Gishnavi Kolli

Extracting and Interpreting the regular season Statistics of basketball players.

OVERVIEW

A project that gives you a better understanding of scraping data from websites and how to analyse them. Usage of various libraries as NumPy, Mat Plot, Pandas.
In the course of completing the project, you use the web scraping function, converting the extracted data into a pandas data Frame, and Storing the analysed data.

Problem Statement

Web scrape basketball statistics from Wikipedia of some of the greatest basketball players and export it as a CSV file format.

Introduction:

The most important first step in any data science project is to obtain good quality data. As a data science student, we will often have to use a variety of different methods to extract data sets. We might be using publically available data, data available via an API, data found in a database or in many cases a combination of these methods. [In this case we extract data from Wikipedia].

Data interpretation refers to the implementation of processes through which data is reviewed for the purpose of arriving at an informed conclusion. The interpretation of data assigns a meaning to the information analyzed and determines its signification and implications.

Software Requirements

1. Programming Language: Python

2. Environment: Jupyter Notebooks / Google Collab

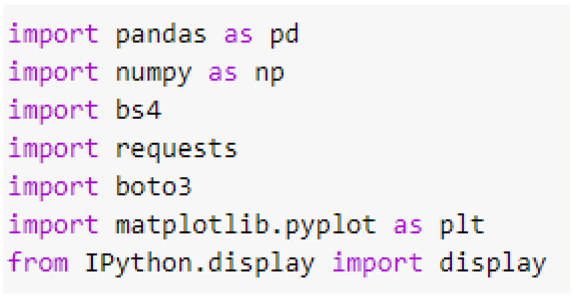
3. Database: CSV (export type)

4. Operation System: Windows XP or above

5. Libraries Used: Beautiful Soup, requests, Pandas, NumPy, boto3, Matplotlib, IPython.

6. Data Source: Wikipedia.

1. Open a New Notebook and import the required libraires



```python
import pandas as pd
import numpy as np
import bs4
import requests
import boto3
import matplotlib.pyplot as plt
from IPython.display import display
```

Introduction to used Packages

Bs4

Beautiful Soup is a Python library for pulling data out of HTML and XML files. It works with your favorite parser to provide idiomatic ways of navigating, searching, and modifying the parse tree. It commonly saves programmers hours or days of work.

[**Installation:** pip install beautifulsoup4

Usage: import bs4]

pandas

pandas python-based data analysis toolkit. It presents a diverse range of utilities, ranging from parsing multiple file formats to converting an entire data table into a NumPy matrix array. This makes pandas a trusted ally in data science and machine learning.

[**Installation:** pip install pandas

Usage: import pandas as pd]

- Numpy is required before installing pandas

Matplotlib

Matplotlib is a plotting library for the Python programming language and its numerical mathematics extension NumPy.

[**Installation:** pip install matplotlib

Usage: import matplotlib.pyplot as plt]

2. Reading the webpage

```python
def get_basketball_stats(link='https://en.wikipedia.org/wiki/Michael_Jordan'):
    response = requests.get(link)
    soup = bs4.BeautifulSoup(response.text, 'html.parser')
```

Here we get all the html code and parse it to take the necessary fields.

***get_basketball_stats** -. The variable stores the url of the website data we want to scrape.

- Then, we declare a variable that stores the requests from the link we provided.
- Then we use bs4 and parse the data into soup

3. Main Function Process

```python
table = soup.find(class_='wikitable sortable')
headers = table.tr
titles = headers.find_all('abbr')
data = {title['title']: [] for title in titles}

for row in table.find_all('tr')[1:]:
    for key, a in zip(data.keys(),row.find_all('td')[2:]):
        data[key].append(''.join(c for c in a.text if (c.isdigit() or c == '.')))

    Min = min([len(x) for x in data.values()])
    for key in data.keys():
        data[key] = list(map(lambda x: float(x), data[key][:Min]))
return data
```

*initialize a variable and assign to the class name of the main class that contains the data to be scrapped

*soup.find()---To find the class name and access it

*create an other variable to store all headers of the row element

*table—we assign main class that contains the data we need to scrape.

*create for loops to loop throughthe rows we iterate from index1 upto rest of the columns of the table *when iterating we check whether the data is a digit or '.' . If yes we append this data to data dictionary.

* we create a variable that stores the minimum value of the data dictionary

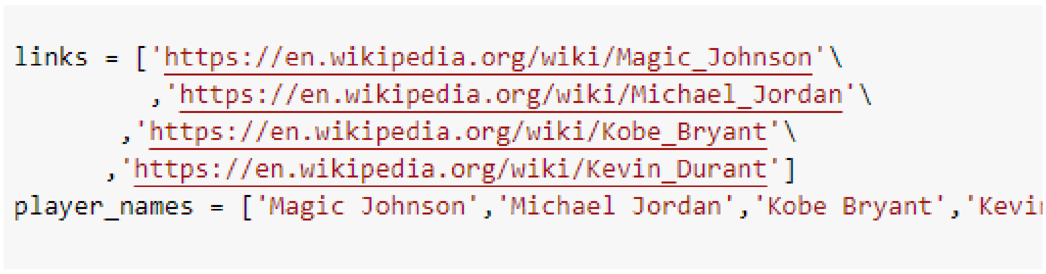
*then create another loop which converts the intiger values to float values upto minimum value

*header- stores all the headers of the row elements.

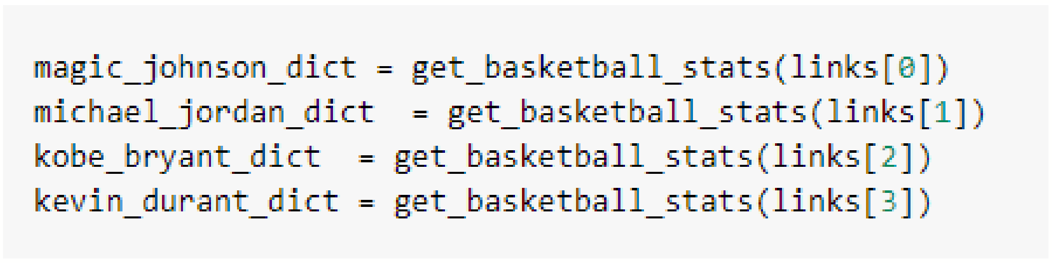
*find_all() We give the header element's class name as an attribute to this method, which helps us to find all the classes with that name

5.Declaring links and names of the personals to scrap the data

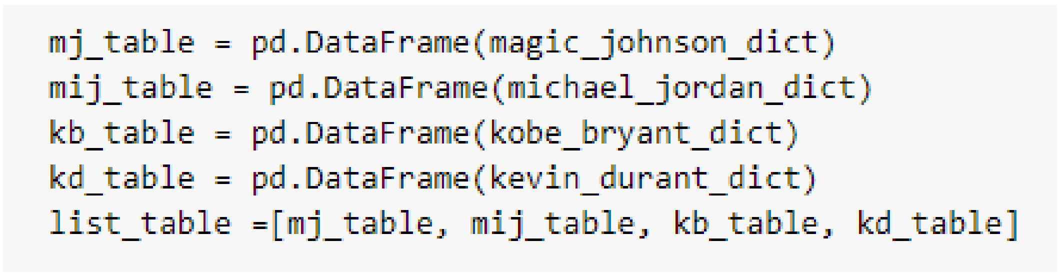
*links - We create a list that stores the links of the players
*player_names -We create another list that stores the names of the players.
*get_basketball_stats-takes the element of the list
*store this in respective variables of the player
*convert the data into data frames and print the output



```
links = ['https://en.wikipedia.org/wiki/Magic_Johnson'\
         ,'https://en.wikipedia.org/wiki/Michael_Jordan'\
       ,'https://en.wikipedia.org/wiki/Kobe_Bryant'\
      ,'https://en.wikipedia.org/wiki/Kevin_Durant']
player_names = ['Magic Johnson','Michael Jordan','Kobe Bryant','Kevi
```

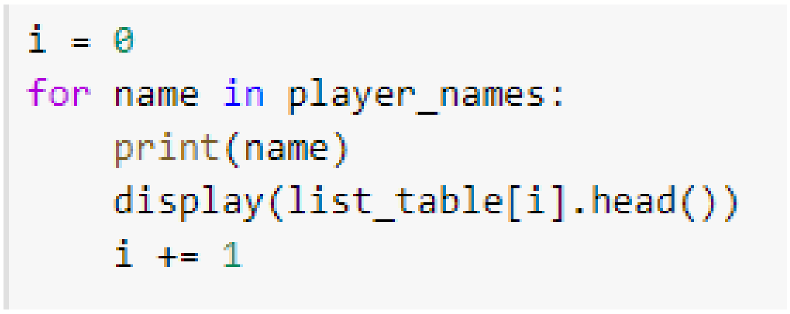


```
magic_johnson_dict = get_basketball_stats(links[0])
michael_jordan_dict  = get_basketball_stats(links[1])
kobe_bryant_dict   = get_basketball_stats(links[2])
kevin_durant_dict = get_basketball_stats(links[3])
```



```
mj_table = pd.DataFrame(magic_johnson_dict)
mij_table = pd.DataFrame(michael_jordan_dict)
kb_table = pd.DataFrame(kobe_bryant_dict)
kd_table = pd.DataFrame(kevin_durant_dict)
list_table =[mj_table, mij_table, kb_table, kd_table]
```

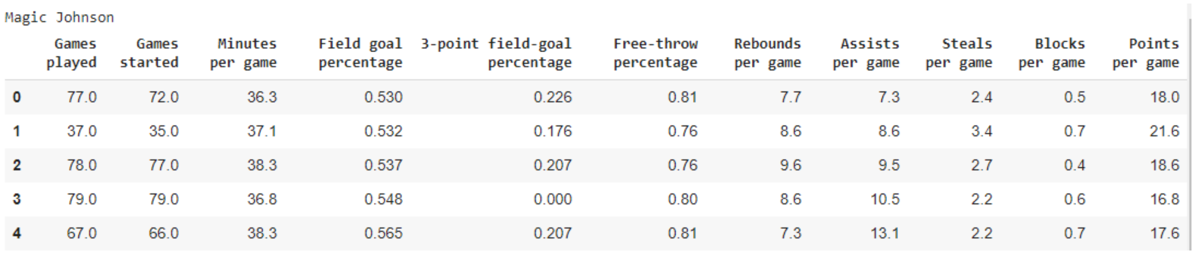
***get_basketball_stats** - We store this in variables of every player.
By using DataFrame we store the data in respective variables.



```
i = 0
for name in player_names:
    print(name)
    display(list_table[i].head())
    i += 1
```

Output:

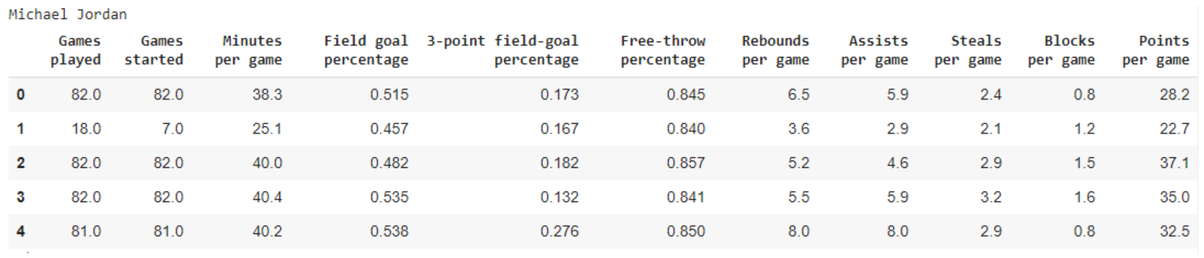
Magic Johnson



Magic Johnson

	Games played	Games started	Minutes per game	Field goal percentage	3-point field-goal percentage	Free-throw percentage	Rebounds per game	Assists per game	Steals per game	Blocks per game	Points per game
0	77.0	72.0	36.3	0.530	0.226	0.81	7.7	7.3	2.4	0.5	18.0
1	37.0	35.0	37.1	0.532	0.176	0.76	8.6	8.6	3.4	0.7	21.6
2	78.0	77.0	38.3	0.537	0.207	0.76	9.6	9.5	2.7	0.4	18.6
3	79.0	79.0	36.8	0.548	0.000	0.80	8.6	10.5	2.2	0.6	16.8
4	67.0	66.0	38.3	0.565	0.207	0.81	7.3	13.1	2.2	0.7	17.6

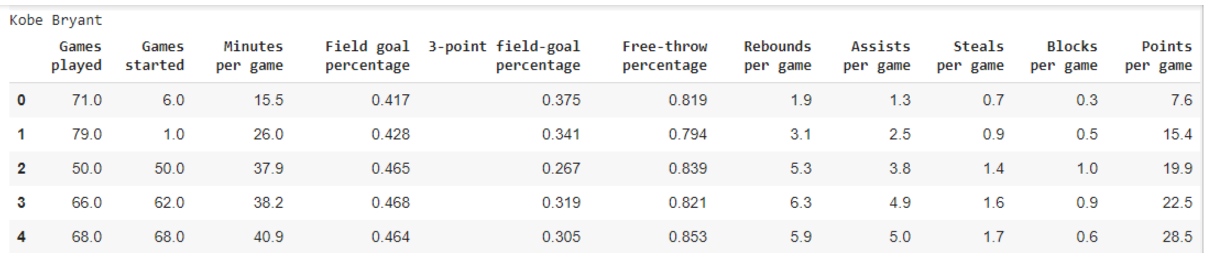
Michael Jordan



Michael Jordan

	Games played	Games started	Minutes per game	Field goal percentage	3-point field-goal percentage	Free-throw percentage	Rebounds per game	Assists per game	Steals per game	Blocks per game	Points per game
0	82.0	82.0	38.3	0.515	0.173	0.845	6.5	5.9	2.4	0.8	28.2
1	18.0	7.0	25.1	0.457	0.167	0.840	3.6	2.9	2.1	1.2	22.7
2	82.0	82.0	40.0	0.482	0.182	0.857	5.2	4.6	2.9	1.5	37.1
3	82.0	82.0	40.4	0.535	0.132	0.841	5.5	5.9	3.2	1.6	35.0
4	81.0	81.0	40.2	0.538	0.276	0.850	8.0	8.0	2.9	0.8	32.5

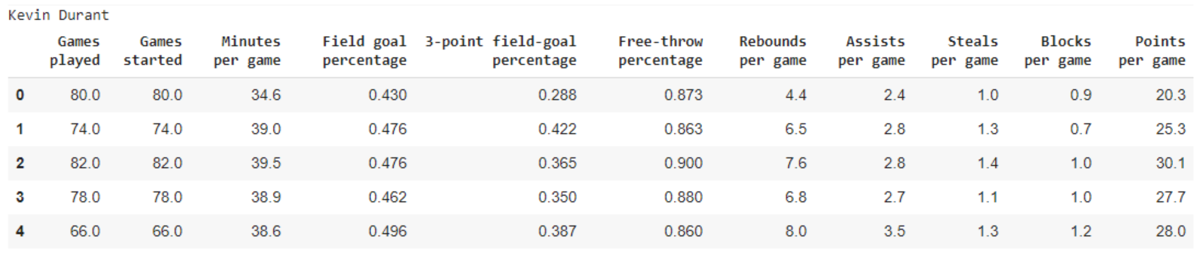
Kobe Bryant



Kobe Bryant

	Games played	Games started	Minutes per game	Field goal percentage	3-point field-goal percentage	Free-throw percentage	Rebounds per game	Assists per game	Steals per game	Blocks per game	Points per game
0	71.0	6.0	15.5	0.417	0.375	0.819	1.9	1.3	0.7	0.3	7.6
1	79.0	1.0	26.0	0.428	0.341	0.794	3.1	2.5	0.9	0.5	15.4
2	50.0	50.0	37.9	0.465	0.267	0.839	5.3	3.8	1.4	1.0	19.9
3	66.0	62.0	38.2	0.468	0.319	0.821	6.3	4.9	1.6	0.9	22.5
4	68.0	68.0	40.9	0.464	0.305	0.853	5.9	5.0	1.7	0.6	28.5

Kevin Durant



Kevin Durant

	Games played	Games started	Minutes per game	Field goal percentage	3-point field-goal percentage	Free-throw percentage	Rebounds per game	Assists per game	Steals per game	Blocks per game	Points per game
0	80.0	80.0	34.6	0.430	0.288	0.873	4.4	2.4	1.0	0.9	20.3
1	74.0	74.0	39.0	0.476	0.422	0.863	6.5	2.8	1.3	0.7	25.3
2	82.0	82.0	39.5	0.476	0.365	0.900	7.6	2.8	1.4	1.0	30.1
3	78.0	78.0	38.9	0.462	0.350	0.880	6.8	2.7	1.1	1.0	27.7
4	66.0	66.0	38.6	0.496	0.387	0.860	8.0	3.5	1.3	1.2	28.0

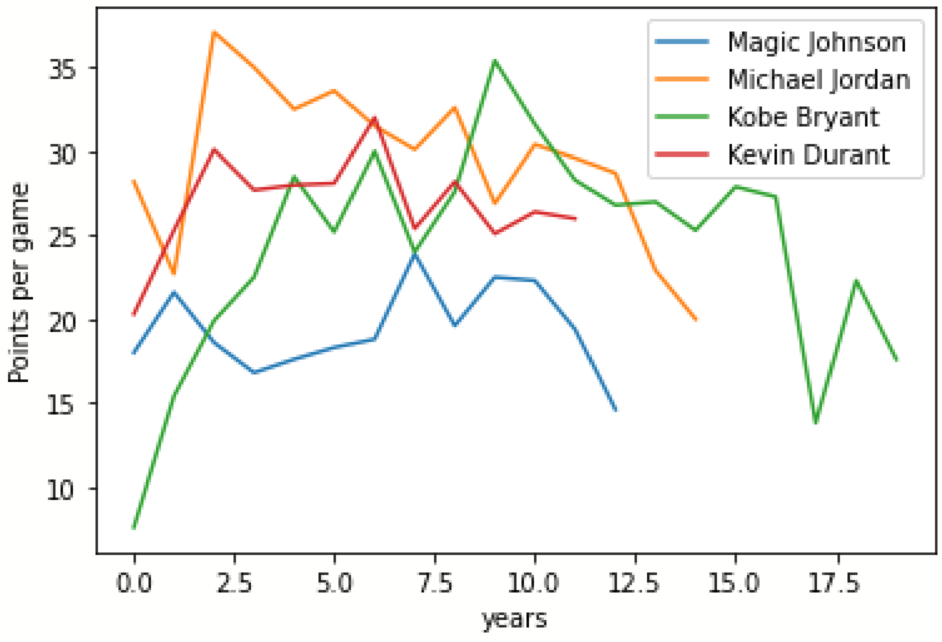
6.Making a plot using matplotlib

Plot function accesses the data from list_tables list and the player name gets assigned to the player names

-plt.xlabel() and plt.ylabel() are used to give the names for x and y axes.

```
j = 0
for name in player_names:
    plt.plot(list_table[j][['Points per game']],label=name)
    plt.legend()
    plt.xlabel('years')
    plt.ylabel('Points per game')
    j += 1
```

Output:



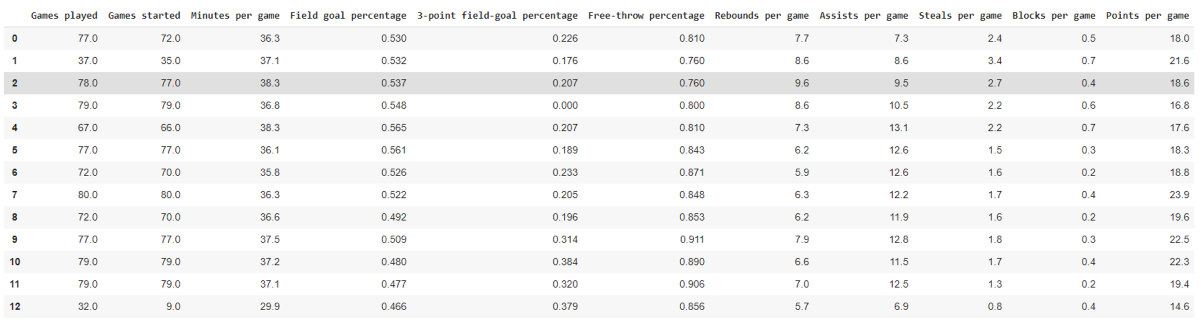
7.Storing the Player Statistics in Object Storage

```
csv_name = 'nba.csv'
mj_table.to_csv(csv_name)
```

We convert the produced data frames into a csv file “nba.csv”

```
mj_table
```

Mj_table is used to prints the table



	Games played	Games started	Minutes per game	Field goal percentage	3-point field-goal percentage	Free-throw percentage	Rebounds per game	Assists per game	Steals per game	Blocks per game	Points per game
0	77.0	72.0	36.3	0.530	0.226	0.810	7.7	7.3	2.4	0.5	18.0
1	37.0	35.0	37.1	0.532	0.176	0.760	8.6	8.6	3.4	0.7	21.6
2	78.0	77.0	38.3	0.537	0.207	0.760	9.6	9.5	2.7	0.4	18.6
3	79.0	79.0	36.8	0.548	0.000	0.800	8.6	10.5	2.2	0.6	16.8
4	67.0	66.0	38.3	0.565	0.207	0.810	7.3	13.1	2.2	0.7	17.6
5	77.0	77.0	36.1	0.561	0.189	0.843	6.2	12.6	1.5	0.3	18.3
6	72.0	70.0	35.8	0.526	0.233	0.871	5.9	12.6	1.6	0.2	18.8
7	80.0	80.0	36.3	0.522	0.205	0.848	6.3	12.2	1.7	0.4	23.9
8	72.0	70.0	36.6	0.492	0.196	0.853	6.2	11.9	1.6	0.2	19.6
9	77.0	77.0	37.5	0.509	0.314	0.911	7.9	12.8	1.8	0.3	22.5
10	79.0	79.0	37.2	0.480	0.384	0.890	6.6	11.5	1.7	0.4	22.3
11	79.0	79.0	37.1	0.477	0.320	0.906	7.0	12.5	1.3	0.2	19.4
12	32.0	9.0	29.9	0.466	0.379	0.856	5.7	6.9	0.8	0.4	14.6

Conclusion

Therefore, we have successfully scraped the Data of basketball statistics from Wikipedia of a few famous personalities.