Project Report

# GitHub URL

(<https://github.com/NiallMcknight/Assignment/upload>)

# Abstract

(Short overview of the entire project and features)

Every underage soccer tournament comprises of a group stage were preselected teams play each other to establish their position. On completion of the group games all teams enter a tournament competition based on their respective group position ie position 1 teams in group A will be paired with position 1 teams in group B, position 2 teams in group A will be paired with position 2 teams in group B and so on.

In big tournament’s such as the Foyle cup the group results are available online via a website but with smaller events these may be emailed in excel format. Smaller competitions normally don’t show group position or tables, only a list of results on a particular day.

The project basically has four stages:

* **Import data**: Get match fixtures and results from web or excel file source which is generally available for all competitions. Code development should allow for simple transfer between indifferent data sources and filter conditions.
* **Group section** **tables**: Interpret the group match results to provide group tables and team position, inclusive of games played and goals scored or conceded. Points will be allocated based on the result as follows: win = 3points, draw = 1point and lose = 0points. The criteria for position within the group table shall be based on total points, goal difference in that order. All group data shall be visible via a table labelled ‘DictTable’.
* **Tournament section**: On update of the group match results and associated group table each team will be assigned to the relevant tournament based on position within their respective group. The section should allow for an easy update of the relevant tournament name and number of allowed teams to future proof the code. **Note:** Due to the number of competitors the highest finishing teams may be required to enter a competition above their respective group position, ie: a team in position 2 may be required to enter the position 1 tournament etc. Goal difference shall be the relevant criteria for selection in this case. The tournament ‘Cup’ pairings will also be previously completed based on group name. All other competition pairings are based on a merged tournament table labelled ‘Grouppos’.
* **Stats:** The data accumulated and manipulated via the above entities shall be analyzed for feedback to the relevant teams and as a method of potentially predicting the result of the tournament matches. Not for betting but to avoid entering the incorrect standard of tournament or a team potentially being paired with the higher placed teams within that section, thus on the right side of the draw.

In 2018 for example we won 4 of our 6 group games subsequently we entered a tournament section at too high a level and got annihilated, completely demoralizing the young lads. If we had known our position after game 2 of the group stages in 2022 we could have had a bye into the final after only one tournament section game resulting in a rest day and less travel as it was known 2 teams hadn’t shown-up.

# Introduction

(Explain why you chose this project use case)

I am a ‘soccer dad’, who helps with the coaching of our local team which my kids play on. My employment is as an automation engineer for a large medical device company and I have previously complete an introductory certificate in python with udemy which covered concepts including lists, dictionaries and slicing data but the majority of the tools utilized on the UCD certificate were new to me. I’d refer to my python knowledge as beginner.

Throughout the duration of this course I was actively trying to acquire resource data for the assignment and whilst within my workplace huge quantities are stored these were not available at my time of asking.

Over the summer 2022 my son competed for Tummery Athletic U12 in the Foyle cup and I assisted with the coaching. The Foyle Cup is a large competition organized by Derry City Council but normally Tummery Athletic U12’s participates in smaller tournaments roughly one a month which are generally organized by independent clubs. Having experienced disappointed in the tournament section of the Foyle Cup in 2018 we were keen as coaches to have the team play at their own level following the group stage.

A review the Foyle Cup 2022 fixture website below at this stage will show the competition is complete with all group tables and the tournament sections populated. The problem statement for my assignment however was established mid-tournament as with 14 groups, 56 teams and goals difference a deciding factor it was impossible as a coach to decipher the information and establish which tournament our team would enter even with the group tables updated daily. Small competitions organized by independent clubs are generally worse as no group tables are provided, only results.

The main aim of the assignment was to establish a method of analyzing the match results as a means to determine with tournament section our team would enter but also in the event Tummery Athletic host a tournament provide a software developed environment for teams to monitor group positions similar to the Foyle Cup.

https://foylecup2022.torneopal.com/taso/sarja.php?turnaus=FC\_0003&sarja=U12

# Dataset

(Provide a description of your dataset and source. Also justify why you chose this source)

As established the 2022 Foyle Cup U12 section has 14 groups and 56 teams with the tables presented on the available website previously declared. My dataset could have been scraped from that webpage but to allow for future tournaments I used a for loop and enter each group table individually then acquire only the match results as a means to future proof the software development.

My dataset therefore is scraped from individual webpages showing the match results of the 14 Foyle Cup U12 group tables to include title. I scraped the home and away teams together with the result for the match played based on the identifiers below:

* ‘title’ The name of the group
* ‘ml\_kotisiisti’ = Home team for the fixture
* ‘ml\_vierassiisti’ = Away team for the fixture
* ‘ml\_tulosklo’ = result, which includes the home and away team scores

# Implementation Process

(Describe your entire process in detail)

My code development began by establishing and labelling the entities of the process each to be identifiable as a class thus keep my code organized and understandable. Import data=’groupclass’, Group Section Table=’generatetable’ and Tournament Section=’competitions’, as detailed previously and in reference to my development code. As there are 14 groups I generated a for loop with the range of teams and formatted the url path for each occurrence. The layout description is as follows and should be read in conjunction to my code:

**Import Data(‘groupclass’):** Using the python tool ‘BeautifulSoup’ and the identifiers from the Foyle Cup webpage source code I scraped the data to provide the title, home team, away team and result which comprised of two digits with ‘- ‘, separator in a string for each group match. I also acquired the date for the match which may be used for future reference ie,if there is a gap between matches does it affect performance. All information was organized into independent list’s labelled, ‘title’, ‘Date’, ‘home’, ‘away’, ‘result’, as I felt slicing unknown data would be less complicated in a list format.

A function ‘groupdict’ was then developed to acquire the number of teams in the group, their name and establish the number of game days and matches per day based on the length of the ‘home’ list.

Game Days = number of teams in group – 1

Matches per day = number of teams in group / 2

A dictionary with the game day as key and the number of matches on that day was developed for each group but no team information or results were populated. It was decided that the ‘key’ identifier of the dictionary would prevent error when populating the data into the group matches and allow for greater accuracy when manipulating. At this stage we have a dictionary similar to the below as each group only had 4nr teams but a review of the development code shows its future proofed to allow for more:

{1: {'Match1': ['Home', 0, 0, 'Away'], 'Match2': ['Home', 0, 0, 'Away']},

2: {'Match1': ['Home', 0, 0, 'Away'], 'Match2': ['Home', 0, 0, 'Away']},

3: {'Match1': ['Home', 0, 0, 'Away'], 'Match2': ['Home', 0, 0, 'Away']}}

Using a developed function ‘groupdata’, and the 3 lists, ‘home’, ‘away’ and ‘results’ the code then populates the results data in the dictionaries created for each group using indexers as variables. Firstly, I removed all’- ‘separator from the ‘results’ to generate a new list named ‘numbers’ with only digits remaining and the results in order. As the result of some matches entered double figures for a single team this was slightly tricky but we found a work around using 2 for loops. All data was populated to give an output similar to below showing all of the matches played in a particular group:

{1: {'Match1': ['Raphoe FC', 1, 3, 'Oxford Utd'], 'Match2': ['Top of the Hill Celtic Colts', 3, 0, 'Don Bosco Colts']},

2: {'Match1': ['Don Bosco Colts', 0, 6, 'Raphoe FC'], 'Match2': ['Oxford Utd', 1, 2, 'Top of the Hill Celtic Colts']},

3: {'Match1': ['Top of the Hill Celtic Colts', 1, 2, 'Raphoe FC'], 'Match2': ['Oxford Utd', 4, 1, 'Don Bosco Colts']}}

We now have a dictionary containing a full list of the results for each particular game on every day of matches as our root data ‘Webpage.GroupMatches’ and a list containing all teams entered in the competition ‘Webpage.Teams’. The code is developed to allow for data incorporation from webpage scraping, excel or other format with minimal effort.

**Group section** **tables(‘generatetable’):** initial function utilizes the python toolkit ‘pandas’ to create 2 DataFrame ‘Games’ and ‘Table’. ‘Games’ contains the ‘Webpage.GroupMatches’ data and ‘Table has column headers listed but no data as yet. The dropna function of pandas ensures that no data within the ‘Games’ DataFrame has been duplicated during ‘Webpage.GroupMatches’ population and if so it gets removed.

Perhaps the most complex body of code within my assignment can be found under the function ‘tablematches’, which contains for loops within for loops, manipulation of lists/dictionaries, index/key referencing, searching for values, numpy to convert any erroneous values and multiple instances of slicing data etc.

We loop through the list ‘Webpage.Teams’ for each team and using the ‘isin()’ feature find them within each group table under the column ‘ Teams’. We acquire the teams index position in the group so as this can be used for slicing of the group values and to populate the match results data. As the ‘Table’ DataFrame initially has no data entered we use numpy to convert to erroneous values to zero which allows calculations to be complete. The index of the ‘Games’ allows us to determine if the team is home or away and the slice index of the result data to update all columns, 'Games', 'Wins', 'Draws', 'Loses', 'Goals For', 'Goals Against'. Finally, we check if the Table contains any null values and replace with zero.

The function ‘tablesort’, and numpy create a new column labelled ‘Goal Diff’ which comprises the Goals scored and conceded columns and declare the group title as the key for the dictionary. We now have the group tables populated via the ‘Webpage.GroupMatches’ data.

As the group tables type is ‘pandas.core.frame.DataDrame’ = pandas Dataframe and are generated with the main for loop which indexes through the url webpage path seen in the code we use the merge function of the pandas toolkit to compile the data into one table ‘Grouppos’ for use in the tournament section of the code. The data has been sorted using the following columns, ‘Position’, ‘Points’ and ‘GD’.

**Tournament section(‘competitions’):** Initially we create 2 number dictionaries, ‘competitions’ sections the teams in tabular form into their relevant competition based on the DataFrame ‘Grouppos’ and ‘compmatches’ comprises the team’s first game in the tournament section.

The function ‘createcomp’ takes the user supplied data ‘comps’ to generate the competitions template prior to population. ‘Comps’ is a dictionary with keys ‘name’ and ‘length’ which details the name of relevant tournament section and the number of teams to allocate. ‘createcomps’ populates ‘competitions’ dictionary with empty data based on ‘Comps’ as follows for the ‘Cup’ competition which has 16 teams and seen on cell ‘In[182]:

Comps = {'name': ['Cup', 'Plate', 'Rosebowl', 'Shield', 'Vase'],

'length': [16, 16, 8, 8, 8]}

{'Cup': ['Team1', 'Team2', 'Team3', 'Team4', 'Team5', 'Team6', 'Team7', 'Team8', 'Team9', 'Team10', 'Team11', 'Team12', 'Team13', 'Team14', 'Team15', 'Team16']}

Function ‘sortcomp’ and cell ‘In[185]’ loops through the ‘Comps data with reference to the ‘Grouppos’ table to assign teams to their relevant competition based on tournament section length and team position. The slice function is the prevalent tool to populate the ‘competitions’ data. As detailed within our problem statement during the 2022 Foyle Cup we were unable to comprehend which section our team would enter after 2 rounds of group matches. If the assignment had been completed at the time the function would have shown that we could lose our last game by 5 goals and still make our targeted ‘Shield’ competition with the add-on of a on the side of the draw were two teams had withdrew.

The final task for our development is to pair the teams in tournament matches based on their position in the ‘competitions’ table. As the ‘Cup’ competition pairings are previously determined based on group title as seen below an independent function ‘cup’ is required. The ‘cup’ function filters the ‘competitions’ table by the column ‘Group’ until a team matching the individual positions in ‘CupParing’ is found. When the team is found the index position is recorded and used as a reference for the ‘Teams’ column to allocate the name of the competitor to the tournament match. We can see from the ‘CupParings’ list that the 2 best 2nd place finishers within the group matches enter the ‘Cup’ tournament, ‘1B2', '2B2'. This inevitably leads to multiple instances of the ‘CupParing’ reference identifier so the ‘cup’ function then switches to the column ‘Position’ to ensure the team finished in 1st place during group matches. '1B2’ refers to first best 2nd place team and ‘2B2’ refers to second best 2nd place team. As the ‘competitions’ table has been sorted the program recognizes the ‘CupParing’ group identifier is not an alphabetical letter using isalpha() and slices the table from the bottom based on the reference 1 =-2, 2=-1.

CupPairing = ['A', '2B2', 'D', 'M', 'E', 'L', 'B', '1B2', 'G', 'J', 'C', 'N', 'F', 'K', 'H', 'I']

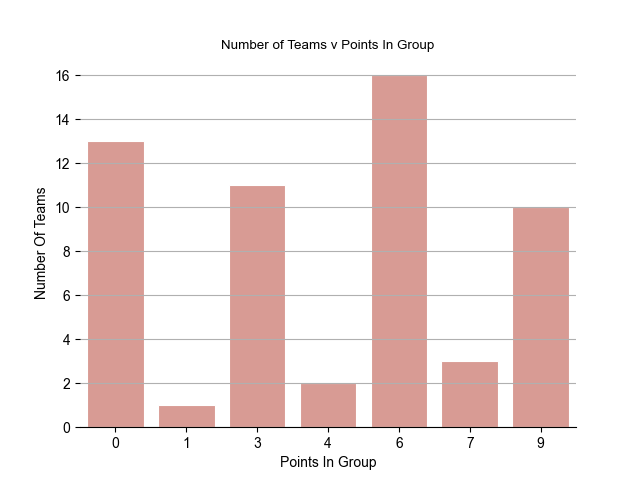
The ‘tournament’ function and cell In[186] simply loop through the remaining competitions and slice the relevant ‘competitions’ table data based on the length. The top team faces the bottom within the competition table so we simply use a pointer to reference until equal to half the length of the competition, thus the top and bottom pointers meet.

We now have a compiled list of first round matches for all tournament competitions as seen via cell In[194]. The remaining processes within our program display visuals relevant to our created data.

# Results

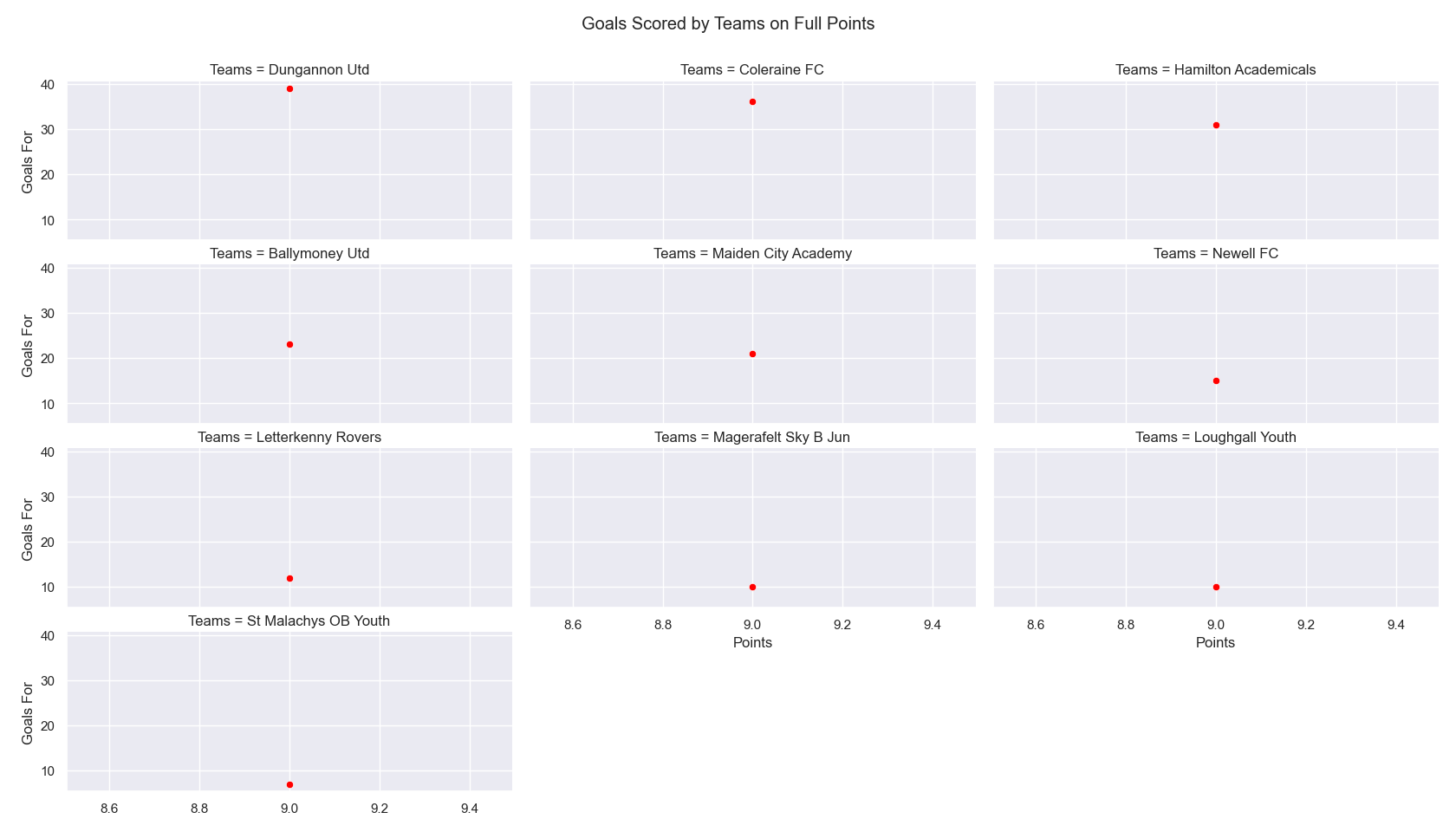
(Include the charts and describe them)

**Points In Group Matches**



The above chart best represented by a bar graph shows that 10 teams out of 14 groups finished on the maximum of 9 points with 13 teams finishing on zero points which derives a strong and weak team in each group. With 11 teams on 3 points and 16 on 6 points the graph shows each group followed a common trend for match results with very few draws.

**Cup Tournament Favorites**



As the previous graph showed each group table prominently follow the same results pattern we can summaries that the level of each team with respect to their group position is similar and use goal difference to attempt to establish the winner of the individual tournament sections. The above graph which has been sorted by goal different for the teams on max points shows Dungannon, Coleraine and Hamilton as the clear favorites for the cup competition. With the 2022 Foyle cup now complete we can compare the above prediction to the actual result which shows Coleraine and Hamilton finished 2nd and 3rd respectively with maiden city the eventual winner. Dungannon won the 2018 cup competition so prior to the tournament section would have been a firm favorite but only managed 5th out of a 16 team tournament which would be viewed as poor. In fairness they won their first match 12 – 0 against 3rd from bottom in the Cup table Raphoe and only lost to Hamilton after penalties, therefore to summaries the above is found to be accurate based on actual v predicted tournament result ie: machine learning.

I plan to add data to show the match day in relation to a team’s performance as I have the match results for each day which can be compared to points accumulated and goals scored/conceded. Potentially this could show a team improved or fatigued as the tournament progressed

As an add-on a distance travelled for each game could be referenced as some teams are from Derry where the tournament was held but in our case for example we travelled a round trip of roughly 86miles/day. Did daily travel to venue influence team performance.

Due to the timeframe these have not been included.

Further reference for machine learning will trend the teams over a period of time through multiple tournaments.

# Insights

(Point out at least 5 insights in bullet points)

* Had the information/code been developed pre-tournament we would have knew a cushion of 5 goals existed and conceding 2-3 additional goals in our final game to give us a greater chance of virtual getting a bye into the final of our section. As stated mid tournament we could comprehend the data to establish which section we would be placed.
* In the events all group stage matches follow a similar pattern the eventual winner for each tournament section can be surmised based on goal difference. All group followed the same pattern.
* The development of python code for this assignment will benefit Tummery in the event the club hosts our own competition.
* Group H was the most entertaining if looking for goals scored
* Group I followed by J and K were the most competitive based on the points distribution for the groups.

# References

(Include any references if required)

* Udemy course completed 2021
* Datacamp referenced for commands to manipulate data.
* stack overflow referenced for commands to manipulate data.