Each team is allowed 11 of 15 players. Each week the manager can select which ones will play.
- Captain - Each week the team can double the points awarded to the player who is the captain. Selecting the player most likely to provide points on the next week on average can boost the team value.
- The game will auto sub for you if a player leaves the game.

Chips (one time moves)

There are some simple heuristics that may be useful to enhance the ML in this case. There are weeks when there are more games than other weeks. These are potentially sources of lots of extra games, if the player takes advantage of them. It is not obvious, however when to take advantage of the Free Hit. There are potentially 10s of strategies that could work.

- Triple Captain - Triple the points awarded to the captain that week
- Wildcard - Free unlimited transfers for one week
- Free Hit - Swap the entire team for only one week
- Bench Boost - Get points for your bench players

Data

The FPL makes data available via an API, and for 4 years vaastav (github) has been collecting the data for all the players, the schedule, and the individual professional teams. By the end of the semester we will have one more complete year (2020-21). The goal is to use 3 years to train the model, and the 2020-21 season as a way to test the model. The data is clean but sadly it is modest in size. We will augment the learning process by generating as many “training” teams as possible. A Static view of the data is available from: https://github.com/vaastav/Fantasy-Premier-League, dynamic streamed data is available through the API (see this article for reference to API points: https://medium.com/@frenzelts/fantasy-premier-league-api-endpoints-a-detailed-guide-acbd5598eb19).

We think we can use the years 2016-17, 2017-18, 2019-20 as training data, and 2020-21 as test data. Note also some players have summary data going back to 2012, that is the totals and averages on all the tracked categories for that player for that specific year. There are no details on the individual games available, so analysis on home vs away for years before the 2016-2017 season are not available. At this time I'm not sure whether that summary data can augment the more detailed data available for the subsequent years.

There are several papers, blogs and articles written about analyzing data in various ways, creating notebooks, and pivoting on specific variables on the data in order to find if there is correlation between some property of the data and the performance of a player or team. Most of these efforts focus on the vaastav collection of historical data, which started in 2016. We have, however, more information available from other sources that could be examined and potentially combined with the vaastav data. In particular I will examine this source: https://datahub.io/sports-data/english-premier-league

While there is only a modest amount of data from the point of view of the "years of data" and "number of games" … there is a great deal of data about each player and each team. It is possible that we may be able to find methodologies such that recommendation engine can suggest moves that are better than the average fantasy-player.

Machine Learning Problem

The goal for the model is to in essence predict which team configuration will maximize the points for a player. This section has a list of ideas about the problem. The ideas listed below may be competing or complementary.

- This is a regression problem, that is the model will try to maximize a specific numeric output that can be easily computed. The output is the number of points the player received.
- The problem can be reduced to thinking about how many points an individual player will produce on a given week.
- The Chips, that is the moves that are done only once per year, can be selected using heuristics, rather than ML, thus a complete recommendation engine would combine both approaches. For example:
    - For Triple captain - From all the players in your team, given your top point getters, if any of those point getters is playing twice in that game week (this happens occasionally during the year), and he is playing weak teams in both games, then select this player as triple captain.
    - For Wildcard - if the team has underperformed for three weeks in a row and more than ½ of the season has passed, then use your wildcard.
    - For bench boost - If this is a GW where all of your players are playing twice, including your bench, use your bench boost. Note that benchboost should probably be used in combination with wildcard.

    - For Free hit - If your players are not playing this coming week (short schedule), and no number of free trades will help you have a complete roster, take the free hit. That is if more than four players are in the schedule, either by injury, or by schedule.
- Since every fantasy team gets one free trade per week, it is possible to fully replace your team in 15 weeks, and it is possible to replace your team completely twice during the season. If you consider this, then a prediction of player performance needs to be accurate on average anywhere from 5-15 weeks, but not necessarily longer.
- Interaction between players in the team. It may go without saying, but it is likely that there are some interactions between members of the team. Say that you pick the best goalie in the league, which plays with the number 2 team. Say you also pick the best midfielder in the league, and he happens to play with the number 1 team. When they play against each other, the ML cannot just simply predict maximum value for both, since they oppose each other. The engine should perhaps consider benching one of the two. Or should it? Is this a heuristic, or is this a learning problem?
- The budget creates interaction between the members of the team that might be very difficult to calculate, but because you are always changing 1x1, that interaction is minimized when making the weekly trades. The same is not true when playing the chips, which allow you to choose completely new teams. For selecting the players to use in the chips, where full teams are selected, we will use our existing algorithms that are not ML. The two chips in question are the wildcard and the free hit.

Problem Simplification

In order to simplify the problem so that we can iterate over the solution and slowly improve on the product, we propose the following function.

```
Fp(P, GW) = points
```

Where Fp is a function, given parameters P (a player) and GW a game week for a specific year, is a prediction of the number of points the model believes that a player will get that gameweek. We believe that a simple function for Fp is:

```
FpAve(P, GW) = Sum(Gw-1, Gw-2 … Gw-n) / n
```

That is the average of the number of points over all known gameweeks where player P played. Then given the actual number of points for a given week: `FpActual(P, GW)`, Fp is better than FpAve if:

```
mod(FpActual(P, GW) - FpAve(P, GW)) > mod(FpActual(P, GW) - Fp(P, GW))
```

We propose that to evaluate an ML model that tries to compute Fp we need to have *n* predictions over *m* players where the model consistently does better than the FpAve. Given our limited data we will first train the model over 38 * 3 weeks, and evaluate all the players that can be found to have played over those number of weeks. On average there are about 500 players per year. This means we have approximately 57000 points.

Computing the Moves

The expectation we have is that initially the ML will not be in the business of computing the moves or learning the rules. The idea is that once we have an Fp() function that we can layer heuristics to select players to trade away, and trade in.

The recommendation engine should suggest 3 players to trade away, and for each of those 3 players, it will suggest 3 replacements. The human will make the final choice. After the season is over we will create synthetic teams that will compute against the best human teams in the league. We will attempt to create an AI driven team that will beat the humans using training only from the past seasons, and combining the heuristics and ML predictions. Our goal is to have a machine that enhances the experience of maintaining the team over the season, by reducing the search for players every week. A side goal that may be fun, will be to create a super team, and compete for the top prize next year!

Product

The product is intended to be released as a website that will provide advice to a customer. The customer will need to provide some means to authenticate with FPL in order to obtain the team details, or the team must be scraped somehow from the website. There are issues with authentication. I believe the best option will be to write an app, rather than a website, where that app relies on a service for computation & history. This way authentication happens only against the device used by the FPL subscriber.

Data Analysis

Addendum

A quote from an ML scientist doing the same project:

“My best approach ended up using a random forest to predict current week points, where strength of opponent was calculated as described, but also using other features. From those predictions I used a linear programmer to select the highest scoring 11 player team based on the predictions. I then added up the scores of what those teams would have gotten that week and compared them to myself, the league average for that week as well as the scout. My machine chosen team did not outperform the scout or myself but it easily beat out the average. So I was happy with those results!”

https://www.reddit.com/r/FantasyPL/comments/eeqgw1/fpl_api_question/

References

- Vaastav's github repo. The parent of all this work.
- Towards Data Science, Medium: FPL API tutorial
- Frenzlet's FPL API Guide, Medium
- VanHerle, Medium: FPL API authentication
- Splunk: ML Part1, winning at FPL
- Splunk: ML Part 2, winning at FPL
- Towards Data Science: AI top ten in FPL