2019

Predict the best team for Fantasy Football

GROUP ASSIGNMENT 2 – WQD7011 NUMERICAL OPTIMIZATION

Member:

1. Aswadi Abdul Rahman (WQD180082)
2. Lee Kwan Li (WQD180019)
3. Chai Kun Ting (WQD180040)
4. Zulkanain Hasan (WQD180031)

Contents

1. Introduction

Fantasy football is a game in which the participants serve as the general managers of virtual professional gridiron football teams. The competitors choose their team rosters by participating in a draft in which all players of a real football league are available. Points are based on the actual performances of the players in the real-world competition.

Once you register yourself for this you are provided with virtual money which you need to spend on selecting 15 players from the 20 teams who are in the Premier League.

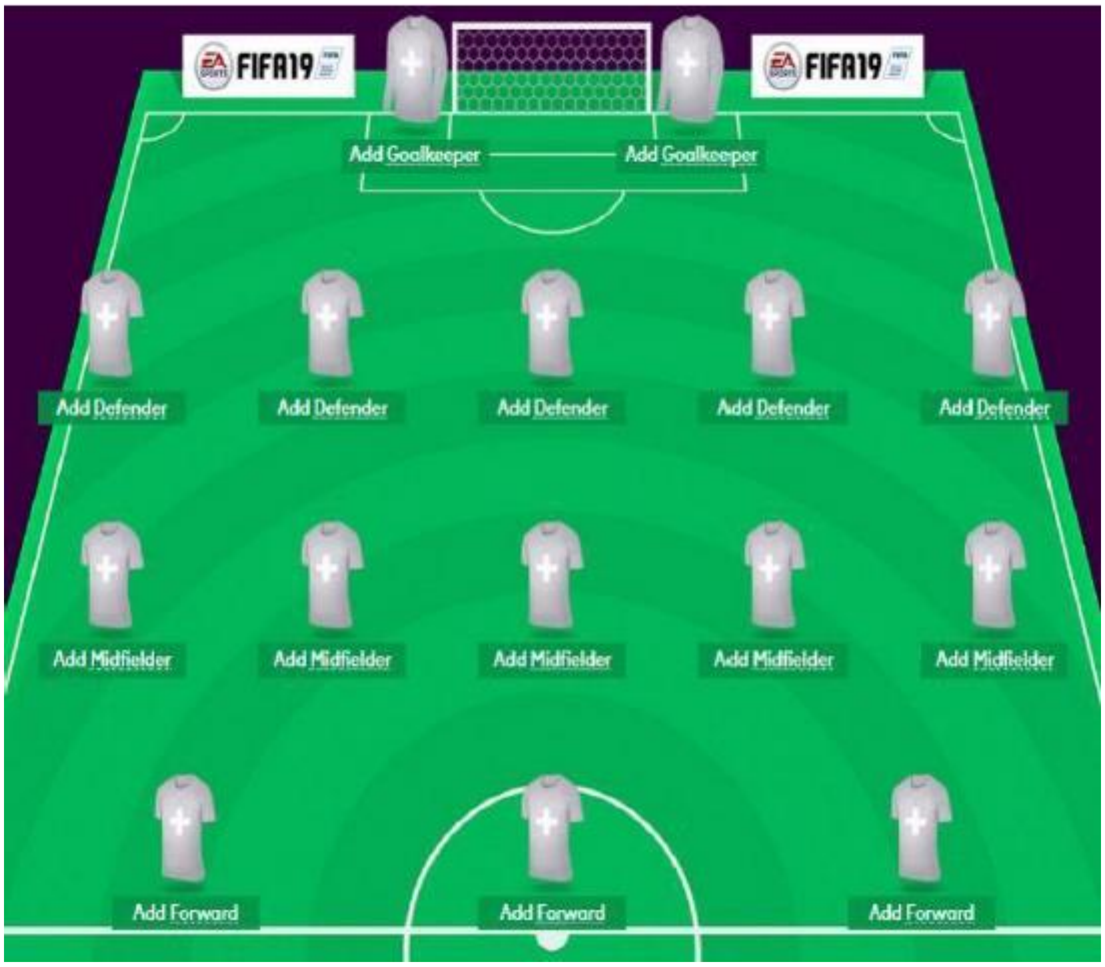


Figure 1: Fantasy football player's selection GUI

2. Dataset

Source from https://www.kaggle.com/delayedkarma/fantasy-premier-league-20182019/downloads/fantasy-premier-league-20182019.zip/6

There will be 21 columns/variable and around 500 rows (depend on which week we choose) inside the dataset. For simplicity we will just choose 8 columns and week 0 (FPL_2018_19_Wk0.csv) for this assignment. Below are the first 10 samples.

No	Name	Team	Position	Cost	Assists	Yellow_cards	Minutes	Points
1	Adam Smith	BOU	DEF	45	3	6	2067	56
2	Adrian	WHU	GKP	45	0	2	1710	72
3	Aguero	MCI	FWD	110	6	2	1960	169
4	Ake	BOU	DEF	50	3	5	3352	102
5	Albrighton	LEI	MID	55	8	5	2533	107
6	Alderweireld	TOT	DEF	60	0	3	1177	43
7	Alexander-Arnold	LIV	DEF	50	2	3	1573	83
8	Alisson	LIV	GKP	55	0	0	0	0
9	Alli	TOT	MID	90	13	7	2957	175
10	Alonso	CHE	DEF	65	2	6	2855	165

3. Objective Function

Given a budget (total cost) 1200 and 15 players for each team constraints, we want to maximize the point earn by using Constrain Optimization (Linear Optimization) algorithm. The 15 players should be divided into below criteria: (2GKP, 5DEF, 5MID, 3FWD)

- 2 x Goal Keeper position (GKP)
- 5 x Defender position (DEF)
- 5 x Mid field position (MID)
- 3 x Forward position (FWD)

By representing each player/row into different variable we can derive point maximizing using below mathematical equation

$$56*x_0 + 72*x_1 + 169*x_2 + 102*x_3 + \ldots \text{Points}*x_n$$

4. Constraints

Below are all the constraints that we need to use while trying to find the maximum points.

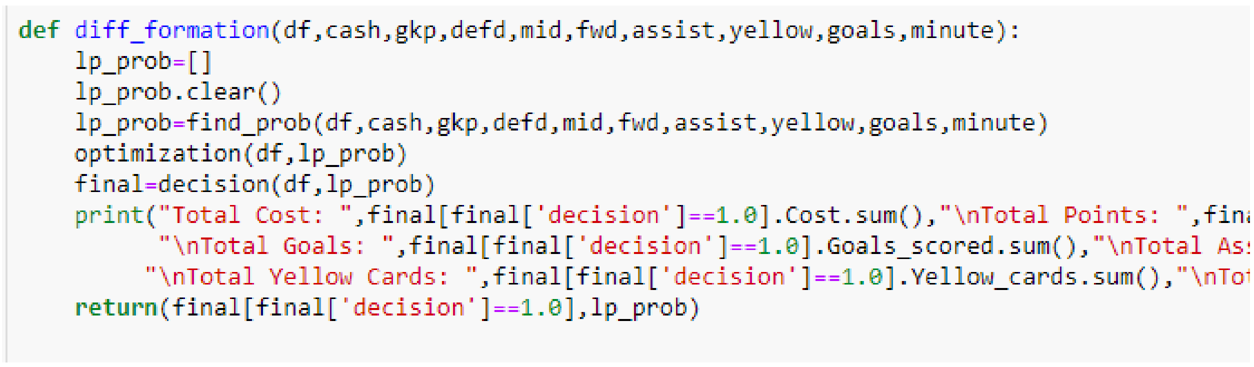
1. Cash constrain: $45*x_0 + 45*x_1 + 110*x_2 + 50*x_3 + \ldots \text{Cost}*x_n <= 1200$
2. GKP Player Position: $x_1 + x_{30} + x_{38} + x_{48} + \ldots x_n$ (GKP position ONLY) = 2
3. DEF Player Position: $x_0 + x_3 + x_5 + x_6 + \ldots x_n$ (DEF position ONLY) = 5

4. MID Player Position: $x_4 + x_8 + x_{10} + x_{11} + \dots x_n$ (MID position ONLY) = 5
5. FWD Player Position: $x_2 + x_{12} + x_{15} + x_{18} + \dots x_n$ (FWD position ONLY) = 3
6. Total assist constraint: $3*x_0 + 6*x_2 + 3*x_3 + \dots \text{Cost}*x_n >= 90$
7. Total yellow card constraint: $6*x_0 + 2*x_1 + 2*x_2 + 5*x_3 + \dots \text{Cost}*x_n <= 20$
8. Total goals scored constraint: $x_0 + 21*x_2 + 2*x_3 + \dots \text{Goals\_scored}*x_n >= 150$
9. Total minutes played constraint: $2067*x_0 + 1710*x_1 + 1960*x_2 + 3352*x_3 + \dots \text{Cost}*x_n >= 44100$

5. Python

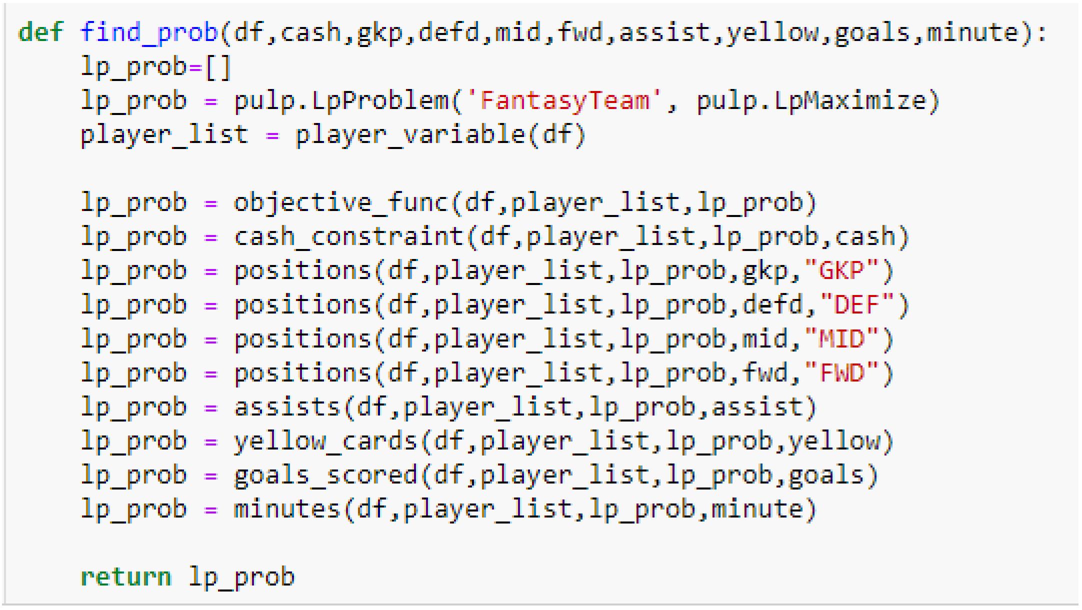
The main code start initiated from this code

final, lp_prob = diff_formation(df,1200,2,5,5,3,90,20,150,44100)



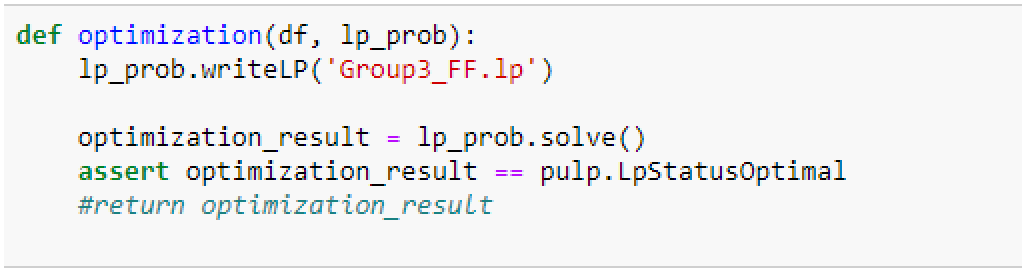
```
def diff_formation(df,cash,gkp,defd,mid,fwd,assist,yellow,goals,minute):
    lp_prob=[]
    lp_prob.clear()
    lp_prob=find_prob(df,cash,gkp,defd,mid,fwd,assist,yellow,goals,minute)
    optimization(df,lp_prob)
    final=decision(df,lp_prob)
    print("Total Cost: ",final[final['decision']==1.0].Cost.sum(),"\nTotal Points: ",fina
          "\nTotal Goals: ",final[final['decision']==1.0].Goals_scored.sum(),"\nTotal As
         "\nTotal Yellow Cards: ",final[final['decision']==1.0].Yellow_cards.sum(),"\nTo
    return(final[final['decision']==1.0],lp_prob)
```

Then after wards the code will call above funtion and it will start with creating mathematical equations from each constraint.



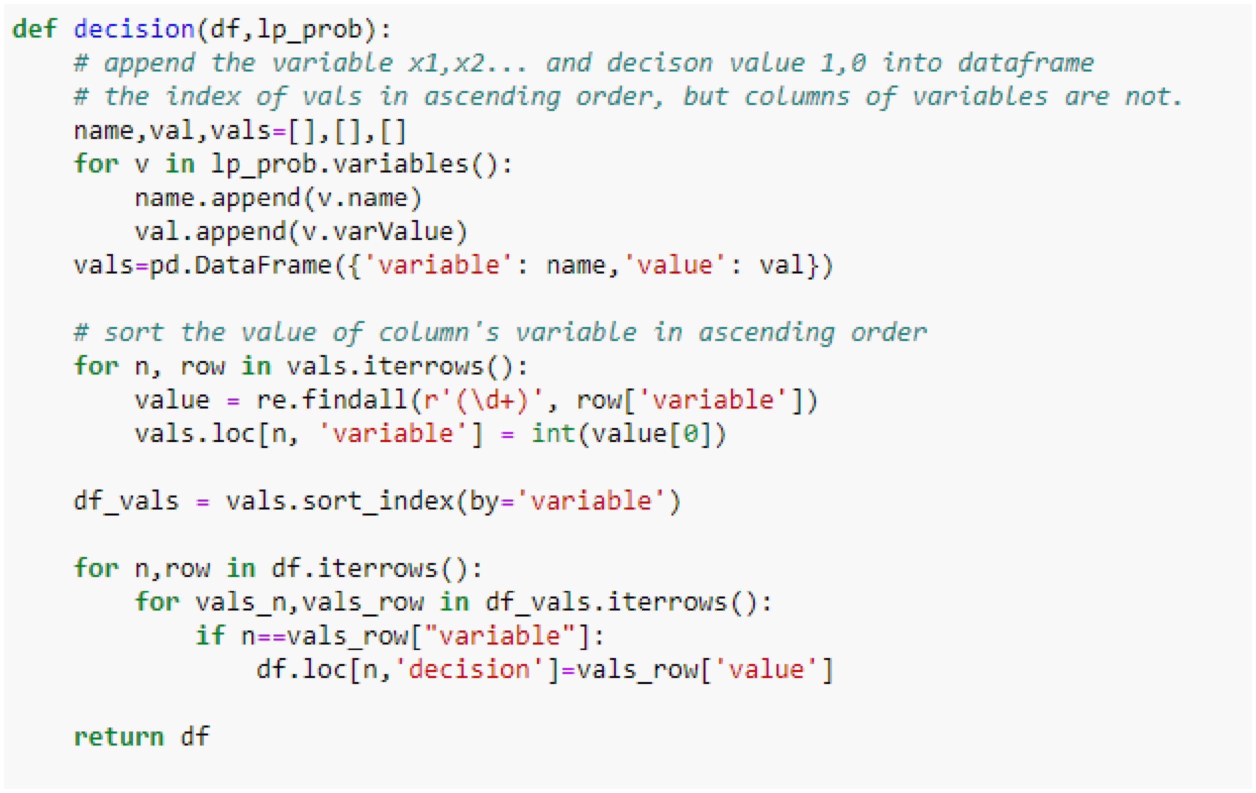
```
def find_prob(df,cash,gkp,defd,mid,fwd,assist,yellow,goals,minute):
    lp_prob=[]
    lp_prob = pulp.LpProblem('FantasyTeam', pulp.LpMaximize)
    player_list = player_variable(df)

    lp_prob = objective_func(df,player_list,lp_prob)
    lp_prob = cash_constraint(df,player_list,lp_prob,cash)
    lp_prob = positions(df,player_list,lp_prob,gkp,"GKP")
    lp_prob = positions(df,player_list,lp_prob,defd,"DEF")
    lp_prob = positions(df,player_list,lp_prob,mid,"MID")
    lp_prob = positions(df,player_list,lp_prob,fwd,"FWD")
    lp_prob = assists(df,player_list,lp_prob,assist)
    lp_prob = yellow_cards(df,player_list,lp_prob,yellow)
    lp_prob = goals_scored(df,player_list,lp_prob,goals)
    lp_prob = minutes(df,player_list,lp_prob,minute)

    return lp_prob
```

Above are main function to get the mathematical equation of each constraint and append in into the result which in here lp_prob.



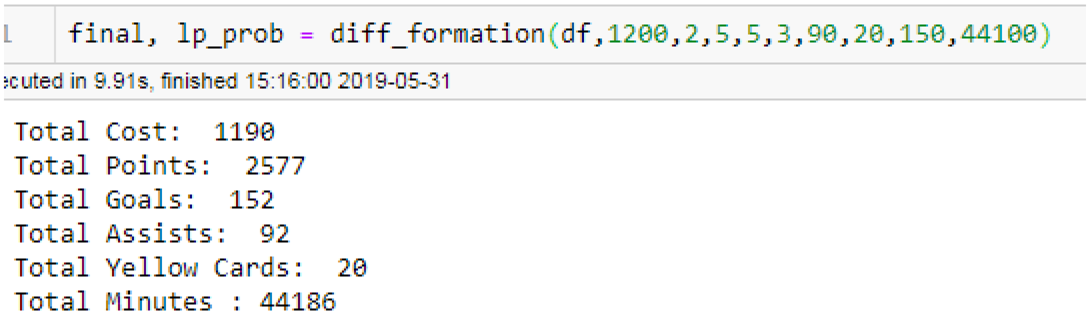
```python
def optimization(df, lp_prob):
    lp_prob.writeLP('Group3_FF.lp')

    optimization_result = lp_prob.solve()
    assert optimization_result == pulp.LpStatusOptimal
    #return optimization_result
```

Then all the constaint mathematical equations is being solve by optimization function which solve the Linear Optimization equations as shown in above code.



```python
def decision(df,lp_prob):
    # append the variable x1,x2... and decison value 1,0 into dataframe
    # the index of vals in ascending order, but columns of variables are not.
    name,val,vals=[],[],[]
    for v in lp_prob.variables():
        name.append(v.name)
        val.append(v.varValue)
    vals=pd.DataFrame({'variable': name,'value': val})

    # sort the value of column's variable in ascending order
    for n, row in vals.iterrows():
        value = re.findall(r'(\d+)', row['variable'])
        vals.loc[n, 'variable'] = int(value[0])

    df_vals = vals.sort_index(by='variable')

    for n,row in df.iterrows():
        for vals_n,vals_row in df_vals.iterrows():
            if n==vals_row["variable"]:
                df.loc[n,'decision']=vals_row['value']

    return df
```

Then lastly it will use above code decision function to itterate the mathematical equations and form dataframe for all the players/variable.



```
final, lp_prob = diff_formation(df,1200,2,5,5,3,90,20,150,44100)
```

ecuted in 9.91s, finished 15:16:00 2019-05-31

```
Total Cost:  1190
Total Points:  2577
Total Goals:  152
Total Assists:  92
Total Yellow Cards:  20
Total Minutes : 44186
```

Above snapshot shows that the main code run successfully and print some of the constraints that we use.

6. Justification

After running the whole script, below are the output and basically the chose player.

No	Name	Team	Position	Cost	Assists	Goals_scored	Yellow_cards	Minutes	Points
1	Azpilicueta	CHE	DEF	65	6	2	1	3330	175
2	Bertrand	SOU	DEF	50	5	0	2	3135	103
3	Christensen	CHE	DEF	55	0	0	0	2067	79
4	Daniels	BOU	DEF	45	3	1	0	3001	94
5	Eriksen	TOT	MID	95	10	11	0	3218	199
6	Fabianski	WHU	GKP	45	0	0	0	3420	157
7	Firmino	LIV	FWD	95	8	15	1	2760	181
8	Kane	TOT	FWD	125	2	29	5	3074	217
9	Mahrez	MCI	MID	90	13	12	2	2948	195
10	Ryan	BHA	GKP	45	0	0	0	3420	146
11	Salah	LIV	MID	130	12	32	1	2905	303
12	Son	TOT	MID	85	8	12	0	2292	178
13	Sterling	MCI	MID	110	17	18	3	2584	229
14	Vardy	LEI	FWD	90	2	20	3	3248	183
15	Walker	MCI	DEF	65	6	0	2	2784	138

So we test our output based on the constrain that we define previously.

Constraint	**Plan**	**Actual**	**Status**
Cost	<= 1200	1190	OK
Goals	>= 150	152	OK
Assists	>= 90	92	OK
Yellow Cards	<= 20	20	OK
Minutes	>= 44100	44186	OK

We also test for the position constraint that we define earlier

Constraint	**Plan**	**Actual**	**Status**
Goal Keeper (GKP)	2	2	OK
Defender (DEF)	5	5	OK

Mid Field (MID)	5	5	OK
Forward (FWD)	3	3	OK

The actual number that we calculate satisfied our constraint and we got **2577** points.

7. Conclusion

Linear Optimization can be used as a tool to solve for most of constraint problem. Below are the players that being chose by the algorithm.

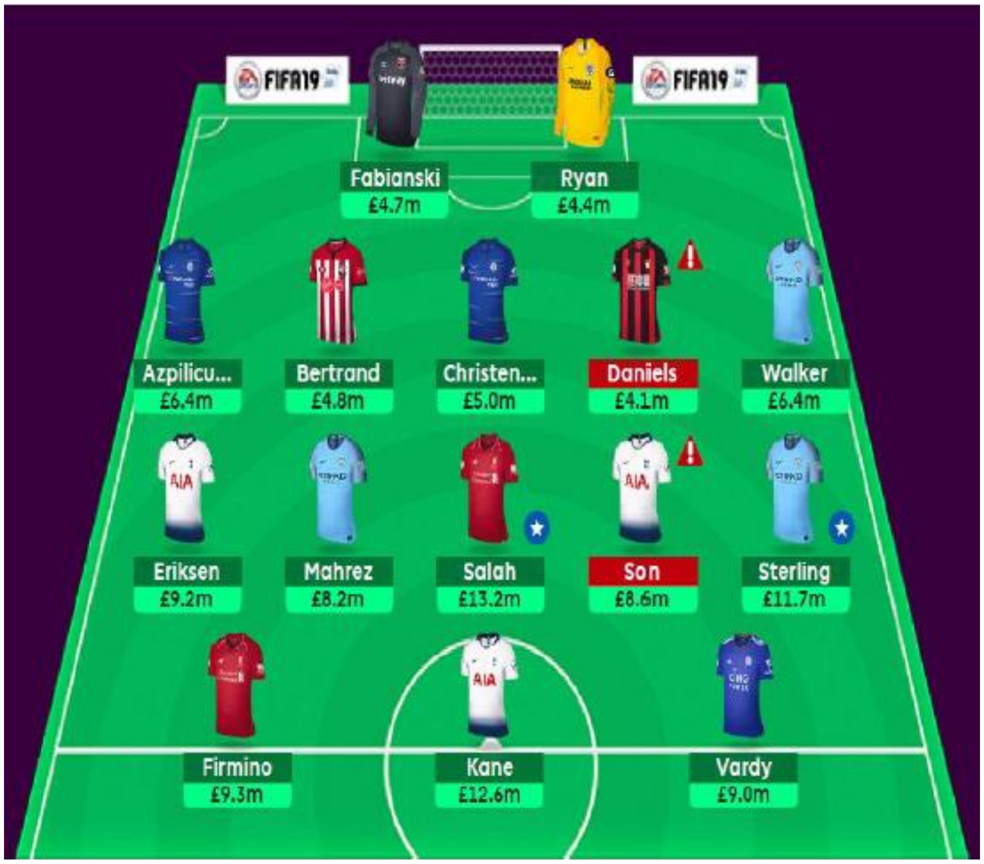


Figure 2: Final selected player in Fantasy Football GUI