CS2006 Python Practical 2

Dataset gathered from: https://www.kaggle.com/artimous/complete-fifa-2017-player-dataset-global (https://www.kaggle.com/artimous/complete-fifa-2017-player-dataset-global) The Dataset is the stats for each player and 50+ attricutes straight out of the video game FIFA 2017.

The first requirment is to refine the data and below the many libraries needed to refine and perform analysis on the data are imported.

In [1]:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import matplotlib.patches as patch
from ipywidgets import *
%matplotlib inline
import matplotlib.pyplot as plt
from mpl_toolkits.mplot3d.axes3d import Axes3D, get_test_data
from mpl_toolkits.mplot3d.art3d import Poly3DCollection
from matplotlib import cm
from operator import itemgetter

import pandas as pd
import numpy as np
import plotly
import plotly.graph_objs as go
import plotly.plotly as py

from ipywidgets import widgets
from IPython.display import display
from plotly.graph_objs import *
from plotly.widgets import GraphWidget
from plotly.offline import download_plotlyjs, init_notebook_mode, plot,
iplot
plotly.offline.init_notebook_mode(connected=True)

import cufflinks as cf
init_notebook_mode(connected=True)
```

```
/cs/home/ea50/.local/lib/python3.5/site-packages/IPython/html.py:14: Shim
Warning:

The `IPython.html` package has been deprecated since IPython 4.0. You sho
uld import from `notebook` instead. `IPython.html.widgets` has moved to `
ipywidgets`.

/cs/home/ea50/.local/lib/python3.5/site-packages/IPython/utils/traitlets.
py:5: UserWarning:

IPython.utils.traitlets has moved to a top-level traitlets package.
```

Read in the csv file

In [2]:

```
low_memory = False
df = pd.read_csv("../data/fifa_data.csv")
```

Check the types of each column in the CSV file

In [3]:

```
df.dtypes
```

Out[3]:

```
Name                   object
Nationality            object
National_Position      object
National_Kit          float64
Club                   object
Club_Position          object
Club_Kit              float64
Club_Joining           object
Contract_Expiry       float64
Rating                  int64
Height                 object
Weight                 object
Preffered_Foot         object
Birth_Date             object
Age                     int64
Preffered_Position     object
Work_Rate              object
Weak_foot               int64
Skill_Moves             int64
Ball_Control            int64
Dribbling               int64
Marking                 int64
Sliding_Tackle          int64
Standing_Tackle         int64
Aggression              int64
Reactions               int64
Attacking_Position      int64
Interceptions           int64
Vision                  int64
Composure               int64
Crossing                int64
Short_Pass              int64
Long_Pass               int64
Acceleration            int64
Speed                   int64
Stamina                 int64
Strength                int64
Balance                 int64
Agility                 int64
Jumping                 int64
Heading                 int64
Shot_Power              int64
Finishing               int64
Long_Shots              int64
Curve                   int64
Freekick_Accuracy       int64
Penalties               int64
Volleys                 int64
GK_Positioning          int64
GK_Diving               int64
GK_Kicking              int64
GK_Handling             int64
GK_Reflexes             int64
dtype: object
```

To begin really refining the data, the first thing done is remove any column with an empty value in it (NaN). This significanly reduces the size of the dataset in this particular case because the way data is provded by EA regarding players and their specefic stats.

In [4]:

```
refinedFifa = df.copy()
refinedFifa = refinedFifa.dropna()
pd.options.display.float_format = '{:,.0f}'.format
```

Below duplicate rows are removed from the dataset.

```
In [5]: refinedFifa = refinedFifa.drop_duplicates()
```

Below the length of the new refined dataset is given below. As you can see it is a major pretty drop from over 17,000 to just over 1000.

```
In [6]: len(refinedFifa)

Out[6]: 1075
```

```
In [7]: len(df)

Out[7]: 17588
```

Here arrays are defined as the only possible options for specefic columns and then there are nested for loops to check each value in the dataset matches one of the values in the predefined array. If the value is not in the array then then that row gets marked for removal and at the end of this block of code that there is loop that removes all of the rows that were marked for removal because they had invalid input in them.

```
In [8]: nationalPositions = ["CAM", "CB", "CDM", "CM", "GK", "LAM", "LB", "LCB",
"LCM", "LDM", "LF", "LM", "LS", "LW", "LWB", "RAM", "RB", "RCB", "RCM",
"RDM", "RF", "RM", "RS", "RW", "RWB", "ST", "Sub"]
clubPositions = ["CAM", "CB", "CDM", "CF", "CM", "GK", "LAM", "LB", "LCB
", "LCM", "LDM", "LF", "LM", "LS", "LW", "LW", "RAM", "RB", "RCB", "RCM"
, "RDM", "RF", "RM" "RS", "RW", "RWB", "Res", "ST", "Sub"]
contractDate = [2017, 2018, 2019, 2020, 2021, 2022, 2023]
prefferedFoot = ["Left", "Right"]
array100 = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 1
8, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 3
6, 37, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47,
            48, 49, 50, 51, 52, 53, 54, 55, 56, 57, 58, 59, 60, 61, 62,
63, 64, 65, 66, 67, 68, 69, 70, 71, 72, 73, 74, 75, 76, 77, 78, 79, 80,
81, 82, 83, 84, 85, 86, 87, 88, 89,
            90, 91, 92, 93, 94, 95, 96, 97, 98, 99, 100]

rowsToRemove = []
rCounter = 0
print(len(rowsToRemove))
natPos = refinedFifa['National_Position']
for natPoCounter in natPos:
    flag = 1
    for nationalPositionCounter in nationalPositions:
                if nationalPositionCounter == natPoCounter:
                    flag = 0
    if flag == 1:
        if rCounter not in rowsToRemove:
            rowsToRemove.append(rCounter)
    rCounter = rCounter + 1
print(len(rowsToRemove))
rCounter = 0

clubPos = refinedFifa['Club_Position']
for clubPosCounter in clubPos:
    flag = 1
    for clubPositionCounter in clubPositions:
                if clubPositionCounter == clubPosCounter:
                    flag = 0
    if flag == 1:
        if rCounter not in rowsToRemove:
            rowsToRemove.append(rCounter)
    rCounter = rCounter + 1
print(len(rowsToRemove))
rCounter = 0

conDate = refinedFifa['Contract_Expiry']
for contractCounter in conDate:
    flag = 1
    for conDateCounter in contractDate:
                if conDateCounter == contractCounter:
                    flag = 0
    if flag == 1:
        if rCounter not in rowsToRemove:
            rowsToRemove.append(rCounter)
    rCounter = rCounter + 1
print(len(rowsToRemove))
rCounter = 0

prefFoot = refinedFifa['Preffered_Foot']
for prefFootCounter in prefFoot:
    flag = 1
    for footCounter in prefferedFoot:
                if footCounter == prefFootCounter:
                    flag = 0
    if flag == 1:
        if rCounter not in rowsToRemove:
            rowsToRemove.append(rCounter)
    rCounter = rCounter + 1
```

```
0
0
47
47
47
1075
47
1028
```

Here the national position data is shown with how many of each position is in the dataset. The same goes for the next few code blocks for Nationality, Club, Club Position, Contract Expiry, Rating, Height, Weight, Preffered Foot, Age, Work Rate, and Preffered Position. This is some of the data that goes on to be plotted using matplotlib.

```
In [9]:  national_position = refinedFifa.groupby("National_Position")
         national_position.size()
```

```
Out[9]: National_Position
        CAM     18
        CB       8
        CDM      9
        CM       9
        GK      47
        LAM      4
        LB      37
        LCB     48
        LCM     25
        LDM     19
        LF       3
        LM      31
        LS      17
        LW       6
        LWB      3
        RAM      2
        RB      37
        RCB     46
        RCM     23
        RDM     18
        RF       2
        RM      33
        RS      15
        RW       6
        RWB      4
        ST      26
        Sub    532
        dtype: int64
```

```
In [10]: nationality = refinedFifa.groupby("Nationality")
         nationality.size()
```

```
Out[10]: Nationality
         Argentina              23
         Australia              23
         Austria                20
         Belgium                23
         Bolivia                21
         Brazil                 23
         Bulgaria               21
         Cameroon               23
         Canada                 22
         Chile                  22
         China PR               23
         Colombia               22
         Czech Republic         23
         Denmark                21
         Ecuador                22
         Egypt                  22
         England                20
         Finland                19
         France                 22
         Germany                23
         Greece                 23
         Hungary                23
         India                  24
         Italy                  23
         Ivory Coast            23
         Mexico                 21
         Netherlands            19
         Northern Ireland       22
         Norway                 23
         Paraguay               22
         Peru                   22
         Poland                 23
         Portugal               19
         Republic of Ireland    23
         Romania                20
         Russia                 20
         Scotland               22
         Slovenia               23
         South Africa           24
         Spain                  22
         Sweden                 22
         Switzerland            22
         Turkey                 22
         United States          22
         Uruguay                20
         Venezuela              20
         Wales                  21
         dtype: int64
```

In [11]:

```
club = refinedFifa.groupby("Club")
club.size()
```

Out[11]:

```
Club
1. FC Köln              1
1. FC Nürnberg          2
1. FSV Mainz 05         2
1860 München            1
1899 Hoffenheim         4
AC Ajaccio              1
ADO Den Haag            1
AIK                     3
AS Monaco               7
AS Nancy                1
AS Saint-Étienne        2
AZ                      1
Aalborg BK              1
Aarhus GF               1
Adelaide United         1
Ajax                    6
Akhisarspor             1
Al Ahli                 1
Al-Ettifaq              1
América                 2
Angers SCO              2
Antalyaspor             1
Arouca                  1
Arsenal                11
Arsenal Tula            1
Aston Villa             5
Atalanta                3
Atl. Nacional           3
Atlas                   2
Atlético Madrid         7
                       ..
Terek Grozny            4
Tigres                  6
Toluca                  2
Tondela                 1
Torino                  4
Toronto FC              2
Toulouse FC             4
Trabzonspor             3
U.N.A.M.                3
Udinese                 4
Uni. Católica           2
Uni. de Chile           3
V. Guimarães            1
Valencia CF             4
Veracruz                2
VfB Stuttgart           2
VfL Wolfsburg           5
Viborg FF               1
Villarreal CF           3
Walsall                 1
Watford                 4
Werder Bremen           5
West Brom               8
West Ham                6
Whitecaps FC            2
Wigan Athletic          1
Wisła Kraków            1
Wolves                  1
Yokohama F. Marinos     1
Zenit                   3
dtype: int64
```

```
In [12]: club_position = refinedFifa.groupby("Club_Position")
         club_position.size()
```

```
Out[12]: Club_Position
         CAM     31
         CB       6
         CDM     12
         CF       1
         CM       7
         GK      80
         LAM      3
         LB      38
         LCB     52
         LCM     31
         LDM     14
         LF       2
         LM      36
         LS      12
         LW      16
         RAM      3
         RB      44
         RCB     49
         RCM     23
         RDM     16
         RF       2
         RW      18
         RWB      4
         Res    231
         ST      41
         Sub    256
         dtype: int64
```

```
In [13]: contract_expiry = refinedFifa.groupby("Contract_Expiry")
         contract_expiry.size()
```

```
Out[13]: Contract_Expiry
         2,017     96
         2,018    214
         2,019    230
         2,020    231
         2,021    137
         2,022     56
         2,023     64
         dtype: int64
```

```
In [14]: overall_rating = refinedFifa.groupby("Rating")
         overall_rating.size()
```

```
Out[14]: Rating
         52     2
         53     2
         54     2
         55     3
         56     1
         57     1
         58     8
         59    10
         60    11
         61     8
         62    11
         63    17
         64    13
         65    23
         66    25
         67    36
         68    25
         69    40
         70    45
         71    54
         72    54
         73    66
         74    67
         75    59
         76    62
         77    55
         78    48
         79    46
         80    42
         81    33
         82    37
         83    46
         84    16
         85    12
         86    16
         87     6
         88    10
         89     8
         90     3
         92     3
         93     1
         94     1
         dtype: int64
```

In [15]:

```
height = refinedFifa.groupby("Height")
height.size()
```

Out[15]:

```
Height
162 cm     1
163 cm     1
164 cm     2
165 cm     4
166 cm     2
167 cm     3
168 cm    11
169 cm    12
170 cm    30
171 cm    16
172 cm    16
173 cm    29
174 cm    26
175 cm    51
176 cm    34
177 cm    21
178 cm    56
179 cm    39
180 cm    74
181 cm    29
182 cm    43
183 cm    69
184 cm    54
185 cm    60
186 cm    39
187 cm    45
188 cm    65
189 cm    36
190 cm    46
191 cm    22
192 cm    24
193 cm    29
194 cm     7
195 cm     9
196 cm    13
197 cm     3
198 cm     3
199 cm     2
201 cm     1
203 cm     1
dtype: int64
```

In [16]:

```
weight = refinedFifa.groupby("Weight")
weight.size()
```

Out[16]:

```
Weight
58 kg     2
59 kg     3
60 kg     6
61 kg     4
62 kg     7
63 kg     5
64 kg    11
65 kg    11
66 kg    19
67 kg    24
68 kg    36
69 kg    19
70 kg    78
71 kg    35
72 kg    45
73 kg    43
74 kg    57
75 kg    68
76 kg    53
77 kg    44
78 kg    55
79 kg    39
80 kg    43
81 kg    32
82 kg    43
83 kg    34
84 kg    35
85 kg    42
86 kg    29
87 kg    16
88 kg    22
89 kg    10
90 kg    15
91 kg    15
92 kg    11
93 kg     6
94 kg     1
95 kg     5
96 kg     2
97 kg     2
98 kg     1
dtype: int64
```

In [17]:

```
foot = refinedFifa.groupby("Preffered_Foot")
foot.size()
```

Out[17]:

```
Preffered_Foot
Left     222
Right    806
dtype: int64
```

In [18]:

```
age = refinedFifa.groupby("Age")
age.size()
```

Out[18]:

```
Age
17      1
18      4
19     16
20     16
21     29
22     62
23     71
24     93
25     96
26     89
27    105
28     89
29     86
30     87
31     61
32     50
33     28
34     14
35     11
36      3
37     10
38      3
39      3
44      1
dtype: int64
```

In [19]:

```
work_rate = refinedFifa.groupby("Work_Rate")
work_rate.size()
```

Out[19]:

```
Work_Rate
High / High        113
High / Low          48
High / Medium      220
Low / High          25
Low / Low            1
Low / Medium        21
Medium / High      123
Medium / Low        37
Medium / Medium    440
dtype: int64
```

In [20]:

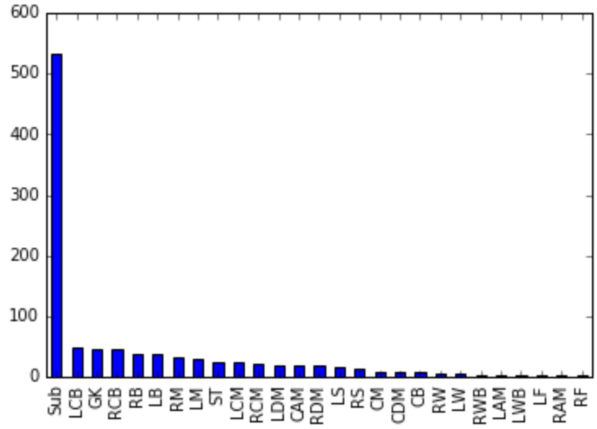
```
pref_position = refinedFifa.groupby("Preffered_Position")
pref_position.size()
```

Out[20]:

```
Preffered_Position
CAM           11
CAM/CF         4
CAM/CM        11
CAM/LM        12
CAM/LM/CM      1
CAM/LM/RM      2
CAM/LW         4
CAM/RM         6
CAM/RM/LM      1
CAM/RW         5
CAM/ST         3
CB           145
CB/CDM        13
CB/CM          2
CB/LB         15
CB/RB         18
CDM           27
CDM/CAM        3
CDM/CB         7
CDM/CM        46
CDM/LM         1
CDM/RB         3
CDM/RM         1
CF/CAM         1
CF/CAM/ST      1
CF/RM          1
CM            19
CM/CAM        16
CM/CB          1
CM/CDM        53
            ...
RM             5
RM/CAM         8
RM/CF          1
RM/CM          2
RM/CM/CDM      1
RM/LM         13
RM/LW          1
RM/LW/LM       1
RM/RB          1
RM/RW          2
RM/ST          3
RW             5
RW/CAM         3
RW/CF          1
RW/CM/LW       1
RW/LM          1
RW/LW          6
RW/RB          2
RW/RM          2
RW/ST          2
RWB/RM         1
ST            88
ST/CAM         8
ST/CF          4
ST/LM         16
ST/LW          8
ST/RM         11
ST/RM/RW       1
ST/RW          4
ST/RW/RM       1
dtype: int64
```

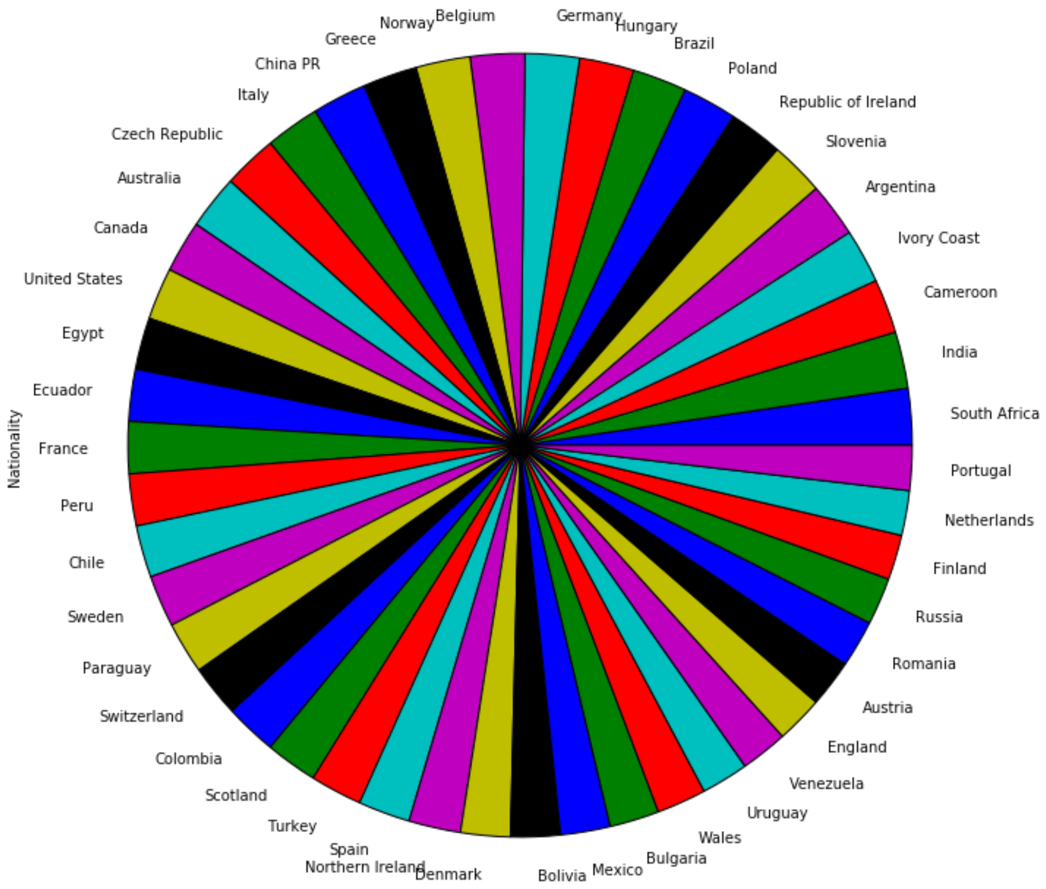
Below is a bar graph of the National Position data and as you can see the majority of the players in the FIFA 2017 game are subs for their respective national teams.

In [21]:
```
refinedFifa['National_Position'].value_counts().plot(kind="bar")
plt.show()
```



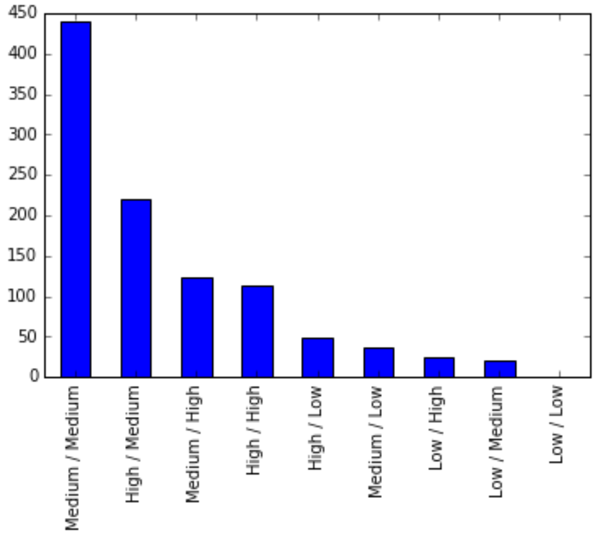
Below is a pie chart of the nationalities represented in the game and as you can see it is a pretty even distrbution of players from each country.

```
In [22]: refinedFifa['Nationality'].value_counts().plot(kind="pie", figsize = (12
         ,12))
         plt.show()
```



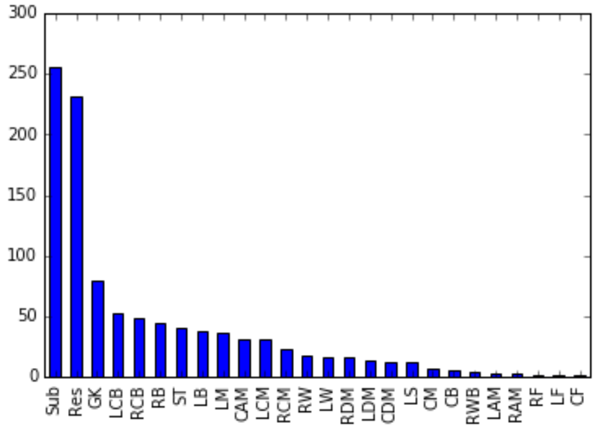
Below is a bar char showing the distribution of work rates among players. The work rate stat in FIFA 17 describes a players leaning towards offense of defense and as such high/medium means high offense skills/medium deffense skills.

```
In [23]: refinedFifa['Work_Rate'].value_counts().plot(kind="bar")
         plt.show()
```



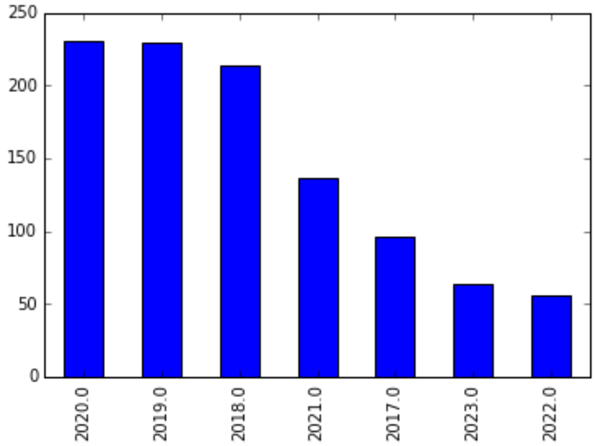
Below is the distribution of club positions among players and as is simmilar to the national position data the majority of players in FIFA are substitutes.

```
In [24]: refinedFifa['Club_Position'].value_counts().plot(kind="bar")
         plt.show()
```



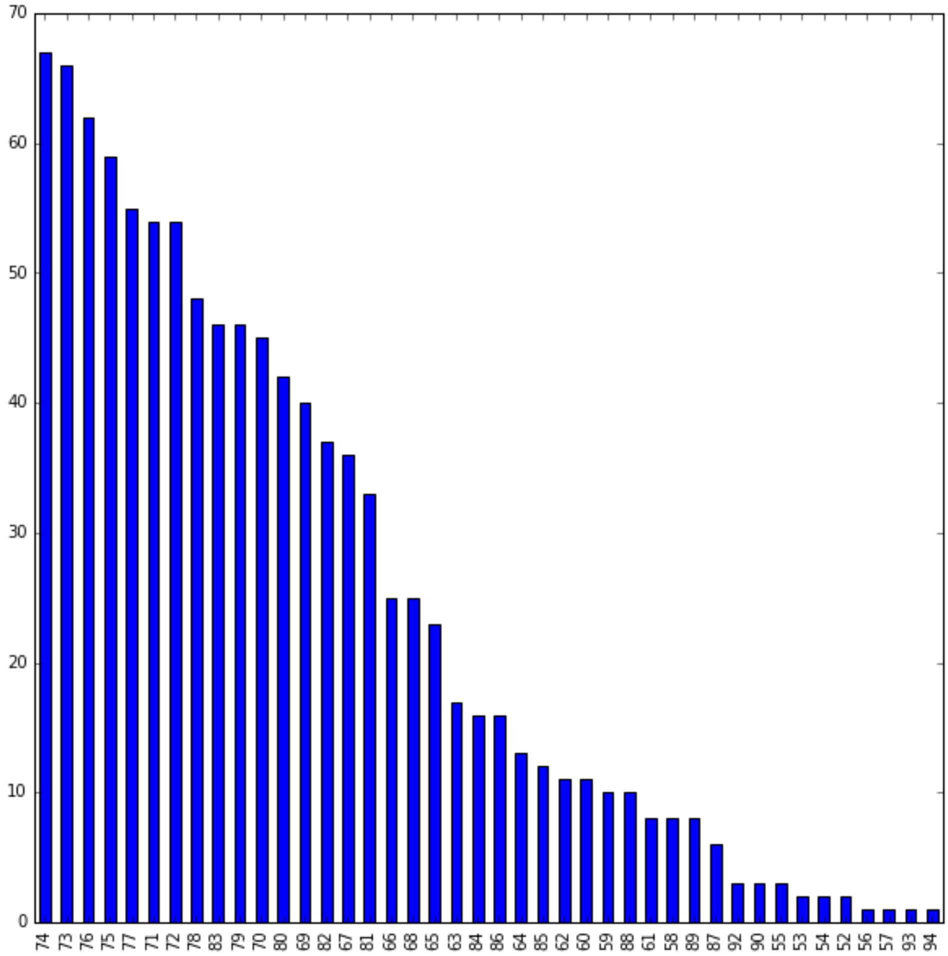
Below is the data that shows how many players have their contract epxiery date in each each in the future and with the players in FIFA17 the majoirty of them have contracts that expire in 2020 but almost the same amount have contracts that expire in 2019 or 2018.

In [25]:
```
refinedFifa['Contract_Expiry'].value_counts().plot(kind="bar")
plt.show()
```



Below is a bar chart showing the distrubiton of rating among the different players. The "Rating" stat in FIFA is based of the over 30 indivudual ratings also in the dataset. Each rating beit the overall Rating or a rating specefic skill is on a scale of 1 - 100 where the higher number means the player is either better overall or better at the specefic skill.

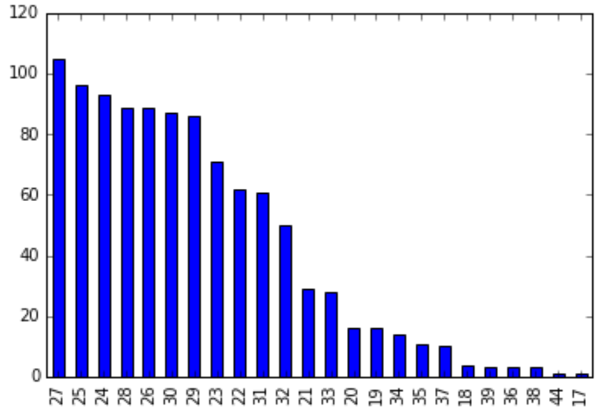
In [26]:
```
refinedFifa['Rating'].value_counts().plot(kind="bar", figsize = (10, 10)
)
plt.show()
```



Below is the age distribution and as you can see there are more 27 year olds than any other age but not by a wide margin.

In [27]:

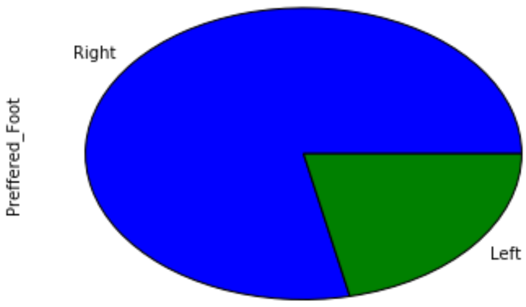
```
refinedFifa['Age'].value_counts().plot(kind="bar")
plt.show()
```



Below is a pie chart showing the distribution between right dominated foot players and left. There is a clear majority of righties than lefties which is not a total surprise as there are more righties in the world as a whole.

In [28]:

```
refinedFifa['Preffered_Foot'].value_counts().plot(kind="pie")
plt.show()
```



Below is a table produced by using groupBy which shows how many righties and lefties play at each position when they are playing for their national teams.

```
In [29]: byPositionAndFoot = refinedFifa[['National_Position', 'Preffered_Foot']]
         .copy()
         byPositionAndFoot = byPositionAndFoot.groupby(['National_Position', 'Pre
         ffered_Foot']).size()
         byPositionAndFoot.reset_index(name='Count')
```

Out[29]:

	National_Position	Preffered_Foot	Count
0	CAM	Left	6
1	CAM	Right	12
2	CB	Right	8
3	CDM	Left	1
4	CDM	Right	8
5	CM	Left	1
6	CM	Right	8
7	GK	Left	4
8	GK	Right	43
9	LAM	Left	2
10	LAM	Right	2
11	LB	Left	29
12	LB	Right	8
13	LCB	Left	16
14	LCB	Right	32
15	LCM	Left	3
16	LCM	Right	22
17	LDM	Left	4
18	LDM	Right	15
19	LF	Right	3
20	LM	Left	7
21	LM	Right	24
22	LS	Left	2
23	LS	Right	15
24	LW	Left	1
25	LW	Right	5
26	LWB	Left	2
27	LWB	Right	1
28	RAM	Right	2
29	RB	Right	37
30	RCB	Left	4
31	RCB	Right	42
32	RCM	Left	3
33	RCM	Right	20
34	RDM	Left	2
35	RDM	Right	16
36	RF	Right	2
37	RM	Left	8
38	RM	Right	25

Below is a table produced by using groupBy which shows how many righties and lefties play at each position when they are playing for their national teams.

In [30]:

```
byClubPositionAndFoot = refinedFifa[['Club_Position', 'Preffered_Foot']]
.copy()
byClubPositionAndFoot = byClubPositionAndFoot.groupby(['Club_Position',
'Preffered_Foot']).size()
byClubPositionAndFoot.reset_index(name='Count')
```

Out[30]:

	Club_Position	Preffered_Foot	Count
0	CAM	Left	8
1	CAM	Right	23
2	CB	Left	1
3	CB	Right	5
4	CDM	Left	3
5	CDM	Right	9
6	CF	Right	1
7	CM	Left	1
8	CM	Right	6
9	GK	Left	9
10	GK	Right	71
11	LAM	Right	3
12	LB	Left	35
13	LB	Right	3
14	LCB	Left	20
15	LCB	Right	32
16	LCM	Left	12
17	LCM	Right	19
18	LDM	Right	14
19	LF	Left	1
20	LF	Right	1
21	LM	Left	13
22	LM	Right	23
23	LS	Left	5
24	LS	Right	7
25	LW	Left	2
26	LW	Right	14
27	RAM	Right	3
28	RB	Right	44
29	RCB	Left	1
30	RCB	Right	48
31	RCM	Right	23
32	RDM	Right	16
33	RF	Left	2
34	RW	Left	7
35	RW	Right	11
36	RWB	Right	4
37	Res	Left	43
38	Res	Right	188

Below is a table produced using groupBy that shows the correleation between height and overall rating. This produces a lot of table entries because the height stat in the dataSet is an exact figure so even a player is a cm different from another than there is a whole new row in the table. In the future possbily refining the height stat in this dataset to produce the height stat as a few ranges of height might yeild better results in this table.

```
In [31]: # height with overall rating
         byHeightAndRating = refinedFifa[['Height', 'Rating']].copy()
         byHeightAndRating = byHeightAndRating.groupby(['Height', 'Rating']).size
         ()
         byHeightAndRating.reset_index(name='Count')
```

Out[31]:

	Height	Rating	Count
0	162 cm	71	1
1	163 cm	84	1
2	164 cm	65	1
3	164 cm	72	1
4	165 cm	66	1
5	165 cm	68	1
6	165 cm	71	1
7	165 cm	86	1
8	166 cm	61	1
9	166 cm	80	1
10	167 cm	69	1
11	167 cm	70	1
12	167 cm	83	1
13	168 cm	65	1
14	168 cm	71	3
15	168 cm	73	1
16	168 cm	74	1
17	168 cm	75	1
18	168 cm	76	1
19	168 cm	78	1
20	168 cm	80	1
21	168 cm	83	1
22	169 cm	63	1
23	169 cm	67	1
24	169 cm	73	2
25	169 cm	74	2
26	169 cm	77	1
27	169 cm	78	1
28	169 cm	79	1
29	169 cm	80	1
...	...	...	...
508	194 cm	77	1
509	194 cm	78	1
510	194 cm	81	1
511	195 cm	58	1
512	195 cm	67	1
513	195 cm	68	1
514	195 cm	69	1
515	195 cm	74	1

Below is a table produced using groupBy that shows the correleation between weight and overall rating. This produces a lot of table entries because the weight stat in the dataSet is an exact figure so even a player is a kg different from another than there is a whole new row in the table. In the future possbily refining the weight stat in this dataset to produce the weight stat as a few ranges of weight like suggested for the height stat might yeild better results in this table.

In [32]:

```
# weight with overall rating
byWeightAndRating = refinedFifa[['Weight', 'Rating']].copy()
byWeightAndRating = byWeightAndRating.groupby(['Weight', 'Rating']).size
()
byWeightAndRating.reset_index(name='Count')
```

Out[32]:

	Weight	Rating	Count
0	58 kg	62	1
1	58 kg	80	1
2	59 kg	68	1
3	59 kg	73	1
4	59 kg	84	1
5	60 kg	65	1
6	60 kg	71	1
7	60 kg	72	1
8	60 kg	73	1
9	60 kg	74	1
10	60 kg	86	1
11	61 kg	73	1
12	61 kg	78	1
13	61 kg	82	1
14	61 kg	85	1
15	62 kg	65	1
16	62 kg	72	1
17	62 kg	73	1
18	62 kg	77	1
19	62 kg	78	1
20	62 kg	83	1
21	62 kg	88	1
22	63 kg	70	1
23	63 kg	74	1
24	63 kg	75	1
25	63 kg	80	1
26	63 kg	82	1
27	64 kg	63	1
28	64 kg	71	2
29	64 kg	75	2
...	...	...	...
529	91 kg	83	2
530	91 kg	85	1
531	91 kg	88	1
532	91 kg	89	1
533	92 kg	65	1
534	92 kg	67	1
535	92 kg	72	1
536	92 kg	76	1

Below is table produced using groupBy that shows the correlation between age and rating and from looking at is looks like the majority of the high ratings reside in the middle of the age range.

In [33]:

```
byAgeAndRating = refinedFifa[['Age', 'Rating']].copy()
byAgeAndRating = byAgeAndRating.groupby(['Age', 'Rating']).size()
byAgeAndRating.reset_index(name='Count')
```

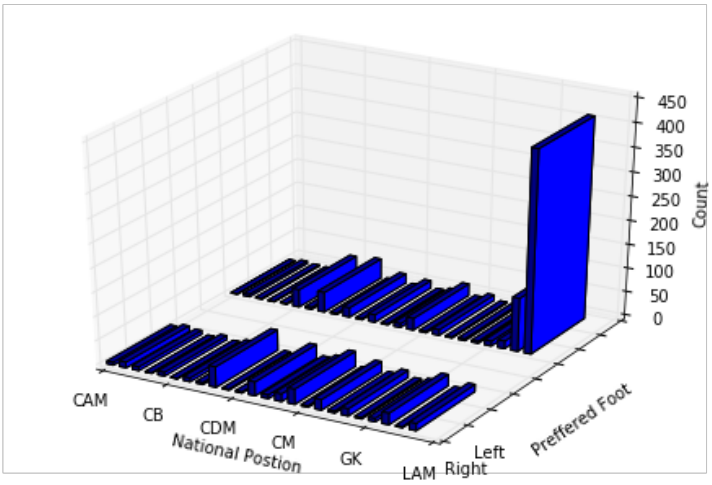
Out[33]:

	Age	Rating	Count
0	17	63	1
1	18	62	1
2	18	75	1
3	18	77	1
4	18	79	1
5	19	53	1
6	19	54	1
7	19	62	1
8	19	64	1
9	19	65	1
10	19	67	1
11	19	70	2
12	19	71	1
13	19	72	1
14	19	75	2
15	19	77	1
16	19	78	2
17	19	81	1
18	20	52	1
19	20	59	1
20	20	61	1
21	20	64	1
22	20	67	1
23	20	69	1
24	20	70	2
25	20	71	1
26	20	74	1
27	20	75	1
28	20	76	1
29	20	77	1
...	...	...	...
358	34	77	1
359	34	80	1
360	34	83	1
361	34	88	1
362	35	69	1
363	35	70	1
364	35	71	1
365	35	73	2

Below is a 3D graph of the table showing the relationship between national position and preffered foot. The x axis is the national position, the y axis is the preffered foot stat, and finally the z axis is the the count when these two variables intersect.

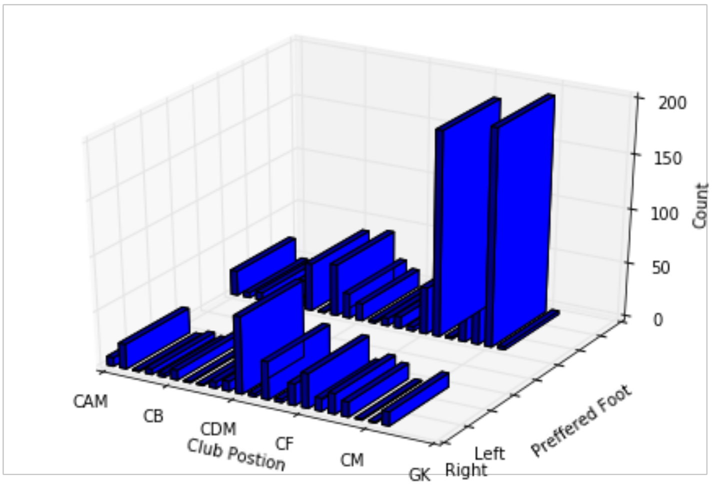
In [34]:

```
# national position and preffered foot - 3d graph
national_position_array = ["CAM", "CB", "CDM", "CM", "GK", "LAM", "LB",
"LCB", "LCM", "LDM", "LF", "LM", "LS", "LW", "LWB", "RAM", "RB", "RCB",
"RCM", "RDM", "RF", "RM", "RS", "RW", "RWB", "ST", "Sub"]
#national_position_array = ["CAM", "CB", "CDM", "CM", "GK", "LAM", "LB",
"LCB", "LCM", "LDM", "LF", "LM", "LS", "LW", "LWB", "RAM", "RB", "RCB",
"RCM", "RDM", "RF", "RM", "RS", "RW", "RWB"]
preffered_foot_array = ["Right", "Left"]

fig = plt.figure()
ax = fig.add_subplot(111, projection = '3d')
ax = Axes3D(fig)
ax.set_xlabel("National Postion")
ax.set_ylabel("Preffered Foot")
ax.set_zlabel("Count")

byPositionAndFoot = refinedFifa[['National_Position', 'Preffered_Foot']]
.copy()
byPositionAndFoot = byPositionAndFoot.groupby(['National_Position', 'Pre
ffered_Foot'])

z = byPositionAndFoot.size().tolist()
axes = byPositionAndFoot.groups.keys()
axes = sorted(axes, key=itemgetter(1))
axes = sorted(axes, key=itemgetter(0))

x = list(range(0, (len(national_position_array)-3)))
y = list(range(0, len(preffered_foot_array)))
X, Y = np.meshgrid(x,y)
zs = np.array(z)
Z = zs.reshape(Y.shape)

values = np.linspace(0.2,1.,X.ravel().shape[0])
colours = plt.cm.Spectral(values)

ax.bar3d(X.ravel(), Y.ravel(), Z.ravel()*0, dx=0.5, dy=0.5, dz=Z.ravel()
)

ax.set_xticklabels(np.array(national_position_array))
ax.set_yticklabels(np.array(preffered_foot_array))

plt.show()
```



Below is a 3D graph showing the relationship between preffered foot and position just like the above chart but in this case club positions are looked at instead of national positions. This was done not only because it is good to compare between national positions and club positions but as well as you can see in the national positions chart, most players are substitutes which is not the case for the club positions. As you can see the distribution of players among different positions and not just the substitue role is much greater in this chart. This can espcially be seen with the center defense midfielder position where there are many more players that play that position for their club team and not their national team.

```
In [35]: # club position and foot
foot_array = ["Right", "Left"]
#clubPositions = ["CAM", "CB", "CDM", "CF", "CM", "GK", "LAM", "LB", "LC
B", "LCM", "LDM", "LF", "LM", "LS", "LW", "LWB", "RAM", "RB", "RCB", "RC
M", "RDM", "RF", "RM" "RS", "RW", "RWB", "Res", "ST", "Sub"]
club_position_array = ["CAM", "CB", "CDM", "CF", "CM", "GK", "LAM", "LB"
, "LCB", "LCM", "LDM", "LF", "LM", "LS", "LW", "LW", "RAM", "RB", "RCB",
"RCM", "RDM", "RF", "RM", "RS", "RW", "RWB", "Res", "ST", "Sub"]

fig = plt.figure()
ax = fig.add_subplot(111, projection = '3d')
ax = Axes3D(fig)
ax.set_xlabel("Club Postion")
ax.set_ylabel("Preffered Foot")
ax.set_zlabel("Count")

byClubPositionAndFoot = refinedFifa[['Club_Position', 'Preffered_Foot']]
.copy()
byClubPositionAndFoot = byClubPositionAndFoot.groupby(['Club_Position',
'Preffered_Foot'])

z = byClubPositionAndFoot.size().tolist() + [3]
axes = byPositionAndFoot.groups.keys()
axes = sorted(axes, key=itemgetter(1))
axes = sorted(axes, key=itemgetter(0))

x = list(range(0, (len(club_position_array)-7)))
y = list(range(0, len(foot_array)))

X, Y = np.meshgrid(x,y)
zs = np.array(z)
Z = zs.reshape(Y.shape)

values = np.linspace(0.2,1.,X.ravel().shape[0])
colours = plt.cm.Spectral(values)

ax.bar3d(X.ravel(), Y.ravel(), Z.ravel()*0, dx=0.5, dy=0.5, dz=Z.ravel()
)

ax.set_xticklabels(np.array(club_position_array))
ax.set_yticklabels(np.array(foot_array))

plt.show()
```



Below is a widget using ipywdigets that allows you to selected a specefic player and see all of there stats and other data that pertains to them like which team they play for and their position.

In [36]:

```
def update (Player = list(refinedFifa['Name'].unique())):
    rating = refinedFifa[(refinedFifa['Name'] == Player)]
    display(rating)

interact(update);
```

In [ ]: