**fplBuddy package by James Abrams**

Data Analyst at Hartley McMaster

james.abrams@hmcm.co.uk

**Introduction**

The aim of this fpl tool is to predict scores for each individual player for the upcoming gameweek. Using the calculated expected points, an optimisation exercise is carried out to determine which players should be selected for the fantasy draft to score the highest number of points. The tool can also be used to inform transfer process decisions, subject to how many transfers the manager in question has available to them.

FPL data is readily available, and an Api exists containing all the fundamentals required for this project. FPL data available includes info on individual player performances past and present, fixtures, teams, team strength indicators, and manager specific data. For a detailed guide on how to interact with the FPL Api, see the following page.

<https://medium.com/@frenzelts/fantasy-premier-league-api-endpoints-a-detailed-guide-acbd5598eb19>

**Methodology**

Two modelling approaches are employed to achieve the expected point return predictions.

* Odds Method

Takes the latest odds data into consideration. Odds data includes details for most likely goal scorers, most likely players to register an assist, and most likely teams to keep a clean sheet. The data is updated on a weekly basis for the upcoming fixtures, informed from live betting site odds. This data is lifted programmatically from the following two sources:  
goal scorers and assists -

<https://www.fantasyfootballpundit.com/premier-league-goalscorer-assist-odds/>

clean sheets –

<https://www.fantasyfootballreports.com/premier-league-clean-sheet-odds/>

This Mult-Factor Linear Regression model is trained on historical data from the 2021/2022 season. Coefficients and R squared value for the odds model are displayed below. Note that the 0.928 value is unrealistically high, as it is based on perfect information (e.g., odds of scoring equal to 100% where a player has scored in the denoted gameweek). A standardised approach is employed to scale the probability of each player scoring and assisting over a range from 0 to 100%, to improve accuracy in line with the aforementioned perfect information observed in the training data set.

Graphical user interface, text, application

Description automatically generated

R-Squared value: 0.928

Correlation for individual indexes. cs\_index has a higher correlation amongst Goalkeepers and Defenders, as more points are awarded to these groups for maintaining a clean sheet (4).



* Index Method

Considers various index metrics to inform predictions. Index metrics include:

* **“james\_points\_index”:** Metric based off of average points per game a player scored last season, and points recorded per game this season. In absence for data recorded last season (ie. For players who joined the league this year, Haaland, Darwin, average points scored per game this season is used by default). The ‘weight’ argument can be called to adjust emphasis on current form / historical performance.
* **“james\_ict\_index”:** Metric based on Average ICT scores. Accounts for last season average index, as well as average index for each match played for the current season. (Option exists to adjust this index with the ‘weight’ argument also)
* **“james\_strength\_index”:** Measure of difficulty for upcoming fixtures for the current gameweek. Takes into account whether fixture is home or away. Experimented with recovery time between matches, difficult to implement in practice (because of Champions league matches etc).

Once again, historical 2021/2022 season data is used to train the model. Coefficients and R squared values for the model are displayed below.

Graphical user interface, text, application, email

Description automatically generated

R-Squared value: 0.3065

Correlation for individual index metrics as predictor for total points are as follows.



Points and ict index have the strongest correlation – but strength index remains influential for predicting big points hauls, so should not be underestimated. Further analysis required to determine which fixtures result in big points hauls…

**Setting up the Optimisation problem**

**Objective Function:**

* The objective for our optimisation problem is to maximise expected points (xP)

**Key assumption:**

* Only the starting 11 players are considered towards the points and budget totals. Substitutes are assumed to be static, although if a player from the starting line up does not start for whatever reason, the subs will invariably be required.

**Constraints:**

* Team Constraint: Each team should be represented by no more than 3 players
* Positional constraints: One goalkeeper (GK),
* Greater than or equal to 3 Defenders (DEF), but less than or equal to 5,
* Greater than or equal to 3 Midfielders (MID), but less than or equal to 5,
* Greater than or equal to 1 Forward (FWD), but less than or equal to 3,
* Hard capped at 11 players across all positions
* Cost constraint: Budget for the 11 players (excluding subs) must not be exceeded
* Additional transfer constraint: Free transfers available to the user

Subject to the Objective variable and Constraints laid out above, the Rglpk\_solve\_LP() function from the Rglpk linear programming package is able to return a solution to the problem. The tool will detail the optimal solution, which 11 players to select, as well as expected team points total for the calculated selection (not accounting for captaincy double points).

<https://www.rdocumentation.org/packages/Rglpk/versions/0.6-4/topics/Rglpk_solve_LP>

**Results**

**Function Glossary**

Key functions used during development:

**devtools::document()** - Take care of the DESCRIPTION and NAMESPACE files

**check()** – Testing the package before it goes live

**use\_data()** - Add latest odds data to the package repo once it becomes available

Key functions for modelling: Note: fplBuddy functions are subject to be renamed.

**fplr package**

fplr::fpl\_get\_gameweek\_next()$id - Return current gameweek number

fplr::fpl\_get\_player\_all() - data from the fpl api about all players

fplr::fpl\_get\_teams() - get team info including strengths etc.

fplr::fpl\_get\_fixtures() - get fixtures info

**fplscrapR package**

fplscrapR::get\_player\_details() - get "player\_details" variable

**fplBuddy package**

fplBuddy::fpl\_gameweek\_predictions() - master function

fplBuddy::fpl\_get\_gw\_joined() - get "gw\_joined" - player\_details manipulation

returns details up to and including the specified gameweek

fplBuddy::fpl\_fixtures - Return all clubs fixtures within a denoted time period

fplBuddy::fpl\_odds\_generator\_gs() - transform goals and assists odds to required format

fplBuddy::fpl\_odds\_generator\_cs() - scrape clean sheet probabilities from the web

fplBuddy::fpl\_calculate\_predictors() - produce "players\_index"

Calculate each index method for our predictions

fplBuddy::str\_convert() - convert special characters in a string to regular utf-8 characters

fplBuddy::fpl\_calculate\_xP() - use indexes to calculate expected points

both methods included, odds method and index method

fplBuddy::fpl\_my\_budget() - Calculate user budget from transfer funds available and squad value

fpl\_get\_api\_response() - Use Python to return detailed user data from the Fpl API

fplBuddy::fpl\_optimise() - Find the dream team for this week

fplBuddy::fpl\_optimise\_my\_team() - Determine which transfers should be made

fplBuddy::fpl\_performance\_comparison() - Compare predicted points Vs actual points scored