
# Fantasy Premier League

**Names:**

Ahmed Nasser abdelkareem Mohamed ( 9 )

Abdelrahman Ahmed Mohamed Abdelfattah Omran ( 36 )

## Problem statement:

With over 6 million players, **Fantasy Premier League** is the biggest **Fantasy** Football game in the world, at this game users can choose their team which consists of real players from the Premier League each with a value and the user shouldn't exceed the amount of money in his bank and based on the actual performance of these real players the user gets points, one thing is that each weak the user has the option to replace one of his players by another and this action is very critical and has a lot of factors to take into consideration like players current form , his next matches difficulty , player's value , ownership (how many other players have this player ) and many other factors (like team captain) that makes one player better than another .

## Goals:

1. Help the users to make the weakly transfers
2. Help them with the timing of using special abilities like wildCard , triple captain

## Specifications:

- Based on an initial line-up and previous performance of each player we could predict a good transfer the user can make to get more points the next gameweek.
- Based on the below dataset we can get the performance of each player in the last 3 years and the current season against each team.

- Dataset

https://github.com/vaastav/Fantasy-Premier-League/tree/master/data

## Detailed description of the Dataset:

The data folder contains the data from the previous 3 seasons as well as the current season. It is structured as follows:

- season/cleaned_players.csv : The overview stats for the season
- season/gws/gw_number.csv : GW-specific stats for the particular season
- season/gws/merged_gws.csv : GW-by-GW stats for each player in a single file
- season/players/player_name/gws.csv : GW-by-GW stats for that specific player
- season/players/player_name/history.csv : Prior seasons history stats for that specific player.

# General Description :

- **Input :**
  - The Training process for the model will take a player for some GameWeek and input this players status at this GW to the Model , his status will be : player's form estimated by FPL official site, player's value , if player's value is increasing or decreasing ,ownership percentage ,his points avg and alot more .
  - Another information that will be fed to the model alongside with player's information is his team's status , avg form of the players , difficulty of the next couple matches, the position in the league and also other factors .

- **Output :**
    - The model should give a score of that player in the next GW
    - That score will be different from the score that the player would get in the GW , because the score will take into consideration other factors other than the next match , as it will take things like ownership percentage (how much will this player be differential) and couple matches ahead- not just the next match- .

- **Training :**
    - Given what the player has already done from the dataset we can train the model to predict his score .

- **The model in Action :**
    - The model will calculate a score for each player in the user's team , and the three with the least scores will be suggested to be replaced with those with the highest scores out of the current team with risk based on the current position in the global table .

## More detailed Description :

Let's start from The Dataset, the dataset has 4 season (2016/2017, 2017/2018, 2018/2019 and 2019/2020) with the current season to be updated each gameweek, for each season we have the data in two formats :

1- separated by gameweeks, where each gameweek contains the current information ( described below) of all players in this specific week and the score they got .

2-separated by players, where each player has a separate csv file with the history of his information and the score he got for each gameweek.

We will use the second format of the data in the following way :

The information(features) we mentioned above :

assists attempted_passes big_chances_created
big_chances_missed bonus bps clean_sheets
clearances_blocks_interceptions completed_passes
creativity dribbles ea_index element
errors_leading_to_goal errors_leading_to_goal_attempt fixture fouls
goals_conceded goals_scored ict_index id influence key_passes
kickoff_time kickoff_time_formatted loaned_in loaned_out
minutes offside open_play_crosses opponent_team
own_goals penalties_conceded penalties_missed penalties_saved
recoveries red_cards round saves selected tackled
tacklestarget_missed team_a_score team_h_score threat **total_points**
transfers_balance transfers_in transfers_out value
was_home winning_goals yellow_cards

These features are provided by the premier league official API each week for each player, we aim to predict a score of a specific player given some features.

These features will be a mixture between :

a) the previous gameweek (like big_chances_created, big_chances_missed, winning_goals, goals_scored, .. ), these will help the model recognize the current form of the player.
b) another will be from the current gameweek (like was_home, transfers_in, selected, opponent_team, .. ), these will help the model to recognize that specific game difficulty, For example : if the opponent_team is Liverpool it is more likely that the score will be low, another one : if there is huge transfers_out for specific player, that means that he might not play.
c) and a third group of features that will be calculated from all the preceded rounds (like the mean and the variance of goals_scored, total_points, ..) , the variance will define if the player is consistent with his performance or not and the mean shows if that performance is good or not.
d) And the last group will be from the future gameweeks : (like opponent_teams in the upcoming 3 matches )

These features will be fed to a fully connected neural network (there will be a model for each of (midfielders, defenders,attackers) as their effective features are different.

One important thing to know here is that the game is not about the points only, for example if you had selected a player with high **selected** value then most of the other users will get the points you got, so the ground truth should not be only **total_points** instead it will be a **function** F( weighted **total_points** in the upcoming 3 gameweeks**, selected, value**), this function will represent the value of buying this player at this gameweek.

Let's make it clear by an example :

**Jamie Vardy in 2018/19 season :**

The following are some of the features of the player in gameweeks 35 and 36 :

In this example we want to predict the function mentioned above of GW36 so the fields with (?) will not be given from the API in GW36 .

| Features | GW 35 | GW 36 |
|---|---|---|
| assists | 0 | 0(?) |
| attempted_passes | 12 | 12(?) |
| big_chances_missed | 0 | 2(?) |
| bonus | 3 | 3(?) |
| bps | 31 | 52(?) |
| completed_passes | 11 | 7(?) |
| creativity | 26.7 | 30.4(?) |
| dribbles | 1 | 1(?) |
| element | 234 | 234(?) |
| fixture | 349 | 355(?) |
| goals_conceded | 2 | 0(?) |
| goals_scored | 1 | 2(?) |
| ict_index | 11.1 | 20.7(?) |
| influence | 41.4 | 71.8(?) |
| key_passes | 2 | 3(?) |
| minutes | 90 | 90(?) |
| offside | 2 | 1(?) |
| opponent_team | 19 | 1 |
| selected | 597135 | 612205 |
| target_missed | 1 | 3(?) |
| team_a_score | 2 | 0(?) |
| team_h_score | 2 | 3(?) |
| threat | 43.0 | 105.0(?) |
| total_points | 9 | 13 (?) |
| transfers_balance | -279714 | -493 |
| transfers_in | 12656 | 43315 |
| transfers_out | 292370 | 43808 |
| value | 90 | 90 |
| was_home | False | True |

**To train the model the input vector will be the following :**

**From GW35 :** total_points(9), threat(43.0), influence(41.4), goals_scored(1), assists(0), bonus(3), …

**From GW36 :** was_home(True), transfers_balance(-493), selected(612205), opponent_team(1), …

**From GW1 to GW35 :** mean(total_points), var(total_points) and so on for those features that show performance consistency.

**The loss will be computed between the Function mentioned above on GW36 features and the model output**

## Evaluation :

- As mentioned above the dataset has the data of 4 seasons(including the current season), so we will train and validate the model using the 3 seasons and we will leave the current season for testing
- The testing will be as follows
    - For each gameweek we will run the model on each player to get his score (the score calculated using the function mentioned before) .This score will identify the player in the model.
    - Then we calculate the function using the actual values.
    - Then for each range of values, The ratio between the sum of the top 5 predicted scores and the actual top 5 scores should be a good indicator of how good the model
        - Why top 5 ? because the game has some restrictions that will make transfers forbidden like the number of the players from one team .
        - Why do we use the sum of the scores ? to make it less affected by the abnormalities like if a player with small value that has a bad history performs well in a single week, so it doesn't matter if the output of the model is not the same as the actual best players.
        - What are the ranges of values ? For example, we might divide the players into three categories based on their values to be sure that the

model works well for players with low values as well as players with high values.
    - Evaluating the model with this model compares the result with the optimal output so the ratio is not expected to be very high
- For the model to give correct predictions that differentiate the user from other users we will care about the ratio between the true positive to the true positive and false negative which implies taking the **Recall** as an evaluation metric.

## plots/graphs will be used :

- Comparing the accuracy of our model with the accuracy of the average player and a top player's choices.
- While training we will plot the training error and validation error for each epoch to define if our model has high variance or high bias or neither.
- We will need to plot the cost/error of the training and validation as the model capacity/degree increases to define what is the best poly degree to use .
- Plotting players' curve of predictions along side with their actual performance
- Using the learned weights we can define the features that affect the performance most and draw graphs that give these statistics.

## Graduation project :

- Both of us are in the same team in the graduation project ,and it is about deep faking in which we use different architectures to provide a fake video/audio of a person using only one image of him .