ASSIGNMENT 1- GATHERING, SCRAPING, MUNGING AND CLEANING OF DATA

ABSTRACT

We are working on a data set comprised of the top 5 English Premier League teams from the year 2018-2019. The dataset will have the 5 teams with their final standings of the season, their squads used for that season and the matches that they played against each other. The extraction of the data is done from three data sources which are, a website (web-scraping), an API and a raw csv file.

DATA GATHERED

1) API

The api consists of all match and team data from all the leagues across Europe. From that we pull out the English Premier League from which we pull out the data for the 5 teams that we desire.

The data extracted consists of team id, team name, date founded, venue. From the team data we pull out the first team squad data for each team. This player data includes id, name, position and nationality.

Importing Libraries

```
import requests
import pandas as pd
```

Importing dataset using API key

```
api_key = '88ed031015d040689e5dda65c9260458'
#authentication requires x-auth-token as header
header = {'X-Auth-Token': api_key}
#base_url
base_url = 'http://api.football-data.org/v2'
```

Selecting the top 5 teams from the dataset

```
url_competition = base_url + '/competitions/2021/teams' #url for english premier league
epl_teams = requests.get(url_competition,headers = header).json()
```

```
teams = (epl_teams['teams']) #only pulling out epl teams
selected_columns = ['id', 'name', 'founded', 'venue']
teams_df = pd.DataFrame(teams,columns = selected_columns) #extracting the selected columns
teams_df = teams_df.iloc[[0,2,4,5,9],:] #selecting top 5 teams for the season via the index
print (teams_df)
```

```
   id                 name  founded                    venue
0  57           Arsenal FC   1886.0          Emirates Stadium
2  61            Chelsea FC   1905.0           Stamford Bridge
4  64          Liverpool FC   1892.0                  Anfield
5  65    Manchester City FC   1880.0             Etihad Stadium
9  73  Tottenham Hotspur FC   1882.0  Tottenham Hotspur Stadium
```

Printing all the relevant data of the players in the 5 teams and adding team id to each player

```
url_teams = base_url + '/teams/'
players = pd.DataFrame()
for ids in teams_df['id']: # getting team ids to get players
    url_each_team = url_teams + str(ids)
    all_players = requests.get(url_each_team, headers = header).json()
    players_team = (all_players['squad'])
    selected_column_1 = ['id', 'name', 'position', 'nationality']
    players_df = pd.DataFrame(players_team,columns = selected_column_1) #extracting player details
    players_df['team_id'] = int(ids) #inserting column for team id
    players = pd.concat([players_df, players])
    print(url_each_team)
```

```
print (players)
```

```
        id             name    position    nationality  team_id
0     3355      Hugo Lloris  Goalkeeper         France       73
1     7991      Michel Vorm  Goalkeeper    Netherlands       73
2     7992   Alfie Whiteman  Goalkeeper        England       73
3     7993  Paulo Gazzaniga  Goalkeeper      Argentina       73
4   133229   Brandon Austin  Goalkeeper        England       73
..     ...              ...         ...            ...      ...
26    8412     Nicolas Pépé    Attacker  Côte d'Ivoire       57
27   61450       Martinelli    Attacker         Brazil       57
28   99813      Bukayo Saka    Attacker        England       57
29  131040  Folarin Balogun    Attacker        England       57
30   11619           Arteta        None          Spain       57

[170 rows x 5 columns]
```

Auditing and Cleaning of data

Changing id to player_id

```
#changing column name - id to player_id
players.columns = ['player_id', 'name', 'position', 'nationality', 'team_id']
```

```
players
```

	player_id	name	position	nationality	team_id
0	3355	Hugo Lloris	Goalkeeper	France	73
1	7991	Michel Vorm	Goalkeeper	Netherlands	73
2	7992	Alfie Whiteman	Goalkeeper	England	73
3	7993	Paulo Gazzaniga	Goalkeeper	Argentina	73
4	133229	Brandon Austin	Goalkeeper	England	73
...	...	...	...	...	...
26	8412	Nicolas Pépé	Attacker	Côte d'Ivoire	57
27	61450	Martinelli	Attacker	Brazil	57
28	99813	Bukayo Saka	Attacker	England	57
29	131040	Folarin Balogun	Attacker	England	57
30	11619	Arteta	None	Spain	57

170 rows × 5 columns

2) Raw Data

The raw data is comprised of a .csv file showcasing all matches from the English Premier League for the year 2018-2019. It shows the date, home team, away team, home goals scored, away goals scored, half time result, half time home goals, half time away goals and the referee for the game. From this data we only extract only the date, home team, away team, full time home goals, full time away goals, full time result, half time home goals, half time away goals, half time result.

Importing the essential libraries

```
import pandas as pd
```

Reading the data available in the CSV file and extracting the relevant information

```
#getting all the matches between the top 5 teams in the english premier league 2018-2019 season
match_pd = footcsv.loc[(footcsv['HomeTeam'].isin(['Arsenal', 'Man City', 'Tottenham', 'Chelsea', 'Liverpool'])) \
            & (footcsv['AwayTeam'].isin(['Arsenal', 'Man City', 'Tottenham', 'Chelsea', 'Liverpool']))]
match_pd
```

	HomeTeam	AwayTeam	FTHG	FTAG	FTR	HTHG	HTAG	HTR
7	Arsenal	Man City	0	2	A	0	1	A
11	Chelsea	Arsenal	3	2	H	2	2	D
45	Tottenham	Liverpool	1	2	A	0	1	A
61	Chelsea	Liverpool	1	1	D	1	0	H
78	Liverpool	Man City	0	0	D	0	0	D
99	Tottenham	Man City	0	1	A	0	1	A
100	Arsenal	Liverpool	1	1	D	0	0	D
124	Tottenham	Chelsea	3	1	H	2	0	H
137	Arsenal	Tottenham	4	2	H	1	2	A
154	Chelsea	Man City	2	0	H	1	0	H
193	Liverpool	Arsenal	5	1	H	4	1	H
209	Man City	Liverpool	2	1	H	1	0	H
220	Arsenal	Chelsea	2	0	H	2	0	H
248	Man City	Arsenal	3	1	H	2	1	H
258	Man City	Chelsea	6	0	H	4	0	H
274	Chelsea	Tottenham	2	0	H	0	0	D
283	Tottenham	Arsenal	1	1	D	0	1	A
312	Liverpool	Tottenham	2	1	H	1	0	H
333	Liverpool	Chelsea	2	0	H	0	0	D
338	Man City	Tottenham	1	0	H	1	0	H

Auditing and Cleaning of data

Shape of data frame and the column names to help select relevant columns

```
print(match_pd.shape)
```

```
(20, 23)
```

```
print(match_pd.columns)
```

```
Index(['Div', 'Date', 'HomeTeam', 'AwayTeam', 'FTHG', 'FTAG', 'FTR', 'HTHG',
       'HTAG', 'HTR', 'Referee', 'HS', 'AS', 'HST', 'AST', 'HF', 'AF', 'HC',
       'AC', 'HY', 'AY', 'HR', 'AR'],
      dtype='object')
```

Checking for missing values and general descriptive information

```
print(matches.info())
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 20 entries, 7 to 338
Data columns (total 8 columns):
HomeTeam    20 non-null object
AwayTeam    20 non-null object
FTHG        20 non-null int64
FTAG        20 non-null int64
FTR         20 non-null object
HTHG        20 non-null int64
HTAG        20 non-null int64
HTR         20 non-null object
dtypes: int64(4), object(4)
memory usage: 1.4+ KB
None
```

The relevant columns were extracted from the data frame and the shape tells us that there are 20 rows and the above counts exactly 20 non-null objects in each category, implying that there are no null values.

Adding match id, team id for home and away both

```
#adding match id and team id to help identify the team
matches.index.name = 'match_id' #turning row name into an index
matches.reset_index(inplace=True) #resetting index
```

```
#team_id_home column list #added mannually
team_id_home = pd.DataFrame([57, 61, 73, 61, 64, 73, 57, 73, 57, 61, 64, 65, 57, 65, 65, 61, 73, 64, 64, 65])
team_id_away = pd.DataFrame([65, 57, 64, 64, 65, 65, 64, 61, 73, 65, 57, 64, 61, 57, 61, 73, 57, 73, 61, 73])
matches = pd.concat([matches, team_id_home, team_id_away], axis=1)
```

---

	match_id	HomeTeam	AwayTeam	FTHG	FTAG	FTR	HTHG	HTAG	HTR	0	0
0	7	Arsenal	Man City	0	2	A	0	1	A	57	65
1	11	Chelsea	Arsenal	3	2	H	2	2	D	61	57
2	45	Tottenham	Liverpool	1	2	A	0	1	A	73	64
3	61	Chelsea	Liverpool	1	1	D	1	0	H	61	64
4	78	Liverpool	Man City	0	0	D	0	0	D	64	65
5	99	Tottenham	Man City	0	1	A	0	1	A	73	65
6	100	Arsenal	Liverpool	1	1	D	0	0	D	57	64
7	124	Tottenham	Chelsea	3	1	H	2	0	H	73	61
8	137	Arsenal	Tottenham	4	2	H	1	2	A	57	73
9	154	Chelsea	Man City	2	0	H	1	0	H	61	65
10	193	Liverpool	Arsenal	5	1	H	4	1	H	64	57
11	209	Man City	Liverpool	2	1	H	1	0	H	65	64
12	220	Arsenal	Chelsea	2	0	H	2	0	H	57	61
13	248	Man City	Arsenal	3	1	H	2	1	H	65	57
14	258	Man City	Chelsea	6	0	H	4	0	H	65	61
15	274	Chelsea	Tottenham	2	0	H	0	0	D	61	73
16	283	Tottenham	Arsenal	1	1	D	0	1	A	73	57
17	312	Liverpool	Tottenham	2	1	H	1	0	H	64	73
18	333	Liverpool	Chelsea	2	0	H	0	0	D	64	61
19	338	Man City	Tottenham	1	0	H	1	0	H	65	73

Renaming and reordering columns

```
#rename columns
matches.columns = ['match_id', 'home_team', 'away_team', 'FTHG', 'FTAG','FTR', 'HTHG', 'HTAG', 'HTR', \
                   'team_id_home', 'team_id_away']
#reorder columns
matches = matches[['match_id', 'team_id_home', 'home_team', 'team_id_away','away_team', 'FTHG', 'FTAG'\
                   ,'FTR', 'HTHG', 'HTAG', 'HTR']]
matches
```

	match_id	team_id_home	home_team	team_id_away	away_team	FTHG	FTAG	FTR	HTHG	HTAG	HTR
0	7	57	Arsenal	65	Man City	0	2	A	0	1	A
1	11	61	Chelsea	57	Arsenal	3	2	H	2	2	D
2	45	73	Tottenham	64	Liverpool	1	2	A	0	1	A
3	61	61	Chelsea	64	Liverpool	1	1	D	1	0	H
4	78	64	Liverpool	65	Man City	0	0	D	0	0	D
5	99	73	Tottenham	65	Man City	0	1	A	0	1	A
6	100	57	Arsenal	64	Liverpool	1	1	D	0	0	D
7	124	73	Tottenham	61	Chelsea	3	1	H	2	0	H
8	137	57	Arsenal	73	Tottenham	4	2	H	1	2	A
9	154	61	Chelsea	65	Man City	2	0	H	1	0	H
10	193	64	Liverpool	57	Arsenal	5	1	H	4	1	H
11	209	65	Man City	64	Liverpool	2	1	H	1	0	H
12	220	57	Arsenal	61	Chelsea	2	0	H	2	0	H
13	248	65	Man City	57	Arsenal	3	1	H	2	1	H
14	258	65	Man City	61	Chelsea	6	0	H	4	0	H
15	274	61	Chelsea	73	Tottenham	2	0	H	0	0	D
16	283	73	Tottenham	57	Arsenal	1	1	D	0	1	A
17	312	64	Liverpool	73	Tottenham	2	1	H	1	0	H
18	333	64	Liverpool	61	Chelsea	2	0	H	0	0	D
19	338	65	Man City	73	Tottenham	1	0	H	1	0	H

3) Web Scraping

We scraped the 2018-2019 English Premier League (EPL) Table from the website https://www.soccerstats.com/latest.asp?league=england_2019 to pull the top 5 teams from that

season. From the data that was available we chose the games won, lost, drawn and the final points for the season.

Defining the URL used for web-scraping

```
#url for football website
URL_football = "https://www.soccerstats.com/latest.asp?league=england_2019"
```

Extracting data from the website

```
def scrapTeamStats(URL):
    #getting data from website
    r = requests.get(URL)
    soup = BeautifulSoup(r.content, "html.parser")
    table = soup.find('table', attrs = {'id': 'btable'})
    for row in table.findAll('tr', attrs = {"class": "odd"}):
        for column in row.findAll('td'):
            #print(column)
            atag = column.findAll("a", attrs={'href': re.compile("^team.asp")})
            #print(atag)
            #if atag is not empty
            if atag != []:
                #append the text of anchor tag to team_name list
                team_name.append(atag[0].contents[0])
                #print(atag[0].contents[0])
            #find the text in <td>
            if column.string != None:
                #stripping of whitespace
                text = column.string.strip()
                #storing only integers and appending to wdl list
                if text.isdigit():
                    wdl.append(text)

scrapTeamStats(URL_football)
```

Note: The data extracted is not in the desired form and hence reshaping was required.

Reshaping the data in a 20X3 format

```
#print(wdl)
#reshaping into 20 x 3 array
wdl_df = pd.DataFrame(np.array(wdl).reshape(20,3), columns = list("WDL"))
team_name_df = pd.DataFrame(np.array(team_name).reshape(20,1), columns = ["Team"])
```

```
scraped_df = pd.concat([team_name_df.reset_index(drop=True), wdl_df], axis=1)
#print(scraped_df.iloc[0:5,])
scraped_df = scraped_df.iloc[0:5,] #extracting only the first five teams
```

```
scraped_df
```

	Team	W	D	L
0	Manchester City	32	2	4
1	Liverpool	30	7	1
2	Chelsea	21	9	8
3	Tottenham	23	2	13
4	Arsenal	21	7	10

Auditing and Cleaning of data

Adding team_id column

```
#we get the the order of the top 5 teams and the wins, draws and losses for each one.
#adding team id column
team_id = [65, 64, 61,73, 57]
team = pd.DataFrame(team_id, columns = ['team_id'])
team = pd.concat([team, scraped_df], axis=1)
team
```

	team_id	Team	W	D	L
0	65	Manchester City	32	2	4
1	64	Liverpool	30	7	1
2	61	Chelsea	21	9	8
3	73	Tottenham	23	2	13
4	57	Arsenal	21	7	10

CONCEPTUALISING OF DATA

Each table made should contain data such that meaningful insights can be conceptualized from it. While extracting the data from the various data sources, we created a table for every data source. Each table thus created is shown below:

From CSV File:

	match_id	team_id_home	home_team	team_id_away	away_team	FTHG	FTAG	FTR	HTHG	HTAG	HTR
0	7	57	Arsenal	65	Man City	0	2	A	0	1	A
1	11	61	Chelsea	57	Arsenal	3	2	H	2	2	D
2	45	73	Tottenham	64	Liverpool	1	2	A	0	1	A
3	61	61	Chelsea	64	Liverpool	1	1	D	1	0	H
4	78	64	Liverpool	65	Man City	0	0	D	0	0	D
5	99	73	Tottenham	65	Man City	0	1	A	0	1	A
6	100	57	Arsenal	64	Liverpool	1	1	D	0	0	D
7	124	73	Tottenham	61	Chelsea	3	1	H	2	0	H
8	137	57	Arsenal	73	Tottenham	4	2	H	1	2	A
9	154	61	Chelsea	65	Man City	2	0	H	1	0	H
10	193	64	Liverpool	57	Arsenal	5	1	H	4	1	H
11	209	65	Man City	64	Liverpool	2	1	H	1	0	H
12	220	57	Arsenal	61	Chelsea	2	0	H	2	0	H
13	248	65	Man City	57	Arsenal	3	1	H	2	1	H
14	258	65	Man City	61	Chelsea	6	0	H	4	0	H
15	274	61	Chelsea	73	Tottenham	2	0	H	0	0	D
16	283	73	Tottenham	57	Arsenal	1	1	D	0	1	A
17	312	64	Liverpool	73	Tottenham	2	1	H	1	0	H
18	333	64	Liverpool	61	Chelsea	2	0	H	0	0	D
19	338	65	Man City	73	Tottenham	1	0	H	1	0	H

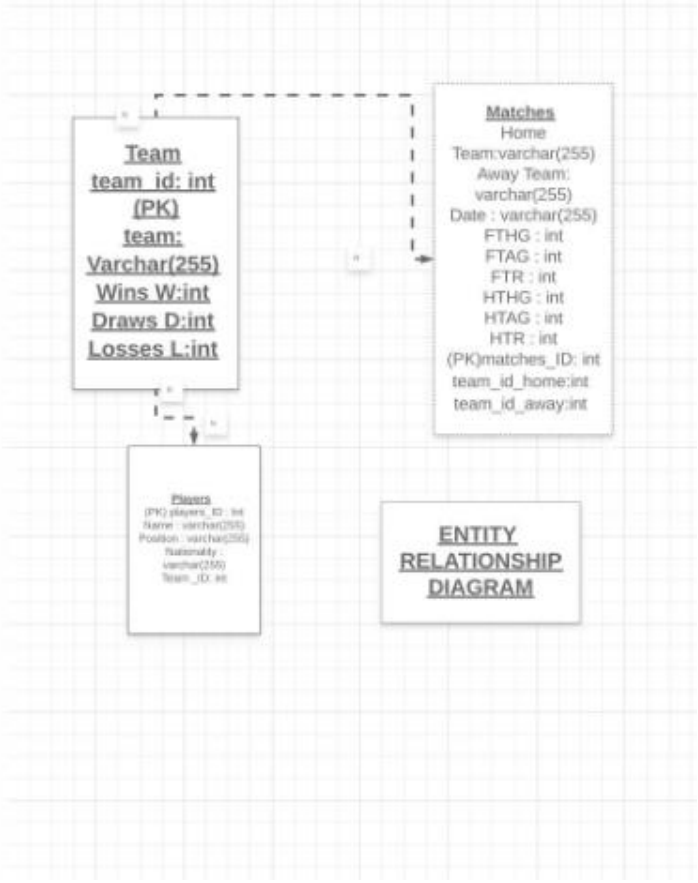
From Website:- (Web Scraping)

	team_id	Team	W	D	L
0	65	Manchester City	32	2	4
1	64	Liverpool	30	7	1
2	61	Chelsea	21	9	8
3	73	Tottenham	23	2	13
4	57	Arsenal	21	7	10

From API:-

	player_id	name	position	nationality	team_id
0	3355	Hugo Lloris	Goalkeeper	France	73
1	7991	Michel Vorm	Goalkeeper	Netherlands	73
2	7992	Alfie Whiteman	Goalkeeper	England	73
3	7993	Paulo Gazzaniga	Goalkeeper	Argentina	73
4	133229	Brandon Austin	Goalkeeper	England	73
...	...	...	...	...	...
26	8412	Nicolas Pépé	Attacker	Côte d'Ivoire	57
27	61450	Martinelli	Attacker	Brazil	57
28	99813	Bukayo Saka	Attacker	England	57
29	131040	Folarin Balogun	Attacker	England	57
30	11619	Arteta	None	Spain	57

ER MODEL



From the ER Model we can conclude that a conceptual schema can be derived which consists data from all the three data sources. The common data in this conceptual schema would be name from the Team Table, Home team, Away Team, Date, Full Time Home team goals (FTHG), Full Time Away team goals (FTAG), Half Time Home team goals (HTHG), Half Time Away team goals (HTAG), Results (R) and Half time result(R) from the Matches table and ID, Name, Position from the Players table.

ACCURACY

Through the course of this report, we believe the data is accurate and all unwanted null values were omitted from the extracted data. No junk values are present in the data hence confirming it to be clean.

REPORT

Code used: -

3 methods were employed for the same: -

1. Using the API

API Key: - 88ed031015d040689e5dda65c9260458
Requests to request data from the table
Pandas to create the data frame

2. From Raw Data

File Used: - C:\Users\Rohan\Desktop\NEU\Term2\DMDD\Soccer\2018-2019.csv
Pandas to create the data frame

3. From website (web scraping)

Website used: - https://www.soccerstats.com/latest.asp?league=england_2019
Pandas to create the data set
BeautifulSoup for scraping the data
Requests to access the data
Numpy for reshaping of data

CONCLUSION

From this assignment relevant data was extracted from various sources. Subsequently, this data was cleaned, null values were checked, and the data was reformatted to create a conceptual schema.

CONTRIBUTION

Own contribution : 50%

External : 30%

Professor : 20%

CITATIONS

http://localhost:8888/notebooks/Desktop/Data/Assignment1.ipynb#

https://coreyms.com/development/python/python-tutorial-web-scraping-beautiful-soup-requests

https://www.soccerstats.com/latest.asp?league=england_2019

https://pandas.pydata.org/pandas-docs/version/0.15/tutorials.html

GITHUB USERNAMES: -

https://github.com/Rohan-George5/Assignment-1-DMDD

https://github.com/yashmahansaria/Yash-Rohan-SEC08-ASSIGNMENT1