FIFA Football Player Analysis

In [1]:
```
# import all the libraries
import numpy as np # import numpy

import pandas as pd # import pandas

import matplotlib.pyplot as plt # import matplotlib

import seaborn as sns # import seaborn

from collections import Counter # counter is the class from collection module which help to find the occurances

%matplotlib inline # bring all the plot inside the note pad

sns.set(style='whitegrid') # chage plot backgroud to white grid

# ignore all warnings
import warnings
warnings.filterwarnings('ignore')
```

```
UsageError: unrecognized arguments: # bring all the plot inside the note pad
```

In [2]:
```
# read the data frame

data = pd.read_csv(r"C:\Users\sunil\Desktop\NIT- Data Science and AI Class\September\9th- Seaborn, Eda practicl
data.head() # read and get first 5 data
```

Out[2]:

	Unnamed: 0	ID	Name	Age	Photo	Nationality	Flag	Overall
0	0	158023	L. Messi	31	https://cdn.sofifa.org/players/4/19/158023.png	Argentina	https://cdn.sofifa.org/flags/52.png	94
1	1	20801	Cristiano Ronaldo	33	https://cdn.sofifa.org/players/4/19/20801.png	Portugal	https://cdn.sofifa.org/flags/38.png	94
2	2	190871	Neymar Jr	26	https://cdn.sofifa.org/players/4/19/190871.png	Brazil	https://cdn.sofifa.org/flags/54.png	92
3	3	193080	De Gea	27	https://cdn.sofifa.org/players/4/19/193080.png	Spain	https://cdn.sofifa.org/flags/45.png	91
4	4	192985	K. De Bruyne	27	https://cdn.sofifa.org/players/4/19/192985.png	Belgium	https://cdn.sofifa.org/flags/7.png	91

5 rows × 89 columns

In [3]:
```
# information about the data frame

data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 18207 entries, 0 to 18206
Data columns (total 89 columns):
 #   Column                    Non-Null Count  Dtype
---  ------                    --------------  -----
 0   Unnamed: 0                18207 non-null  int64
 1   ID                        18207 non-null  int64
 2   Name                      18207 non-null  object
 3   Age                       18207 non-null  int64
 4   Photo                     18207 non-null  object
 5   Nationality               18207 non-null  object
 6   Flag                      18207 non-null  object
 7   Overall                   18207 non-null  int64
 8   Potential                 18207 non-null  int64
 9   Club                      17966 non-null  object
 10  Club Logo                 18207 non-null  object
 11  Value                     18207 non-null  object
 12  Wage                      18207 non-null  object
 13  Special                   18207 non-null  int64
 14  Preferred Foot            18159 non-null  object
 15  International Reputation  18159 non-null  float64
 16  Weak Foot                 18159 non-null  float64
 17  Skill Moves               18159 non-null  float64
 18  Work Rate                 18159 non-null  object
 19  Body Type                 18159 non-null  object
 20  Real Face                 18159 non-null  object
 21  Position                  18147 non-null  object
 22  Jersey Number             18147 non-null  float64
 23  Joined                    16654 non-null  object
```

```
 24  Loaned From               1264 non-null   object
 25  Contract Valid Until      17918 non-null  object
 26  Height                    18159 non-null  object
 27  Weight                    18159 non-null  object
 28  LS                        16122 non-null  object
 29  ST                        16122 non-null  object
 30  RS                        16122 non-null  object
 31  LW                        16122 non-null  object
 32  LF                        16122 non-null  object
 33  CF                        16122 non-null  object
 34  RF                        16122 non-null  object
 35  RW                        16122 non-null  object
 36  LAM                       16122 non-null  object
 37  CAM                       16122 non-null  object
 38  RAM                       16122 non-null  object
 39  LM                        16122 non-null  object
 40  LCM                       16122 non-null  object
 41  CM                        16122 non-null  object
 42  RCM                       16122 non-null  object
 43  RM                        16122 non-null  object
 44  LWB                       16122 non-null  object
 45  LDM                       16122 non-null  object
 46  CDM                       16122 non-null  object
 47  RDM                       16122 non-null  object
 48  RWB                       16122 non-null  object
 49  LB                        16122 non-null  object
 50  LCB                       16122 non-null  object
 51  CB                        16122 non-null  object
 52  RCB                       16122 non-null  object
 53  RB                        16122 non-null  object
 54  Crossing                  18159 non-null  float64
 55  Finishing                 18159 non-null  float64
 56  HeadingAccuracy           18159 non-null  float64
 57  ShortPassing              18159 non-null  float64
 58  Volleys                   18159 non-null  float64
 59  Dribbling                 18159 non-null  float64
 60  Curve                     18159 non-null  float64
 61  FKAccuracy                18159 non-null  float64
 62  LongPassing               18159 non-null  float64
 63  BallControl               18159 non-null  float64
 64  Acceleration              18159 non-null  float64
 65  SprintSpeed               18159 non-null  float64
 66  Agility                   18159 non-null  float64
 67  Reactions                 18159 non-null  float64
 68  Balance                   18159 non-null  float64
 69  ShotPower                 18159 non-null  float64
 70  Jumping                   18159 non-null  float64
 71  Stamina                   18159 non-null  float64
 72  Strength                  18159 non-null  float64
 73  LongShots                 18159 non-null  float64
 74  Aggression                18159 non-null  float64
 75  Interceptions             18159 non-null  float64
 76  Positioning               18159 non-null  float64
 77  Vision                    18159 non-null  float64
 78  Penalties                 18159 non-null  float64
 79  Composure                 18159 non-null  float64
 80  Marking                   18159 non-null  float64
 81  StandingTackle            18159 non-null  float64
 82  SlidingTackle             18159 non-null  float64
 83  GKDiving                  18159 non-null  float64
 84  GKHandling                18159 non-null  float64
 85  GKKicking                 18159 non-null  float64
 86  GKPositioning             18159 non-null  float64
 87  GKReflexes                18159 non-null  float64
 88  Release Clause            16643 non-null  object
dtypes: float64(38), int64(6), object(45)
memory usage: 12.4+ MB
```

In [4]:
```
#shape of the data frmae
data.shape
```

Out[4]:
```
(18207, 89)
```

explore age variable

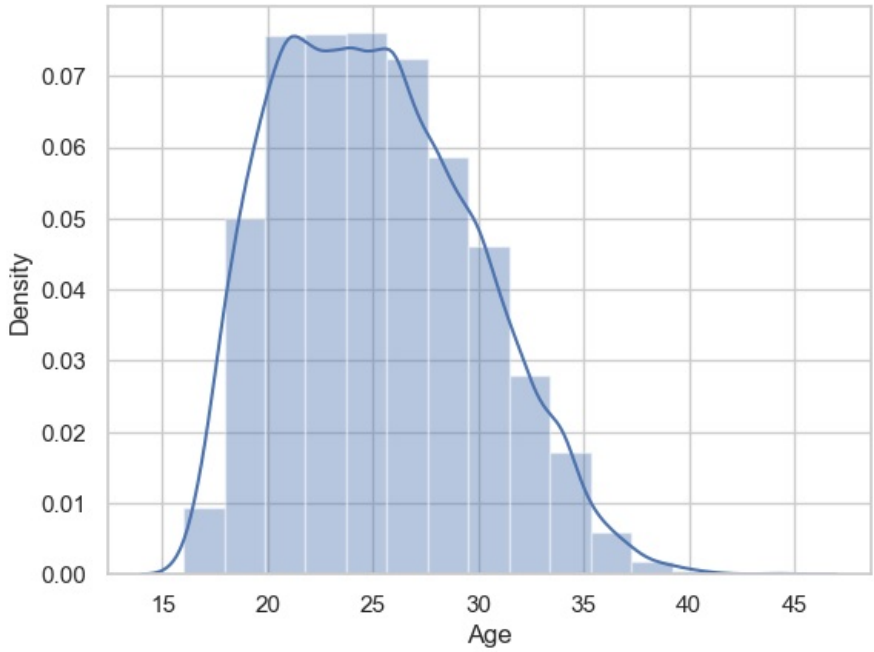
Here we explore the age attribute with the distplot

In [5]:
```
data['Age'].head()
```

Out[5]:
```
0    31
1    33
2    26
3    27
4    27
Name: Age, dtype: int64
```

In [6]:
```
# plot a distplot
sns.set(style='whitegrid') # background is whitegrid
sns.distplot(data['Age'],bins=15) # plot the graph with age variable of 15 bar
plt.show()
```

```
C:\Users\sunil\AppData\Local\Temp\ipykernel_14544\2219252783.py:3: UserWarning:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with
similar flexibility) or `histplot` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  sns.distplot(data['Age'],bins=15) # plot the graph with age variable of 15 bar
```

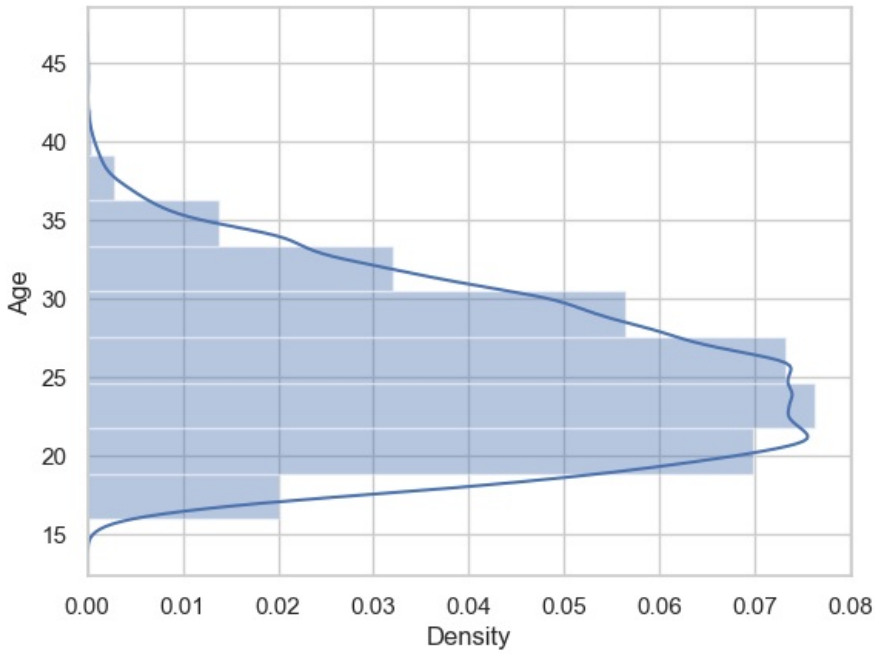


here we can see that the age variable slightly positively skewed

In [7]:
```
# we can do this plot with veertical

sns.distplot(data['Age'],bins=10,vertical=True) # vertical= True use to bring the graph into vertical
```

```
C:\Users\sunil\AppData\Local\Temp\ipykernel_14544\2138165478.py:3: UserWarning:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with
similar flexibility) or `histplot` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  sns.distplot(data['Age'],bins=10,vertical=True) # vertical= True use to bring the graph into vertical
```

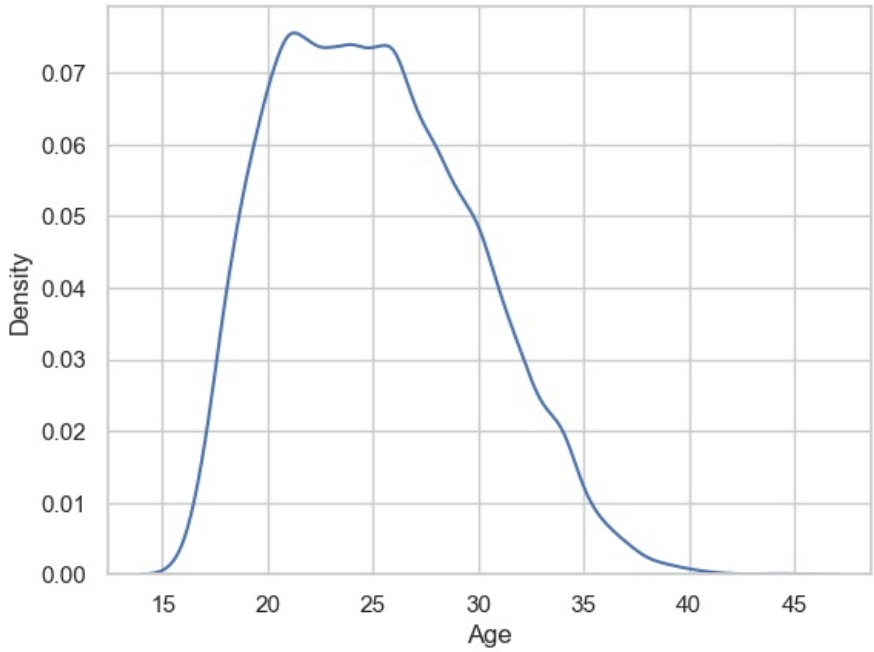
Out[7]:
```
<Axes: xlabel='Density', ylabel='Age'>
```



KDE plot (Kernel density plot)

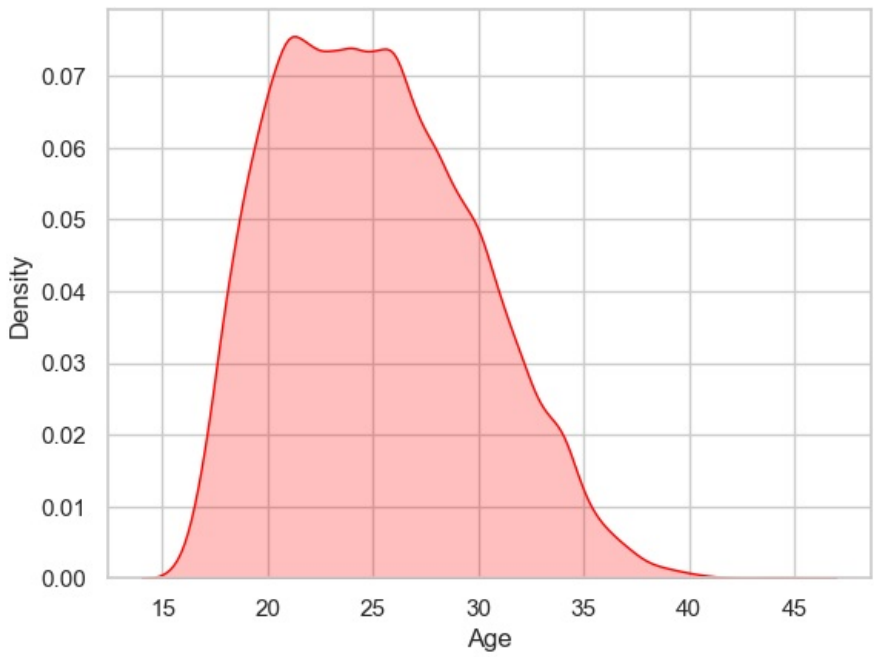
In [8]:
```
sns.kdeplot(data['Age']) # normal kde plot
```

Out[8]:
```
<Axes: xlabel='Age', ylabel='Density'>
```



In [9]:
```
# here we have to add some shade/fill to it and also color

sns.kdeplot(data['Age'],fill=True,color='red')
```

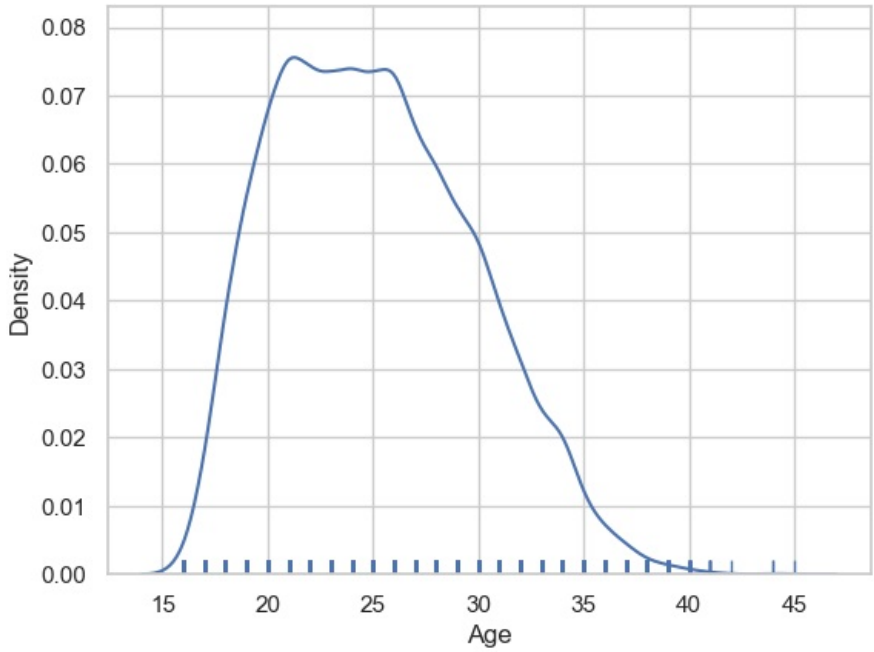
Out[9]:
```
<Axes: xlabel='Age', ylabel='Density'>
```



In [10]:
```
## without use kdeplot() draw the plot

sns.distplot(data['Age'],hist=False,rug=True, bins=10)
```

```
C:\Users\sunil\AppData\Local\Temp\ipykernel_14544\2241101863.py:3: UserWarning:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with
similar flexibility) or `kdeplot` (an axes-level function for kernel density plots).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  sns.distplot(data['Age'],hist=False,rug=True, bins=10)
```

Out[10]:
```
<Axes: xlabel='Age', ylabel='Density'>
```

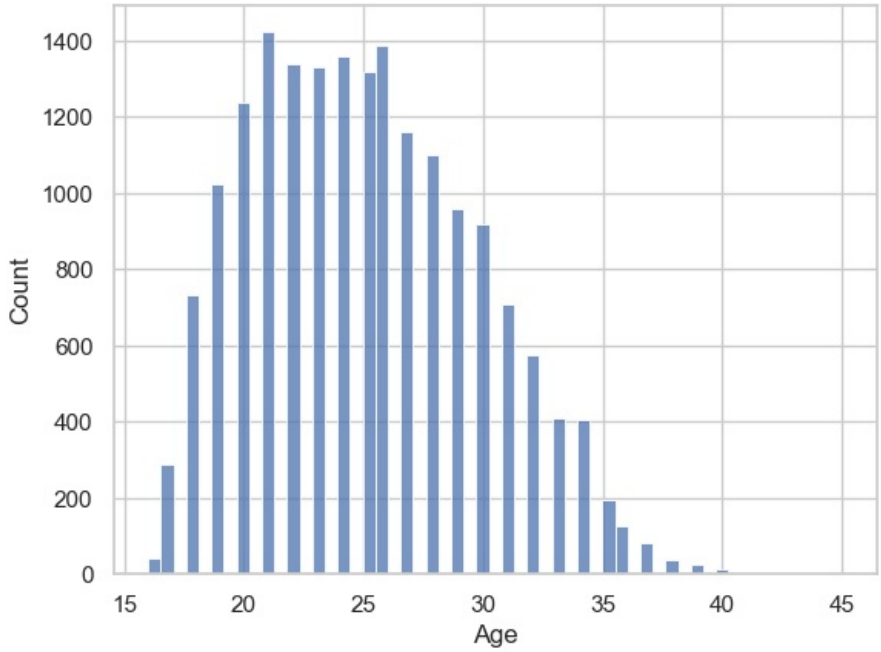


histogram

here we work on histogram with age variable

In [11]:
```
sns.histplot(data['Age']) # histplot of age variable
```

Out[11]: <Axes: xlabel='Age', ylabel='Count'>



Explore the prefered foot variable

In [12]:
```
# find out how many unique value on preferred foot value
data['Preferred Foot'].unique()
```

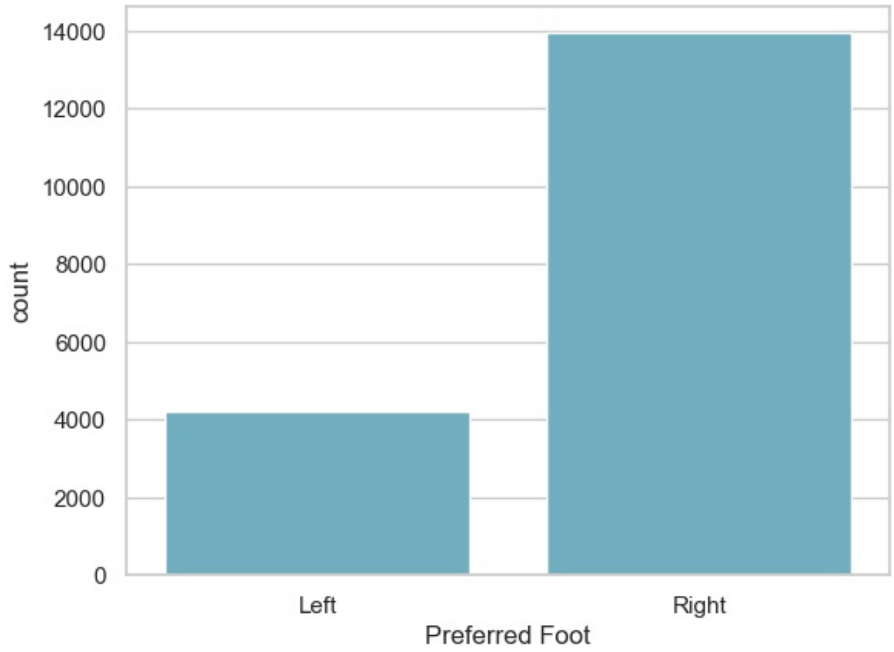
Out[12]:
```
array(['Left', 'Right', nan], dtype=object)
```

In [13]:
```
data['Preferred Foot'].value_counts() # distinct cout the value of preferred foot variable
```

Out[13]:
```
Preferred Foot
Right    13948
Left      4211
Name: count, dtype: int64
```

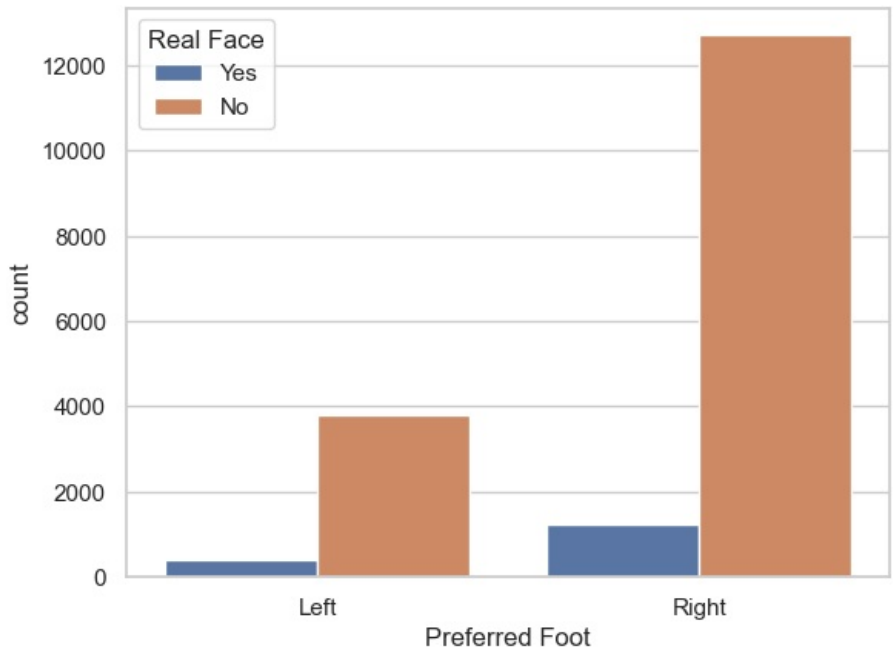
In [14]:
```
sns.countplot(x=data['Preferred Foot'],color="c") # count plot
```

Out[14]: <Axes: xlabel='Preferred Foot', ylabel='count'>



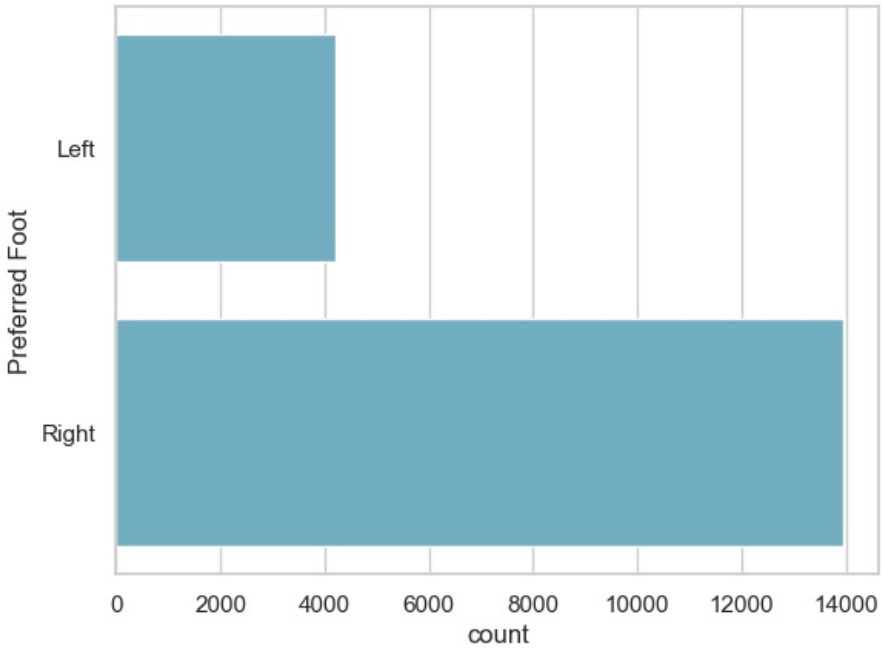
In [15]:
```
sns.countplot(data,x='Preferred Foot',hue='Real Face') # count plot in x axis use preferred foot and hue as rea
```

Out[15]: <Axes: xlabel='Preferred Foot', ylabel='count'>



In [16]:
```
sns.countplot(y=data['Preferred Foot'],color='c') # cahge the cout plot with vertical means take preferred foot
```

Out[16]:
```
<Axes: xlabel='count', ylabel='Preferred Foot'>
```



explore international reputation variable

In [17]:
```
data['International Reputation'].unique # get all the unique value from the international raputation
```

Out[17]:
```
<bound method Series.unique of 0        5.0
1        5.0
2        5.0
3        4.0
4        4.0
         ...
18202    1.0
18203    1.0
18204    1.0
18205    1.0
18206    1.0
Name: International Reputation, Length: 18207, dtype: float64>
```

In [18]:
```
data['International Reputation'].value_counts() # distinct cout of the international raputation
```

Out[18]:
```
International Reputation
1.0    16532
2.0     1261
3.0      309
4.0       51
5.0        6
Name: count, dtype: int64
```

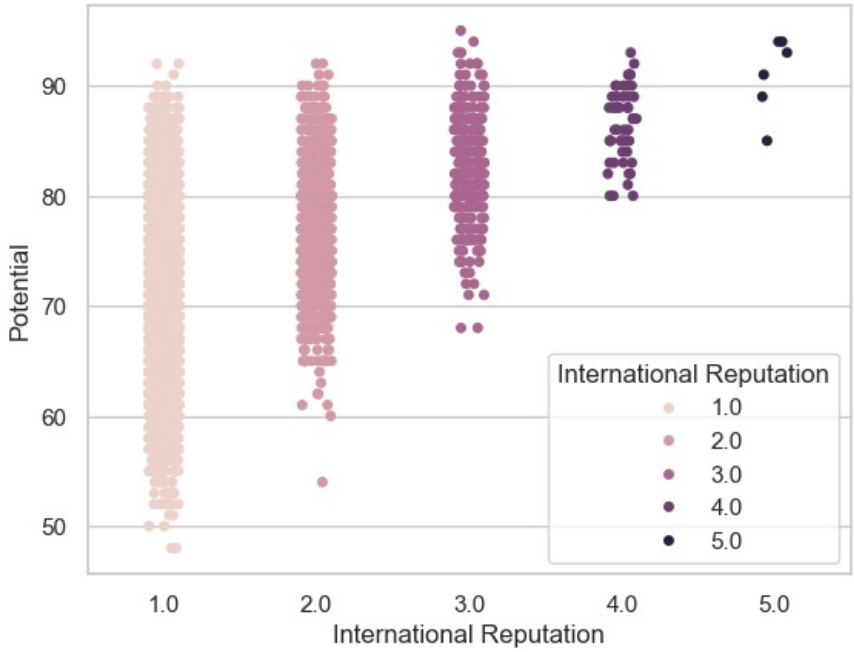
In [19]:
```
data['Potential'].unique() # get all the unique value of potential variable
```

Out[19]:
```
array([94, 93, 92, 91, 90, 89, 95, 88, 87, 86, 85, 84, 83, 82, 81, 80, 79,
       78, 77, 76, 75, 74, 73, 72, 71, 70, 69, 68, 67, 66, 65, 64, 63, 62,
       61, 60, 59, 58, 57, 56, 55, 54, 53, 52, 51, 50, 48], dtype=int64)
```

Stripplot

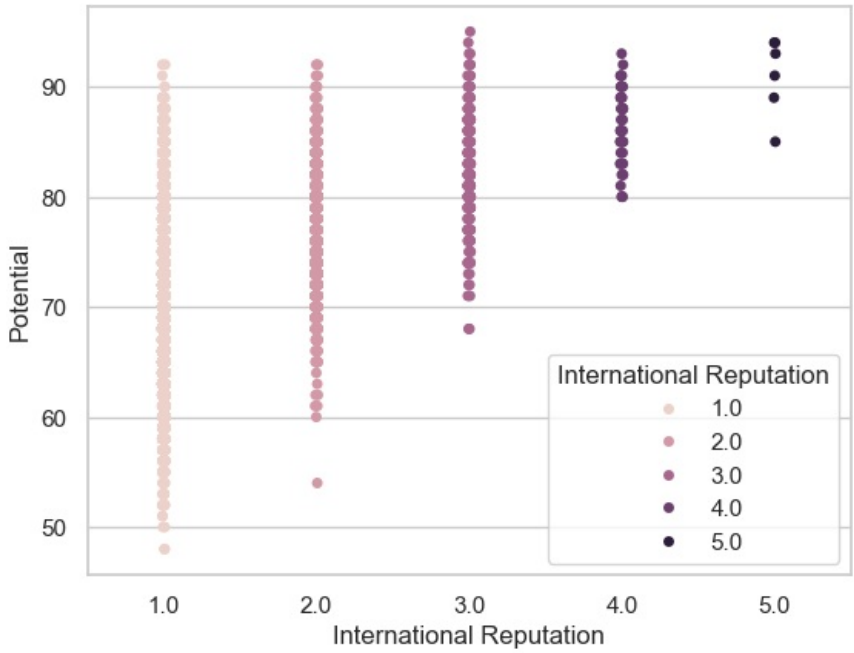
In [20]:
```
# strip plot between international raputation and potential, ading hue at international raputatuion
sns.stripplot(x=data['International Reputation'],y=data['Potential'],hue=data['International Reputation'])
```

Out[20]:
```
<Axes: xlabel='International Reputation', ylabel='Potential'>
```



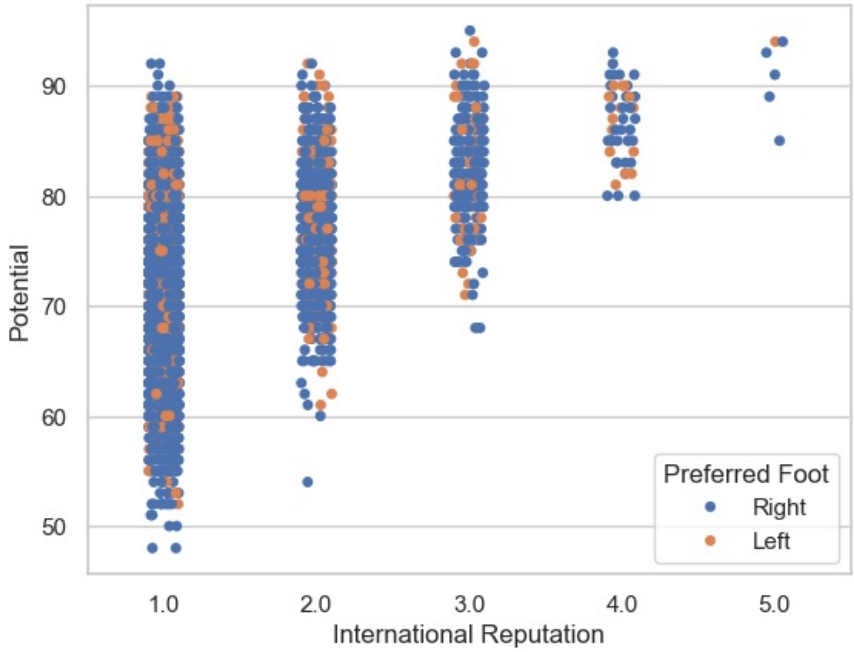
In [21]:
```
# add jitter value to make the whole plot data points to squized
sns.stripplot(x=data['International Reputation'],y=data['Potential'],hue=data['International Reputation'],jitte
```

Out[21]:
```
<Axes: xlabel='International Reputation', ylabel='Potential'>
```



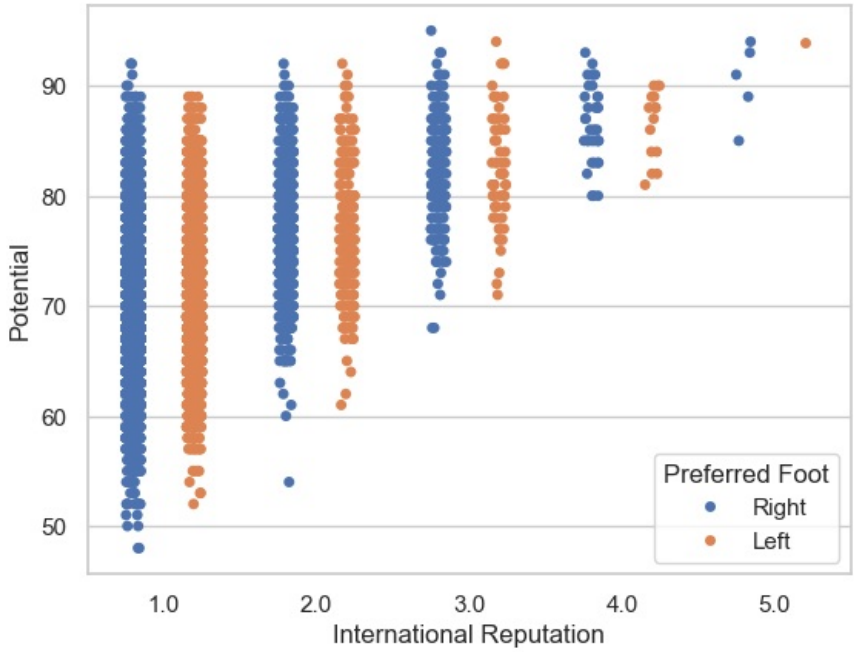
In [22]:
```
# adding hue as preffered foot
sns.stripplot(x=data['International Reputation'],y=data['Potential'],hue=data['Preferred Foot'])
```

Out[22]: <Axes: xlabel='International Reputation', ylabel='Potential'>



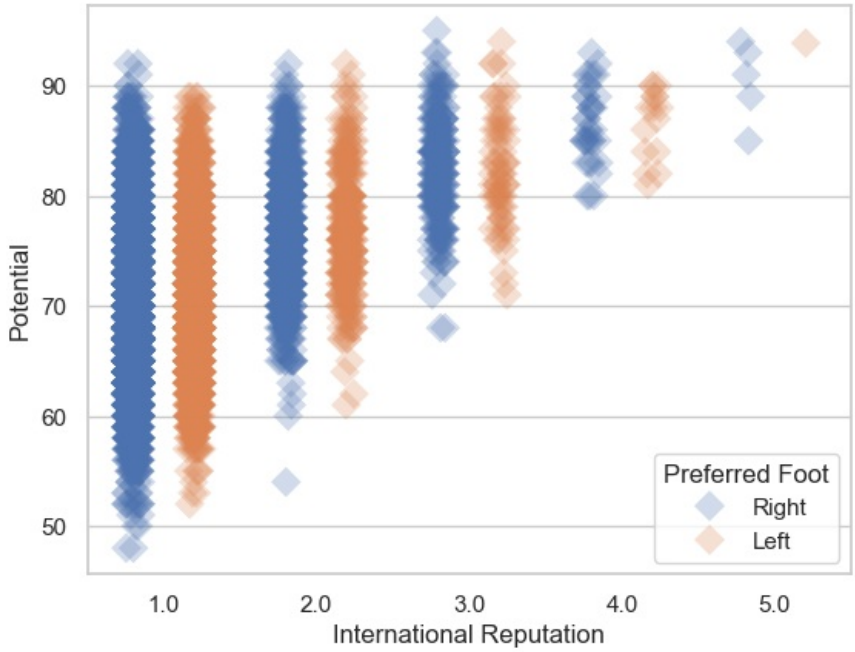
In [23]:
```
# adding in the plot dodge
sns.stripplot(x=data['International Reputation'],y=data['Potential'],hue=data['Preferred Foot'],dodge=True)
```

Out[23]: <Axes: xlabel='International Reputation', ylabel='Potential'>



In [24]:
```
# adding marker and alpha(for less view the fig)
sns.stripplot(x=data['International Reputation'],y=data['Potential'],hue=data['Preferred Foot'],dodge=True, marl
```

Out[24]: <Axes: xlabel='International Reputation', ylabel='Potential'>

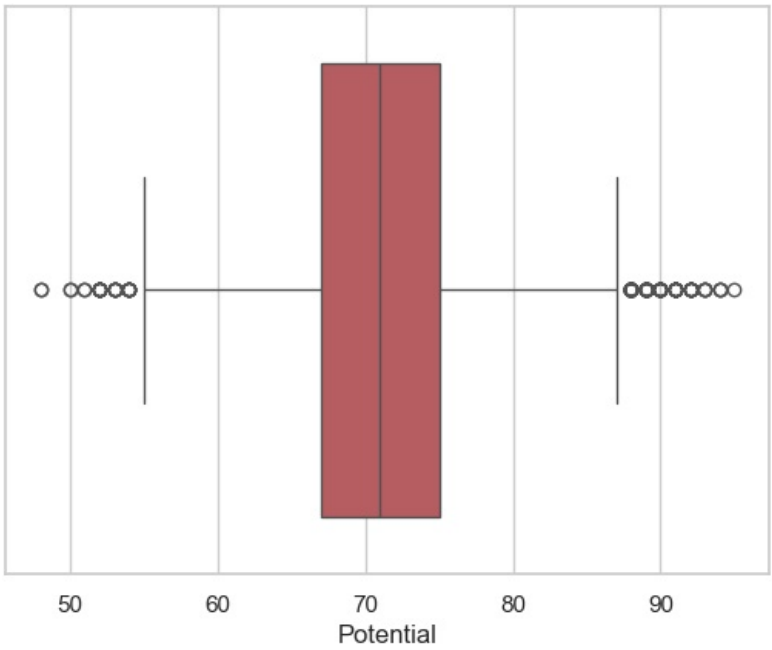


Box- plot

work on potential variable

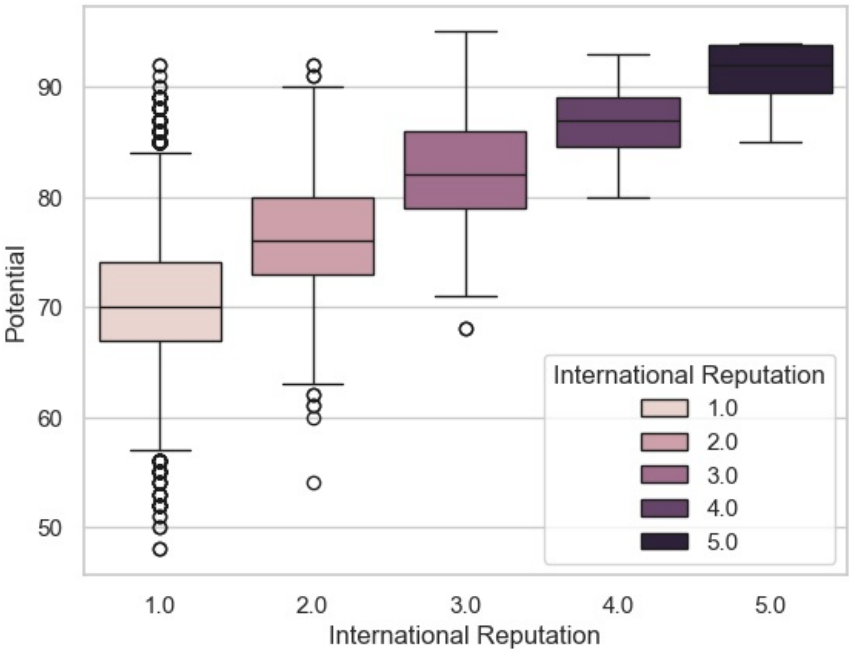
In [25]:
```
# box plot on potential variable with color red
sns.boxplot(x=data['Potential'],color='r',)
```

Out[25]:
```
<Axes: xlabel='Potential'>
```



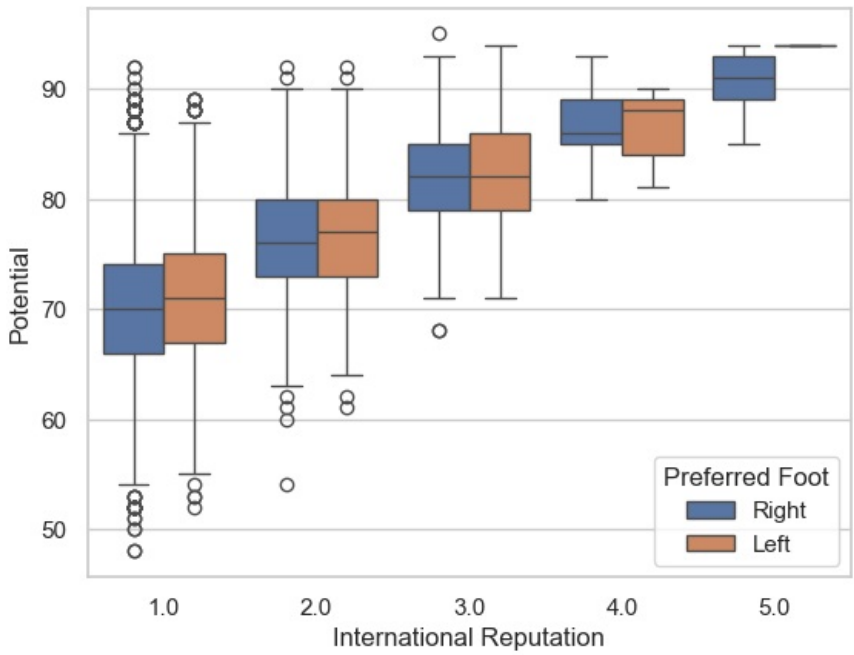
In [26]:
```
# potential at x axis and internatioonal at y axis and also hue
sns.boxplot(y=data['Potential'],x=data['International Reputation'],hue=data['International Reputation'])
```

Out[26]:
```
<Axes: xlabel='International Reputation', ylabel='Potential'>
```



In [31]:
```
# adding hue as preferred variable
sns.boxplot(y=data['Potential'],x=data['International Reputation'],hue=data['Preferred Foot'])
```

Out[31]: <Axes: xlabel='International Reputation', ylabel='Potential'>

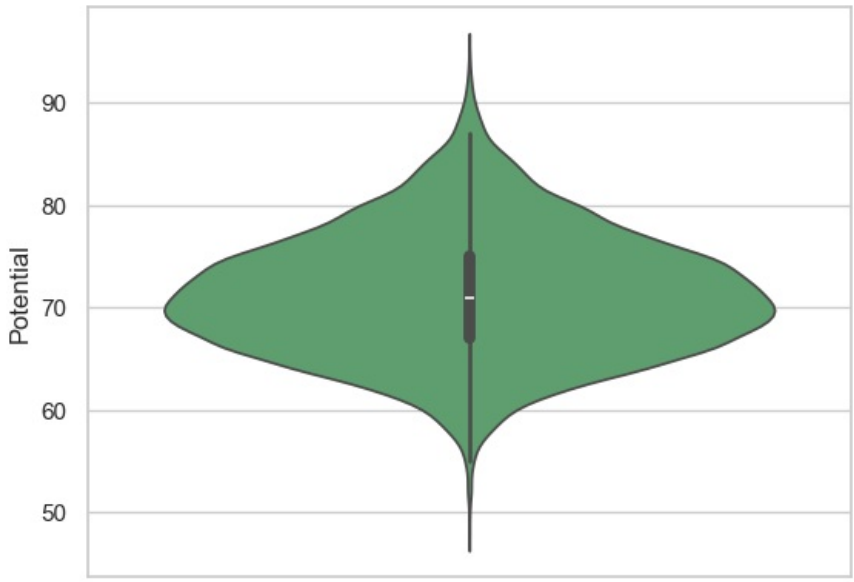


voilin plot

work on potential variable

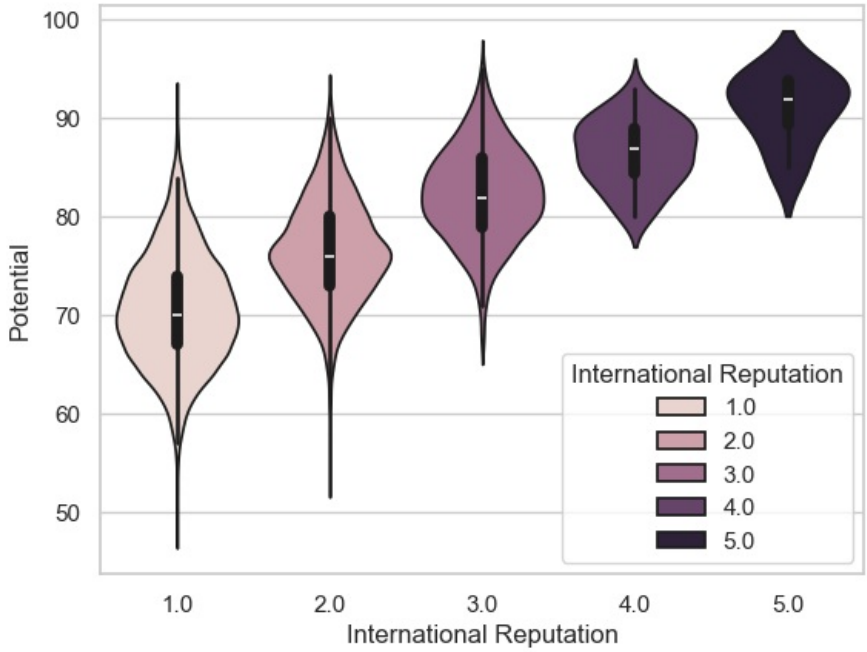
In [32]:
```
# voilin plot of potential variable with green color
sns.violinplot(data['Potential'],color="g")
```

Out[32]: <Axes: ylabel='Potential'>



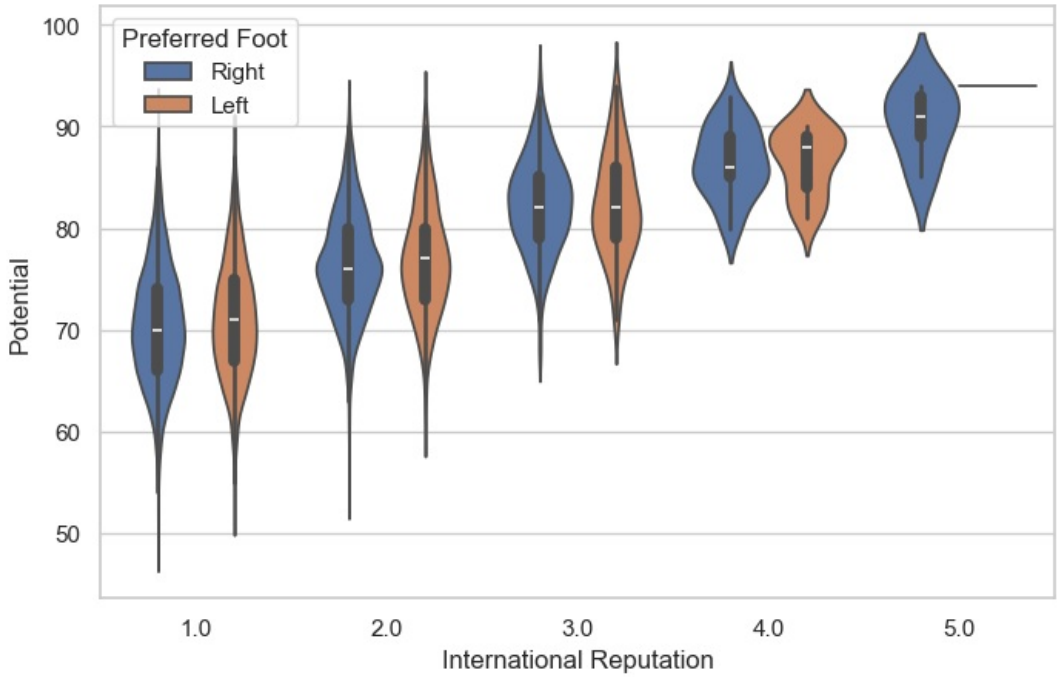
In [33]:
```
## potential at x axis and internatioonal at y axis and also hue
sns.violinplot(x=data['International Reputation'],y=data['Potential'],hue=data['International Reputation'])
```

Out[33]: <Axes: xlabel='International Reputation', ylabel='Potential'>



In [34]:
```
plt.subplots(figsize=(8,5))  # chage figure size

# adding hue as preferred variable
sns.violinplot(x=data['International Reputation'],y=data['Potential'],hue=data['Preferred Foot'])
```

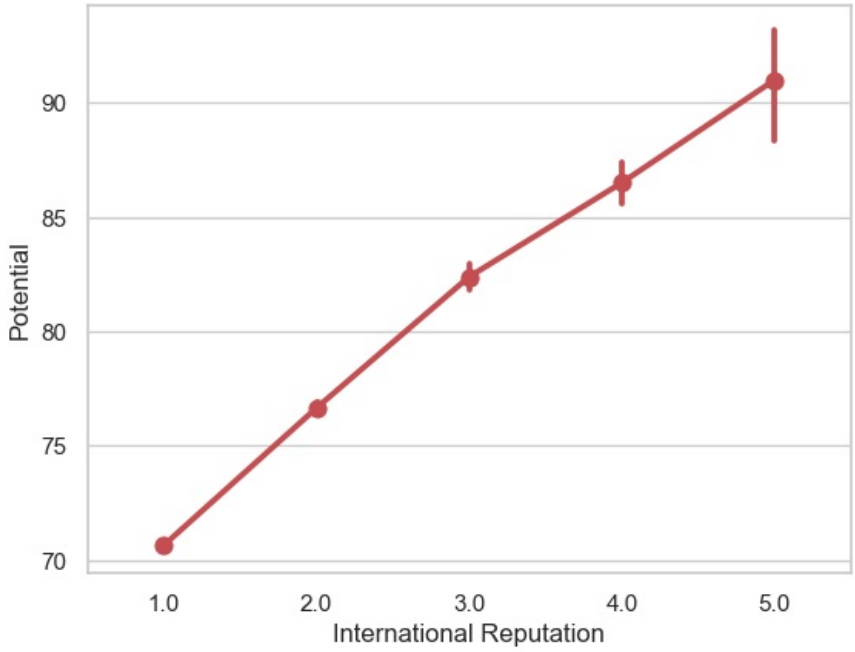
Out[34]:
```
<Axes: xlabel='International Reputation', ylabel='Potential'>
```



point plot

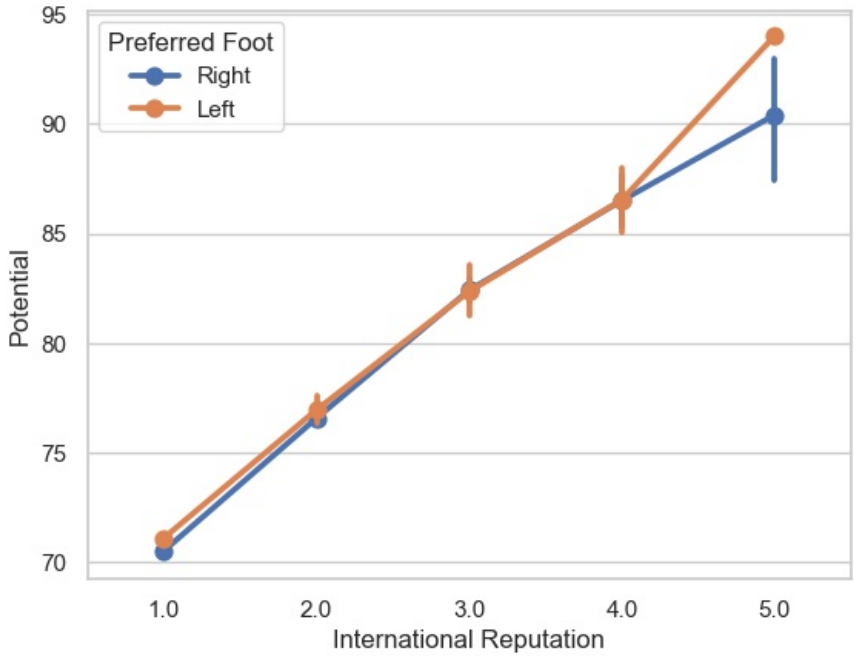
In [35]:
```
# point plot with international reputationa and potential variable with color red
sns.pointplot(x=data['International Reputation'],y=data['Potential'],color='r')
```

Out[35]:
```
<Axes: xlabel='International Reputation', ylabel='Potential'>
```



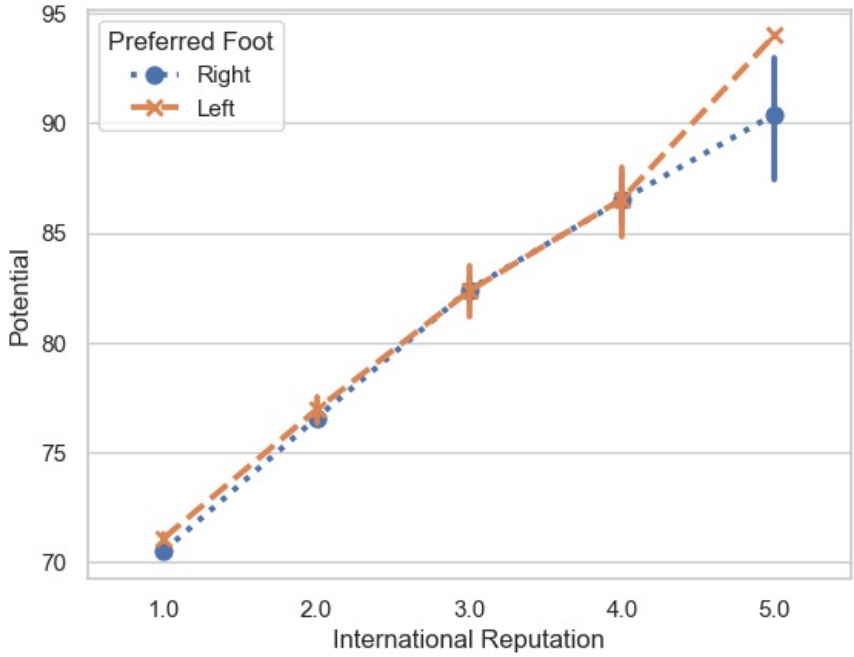
In [36]:
```
# add hue as preferred foot
sns.pointplot(x=data['International Reputation'],y=data['Potential'],hue=data['Preferred Foot'])
```

Out[36]:
```
<Axes: xlabel='International Reputation', ylabel='Potential'>
```



In [37]:
```
# adding markers and line style
sns.pointplot(x=data['International Reputation'],y=data['Potential'],hue=data['Preferred Foot'],markers=['o','x
```

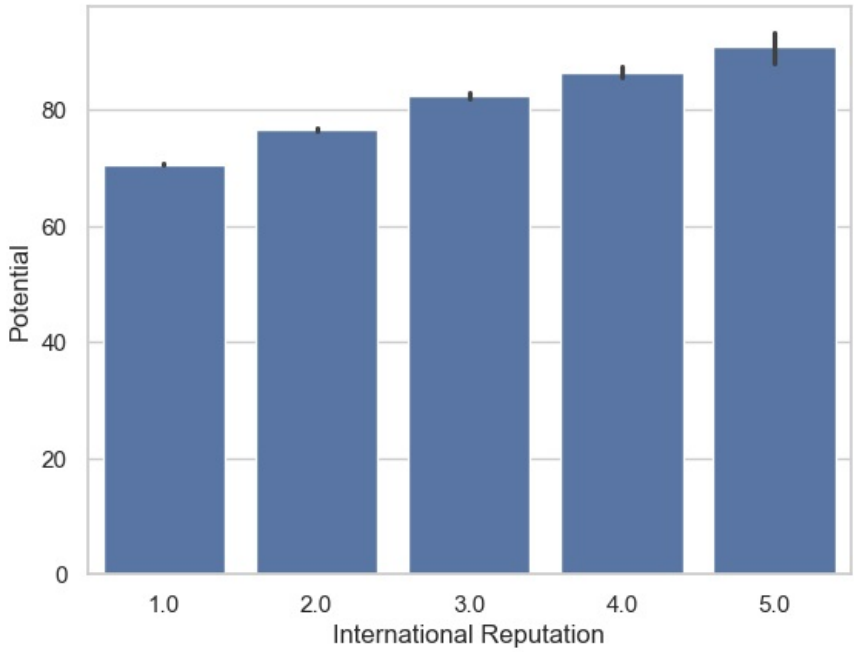
Out[37]:
```
<Axes: xlabel='International Reputation', ylabel='Potential'>
```



Bar plot

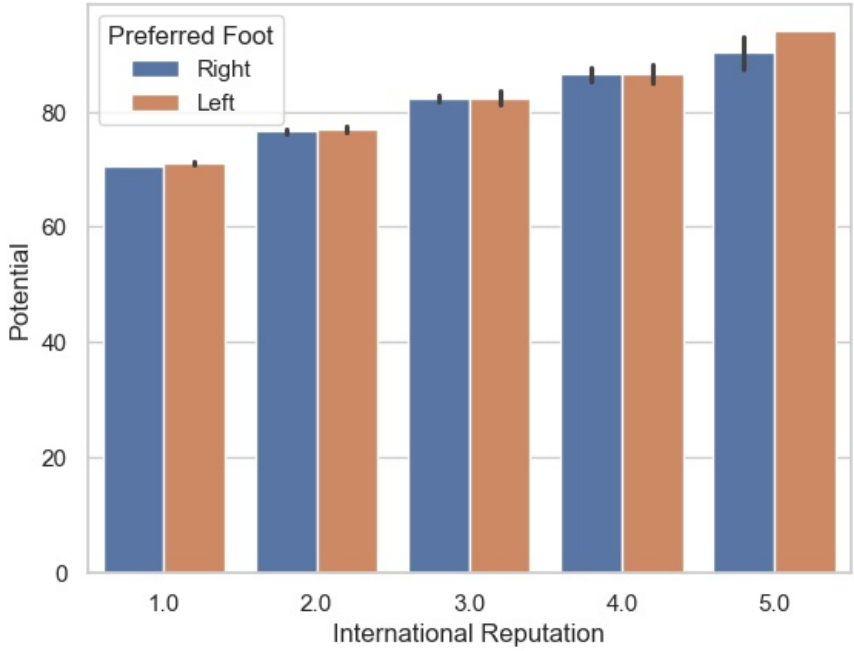
In [38]:
```
# bar plot respect to international abd potential variable
sns.barplot(x=data['International Reputation'],y=data['Potential'])
```

Out[38]:
```
<Axes: xlabel='International Reputation', ylabel='Potential'>
```



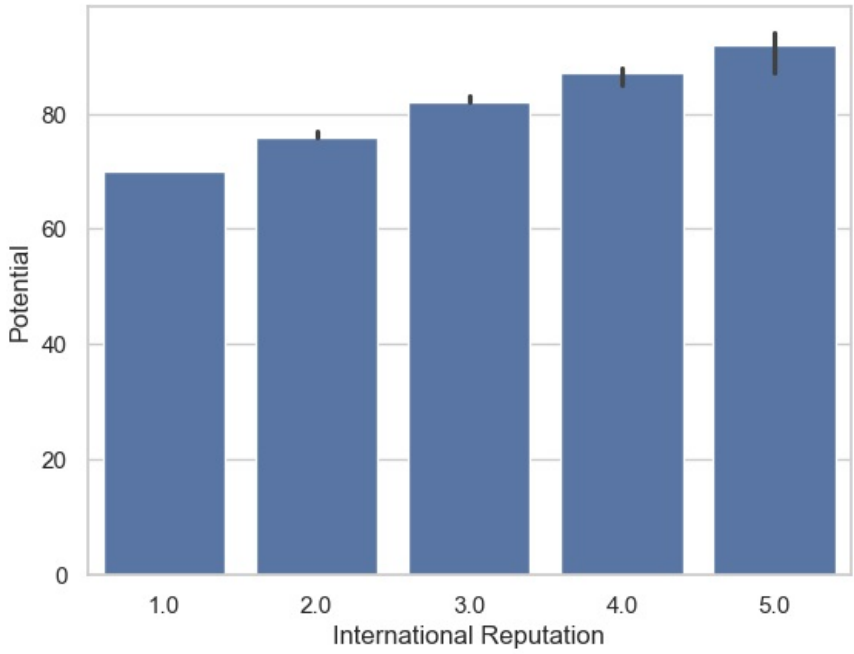
In [39]:
```
# adding hue as preferred foot
sns.barplot(x=data['International Reputation'],y=data['Potential'],hue=data['Preferred Foot'])
```

Out[39]: <Axes: xlabel='International Reputation', ylabel='Potential'>



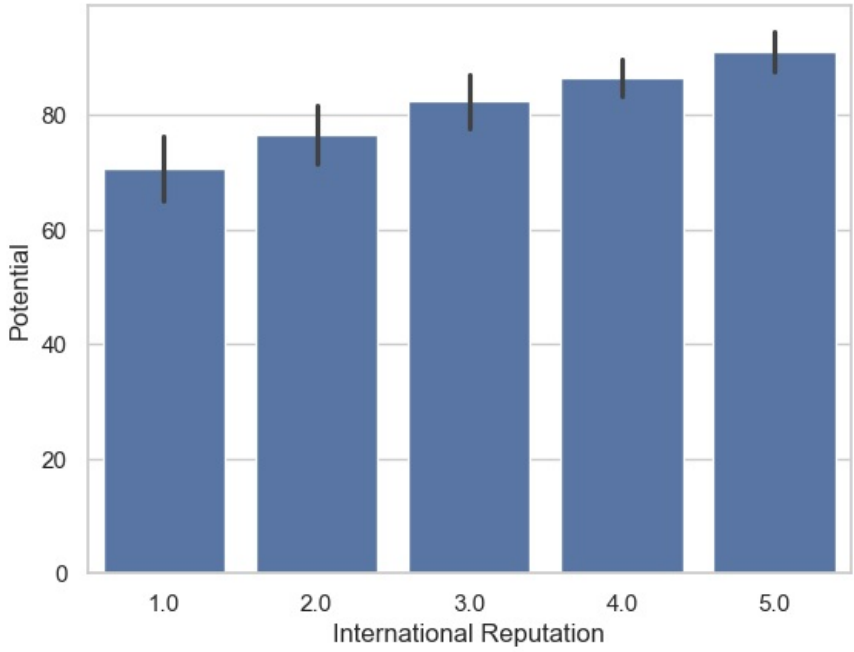
In [41]:
```
# use estimater as median
from numpy import median
sns.barplot(x=data['International Reputation'],y=data['Potential'],estimator=median)
```

Out[41]: <Axes: xlabel='International Reputation', ylabel='Potential'>



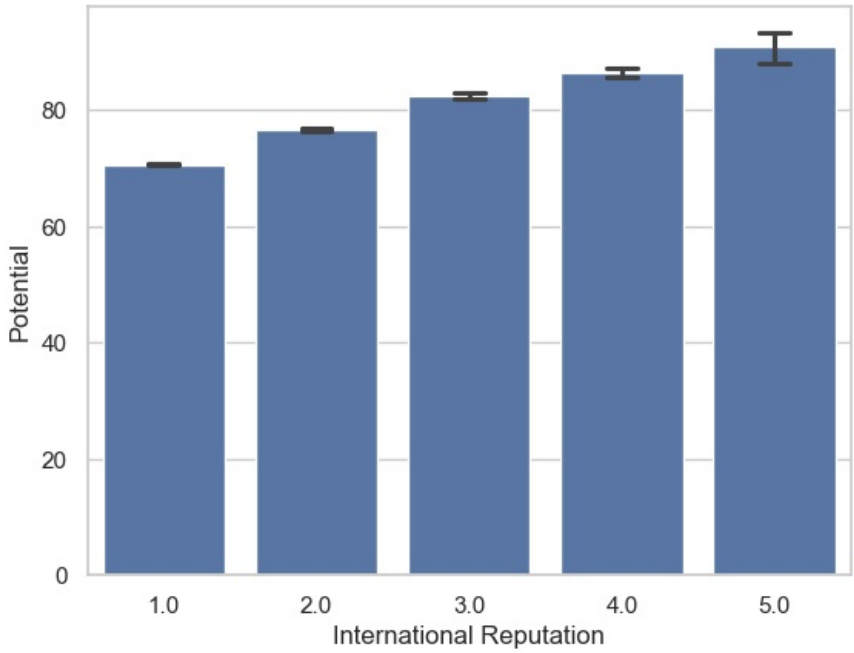
In [42]:
```
# use error bar as sd
sns.barplot(x=data['International Reputation'],y=data['Potential'],errorbar='sd')
```

Out[42]: <Axes: xlabel='International Reputation', ylabel='Potential'>



In [43]:
```
# use caps size
sns.barplot(x=data['International Reputation'],y=data['Potential'],capsize=0.2)
```

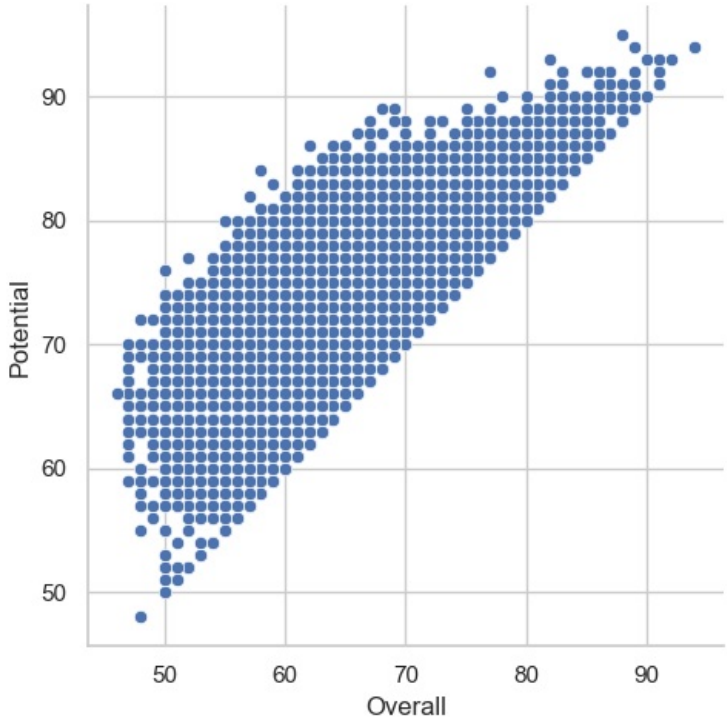
Out[43]:
```
<Axes: xlabel='International Reputation', ylabel='Potential'>
```



Rel plot

In [44]:
```
# relation plot respect to overall and potential variable
sns.relplot(data,x=data['Overall'],y=data['Potential'])
```

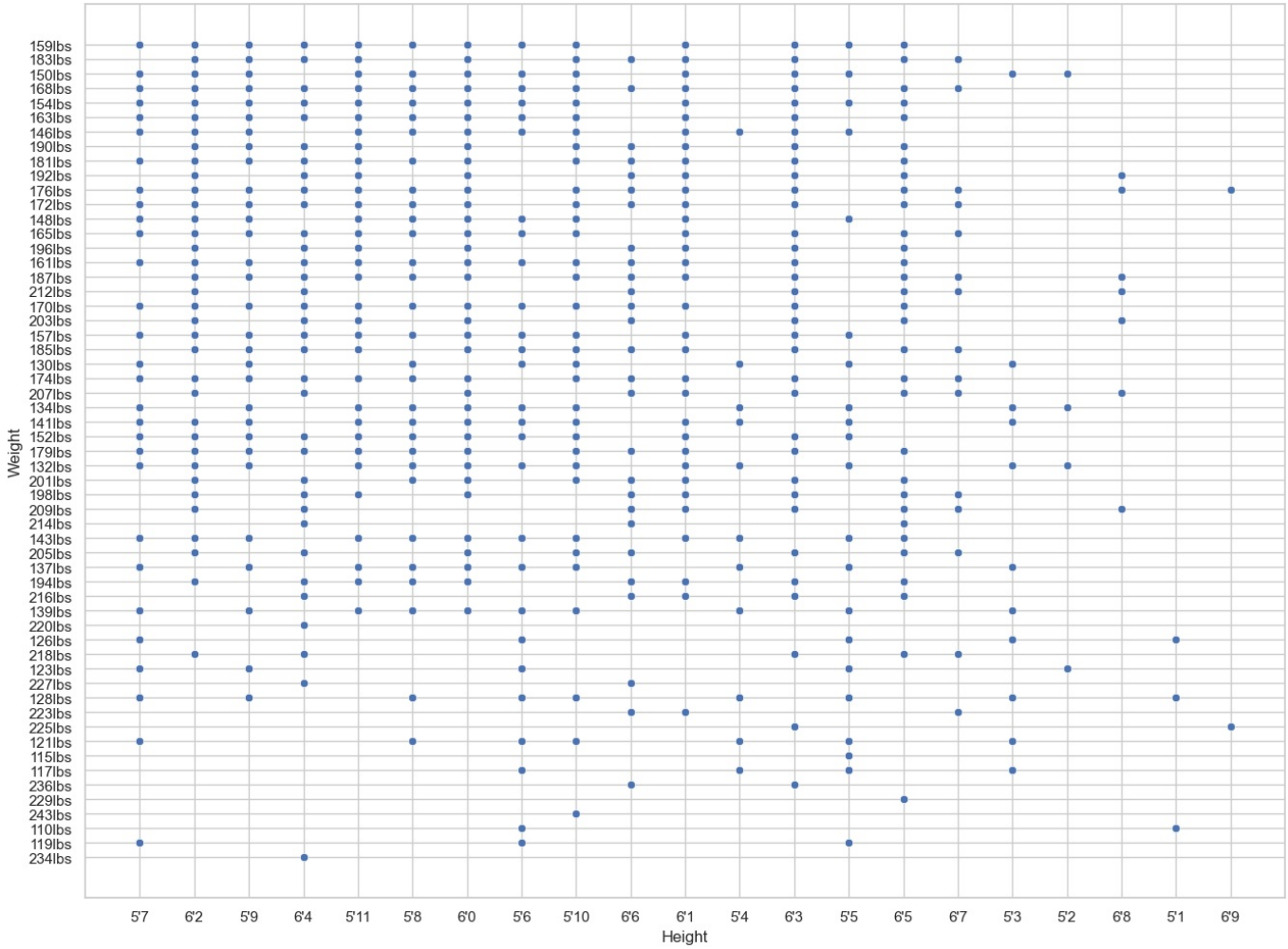
Out[44]: <seaborn.axisgrid.FacetGrid at 0x1d91fdbeba0>



scatter plot

In [45]:
```
plt.subplots(figsize=(16,12)) # chage the plot size

# scatter plot respect to height and weight
sns.scatterplot(x=data['Height'],y=data['Weight'])
```

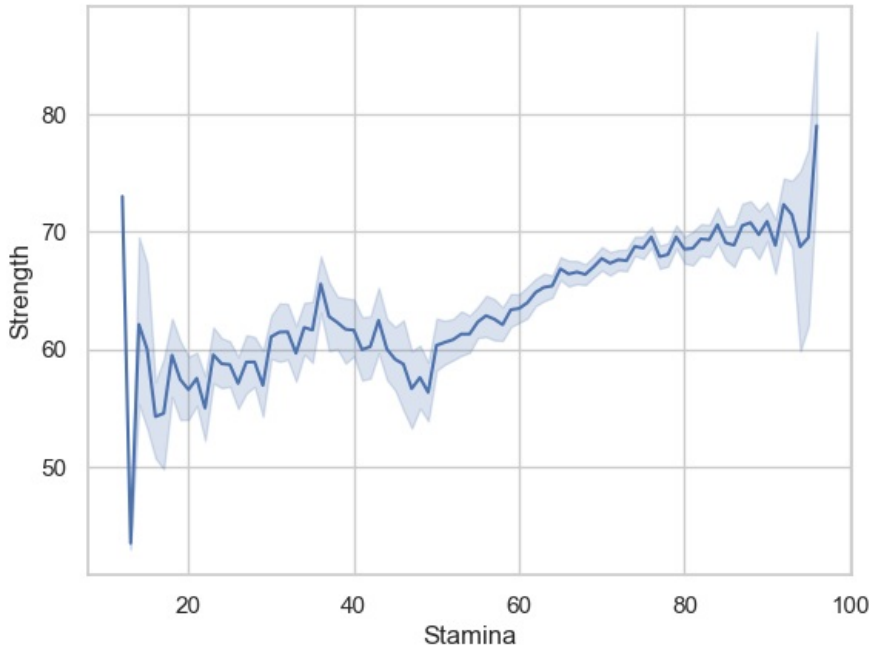
Out[45]: <Axes: xlabel='Height', ylabel='Weight'>



line plot

In [46]:
```
# line plot respect tt stamina and strength
sns.lineplot(x=data['Stamina'],y=data['Strength'])
```

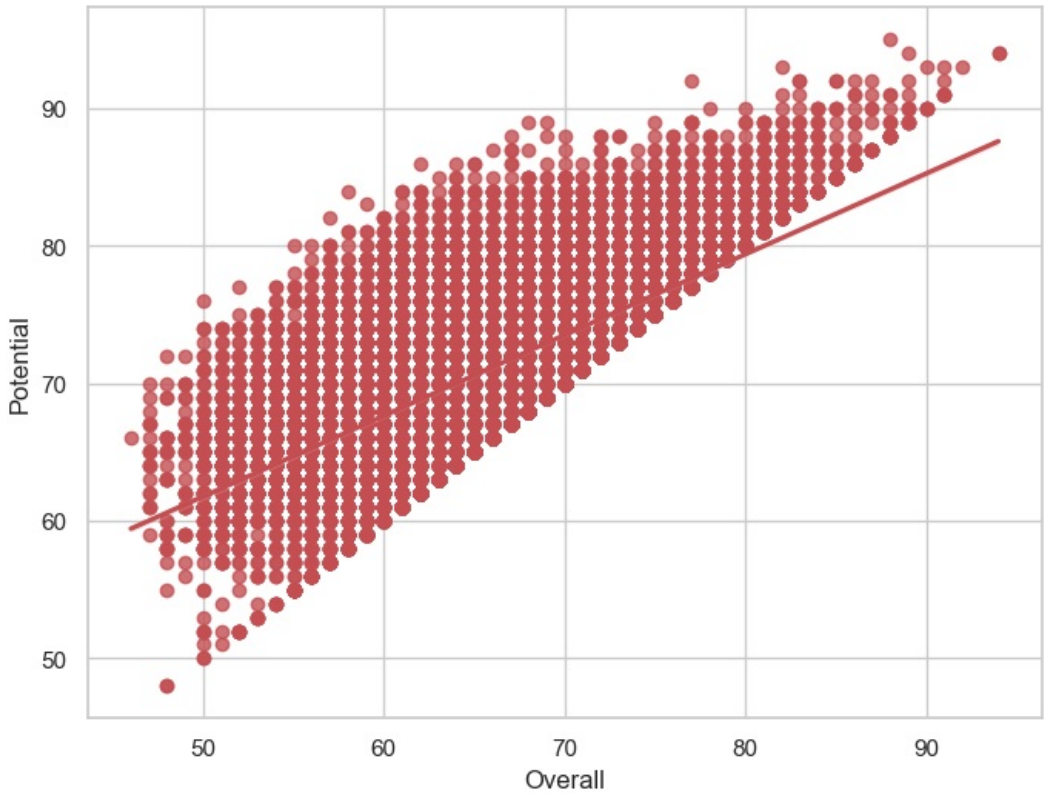
Out[46]:
```
<Axes: xlabel='Stamina', ylabel='Strength'>
```



reg plot

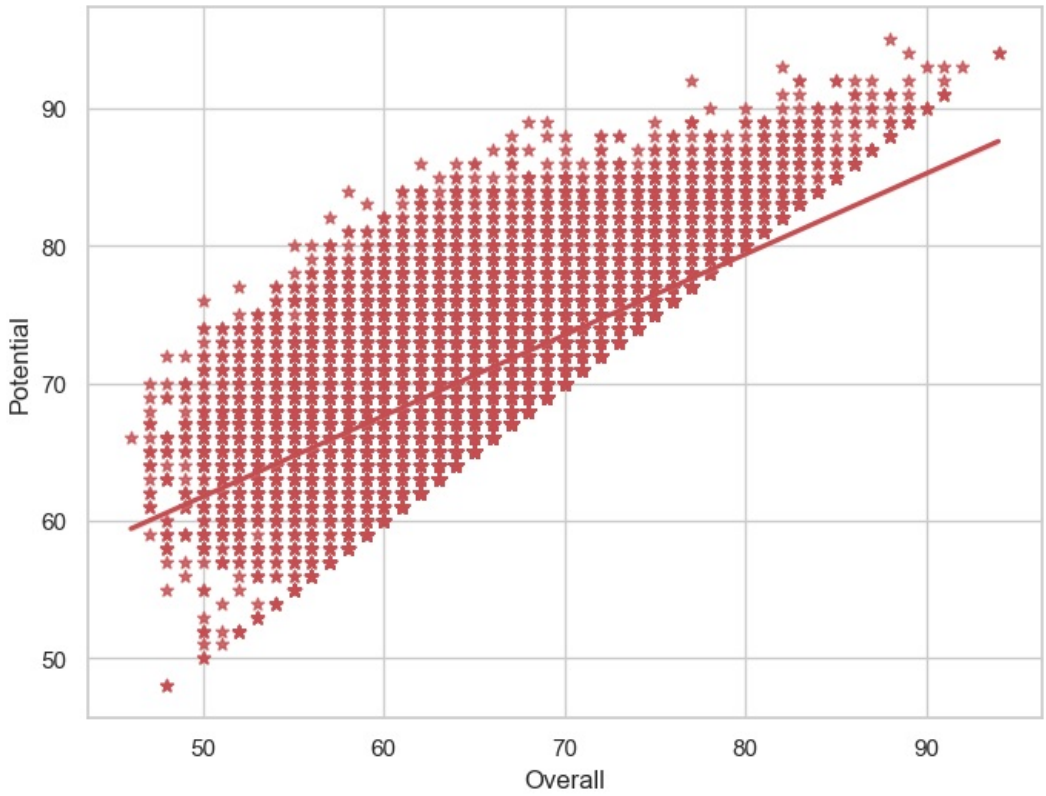
In [47]:
```
plt.subplots(figsize=(8,6))

# regression plot respect to overall, potential with red color
sns.regplot(x=data['Overall'],y=data['Potential'],color='r')
```

Out[47]:
```
<Axes: xlabel='Overall', ylabel='Potential'>
```



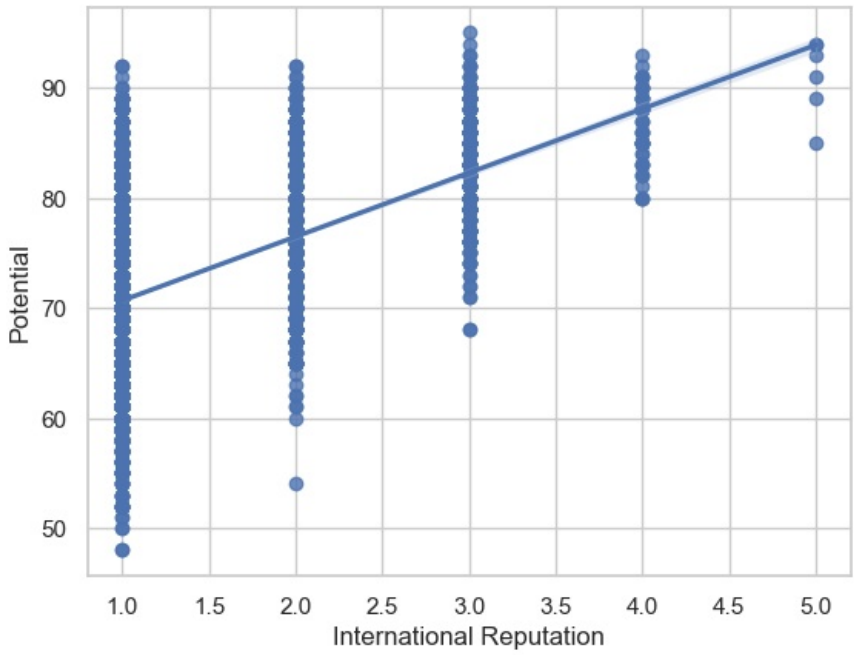
In [48]:
```
plt.subplots(figsize=(8,6))
# adding markers on it
sns.regplot(x=data['Overall'],y=data['Potential'],color='r',marker='*')
```

Out[48]:
```
<Axes: xlabel='Overall', ylabel='Potential'>
```



In [49]:
```
sns.regplot(x=data['International Reputation'],y=data['Potential'])
```

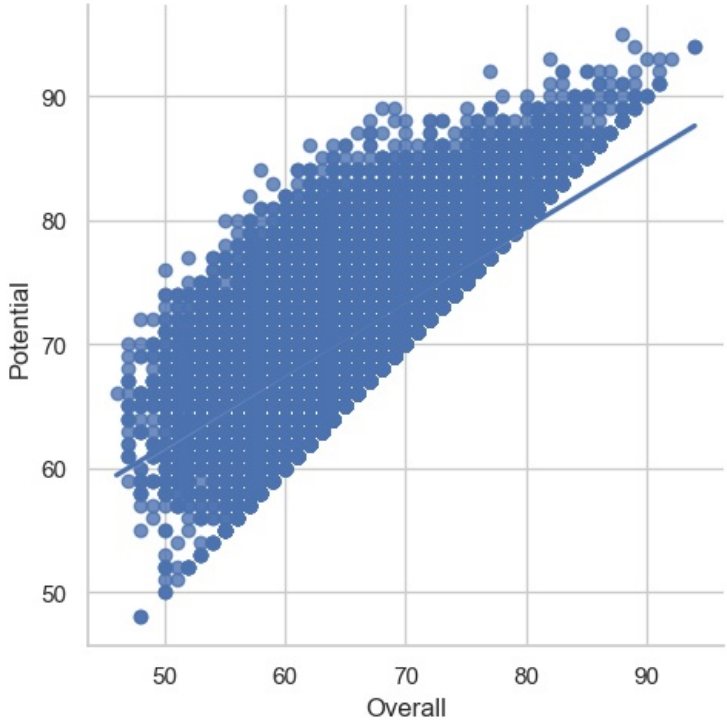
Out[49]:
```
<Axes: xlabel='International Reputation', ylabel='Potential'>
```



lmplot

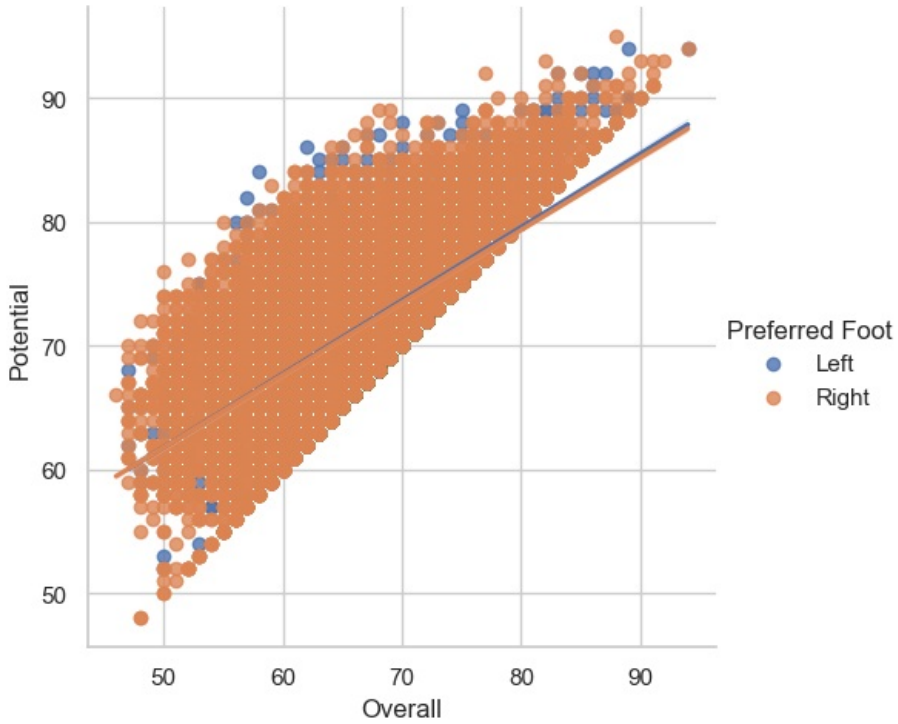
In [50]:
```
# lmplot respect to oc=verall and potential
sns.lmplot(data,x='Overall',y='Potential')
```

Out[50]:
```
<seaborn.axisgrid.FacetGrid at 0x1d91ed6a2d0>
```



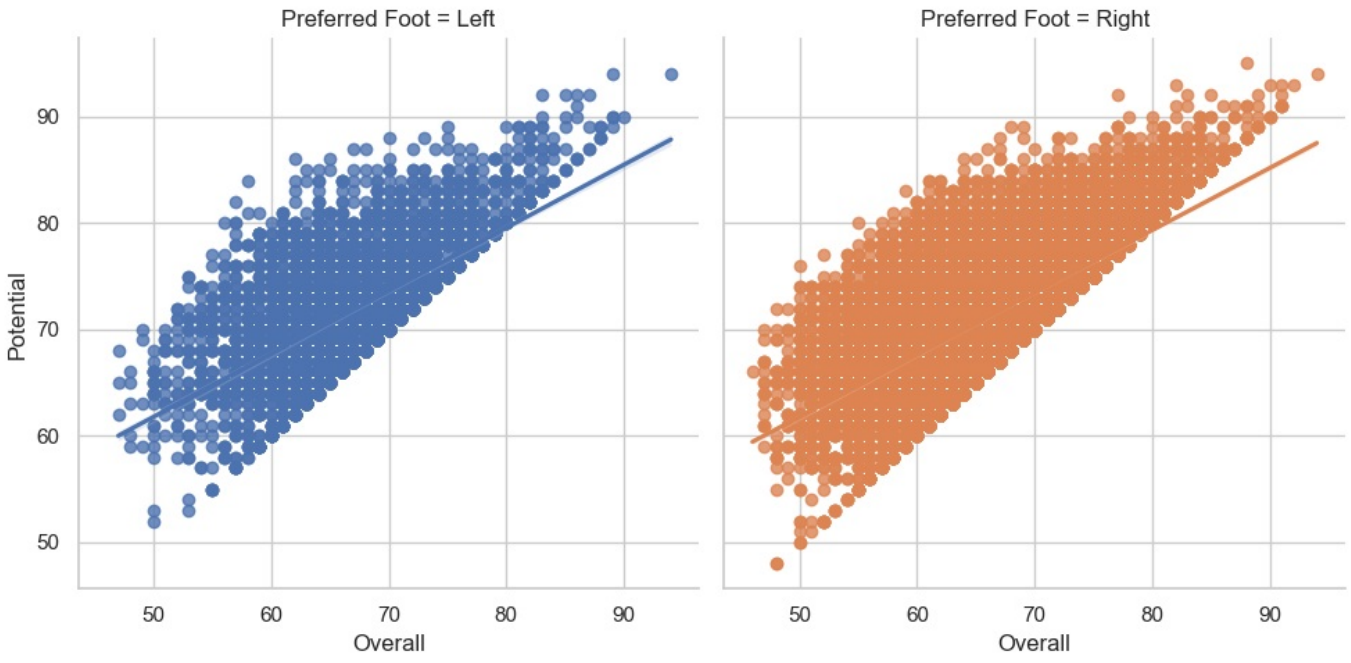
In [51]:
```
# hue as preferred foot
sns.lmplot(data,x='Overall',y='Potential',hue='Preferred Foot')
```

Out[51]: <seaborn.axisgrid.FacetGrid at 0x1d91f898740>



In [53]:
```
# creat graph according to preferred foot
sns.lmplot(data,x='Overall',y='Potential',col='Preferred Foot',hue='Preferred Foot')
```

Out[53]: <seaborn.axisgrid.FacetGrid at 0x1d91fca7950>

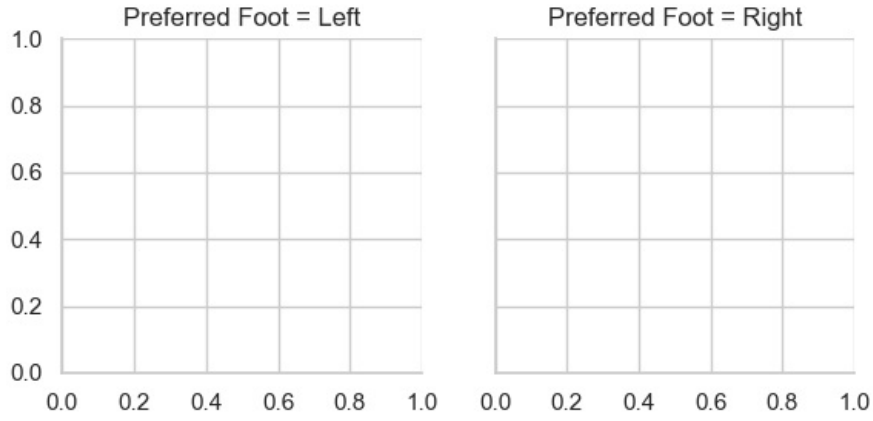


In [ ]:

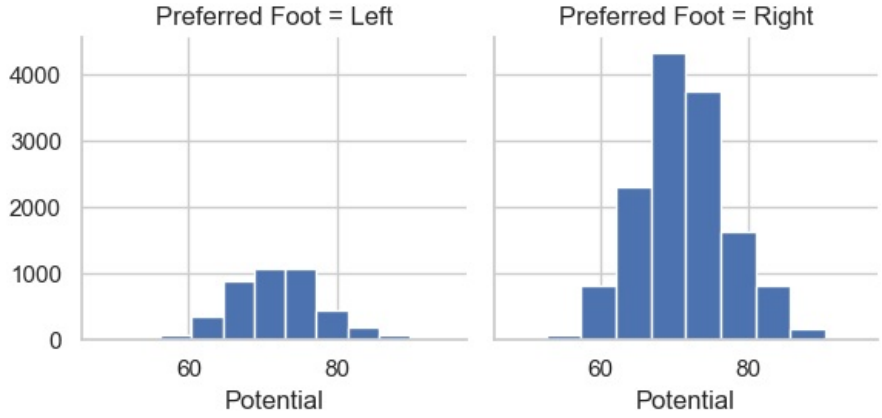
facetgrade plot

In [54]:
```
# facetgrade acording to preferred foot
sns.FacetGrid(data,col='Preferred Foot')
```

Out[54]:
```
<seaborn.axisgrid.FacetGrid at 0x1d91fcae1b0>
```

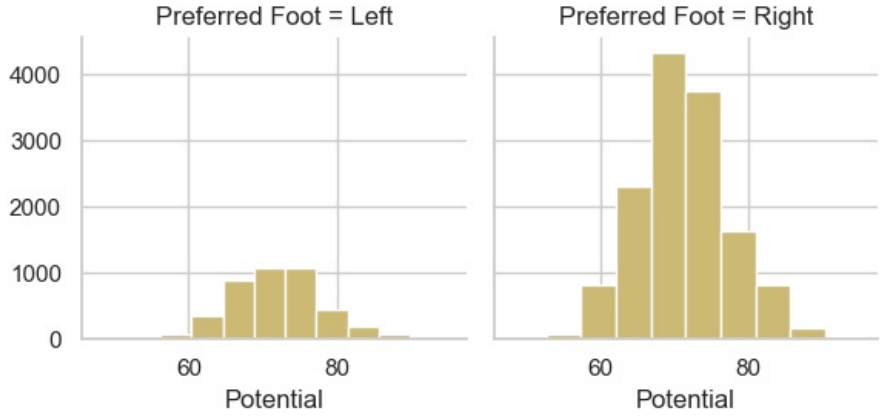


In [55]:
```
a = sns.FacetGrid(data,col='Preferred Foot')
# plot histogrph plot of potential
a =a.map(plt.hist,'Potential')
```

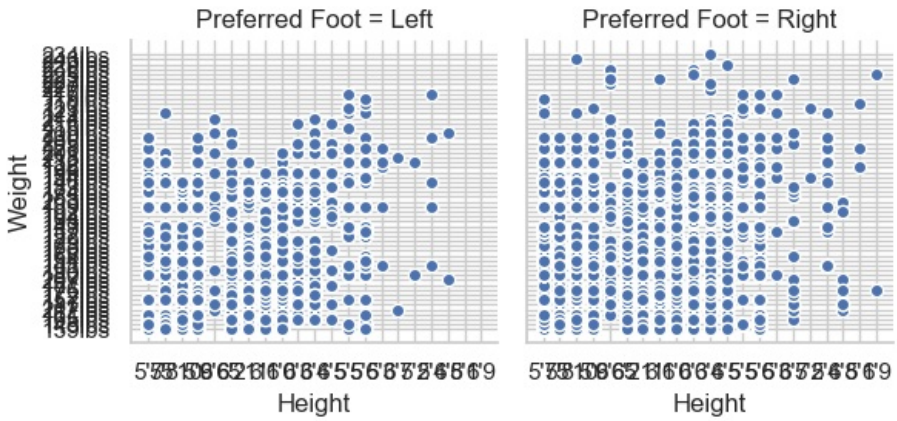


In [56]:
```
a = sns.FacetGrid(data,col='Preferred Foot')
# histogrph of potential attribute color yellow, bins 10
```

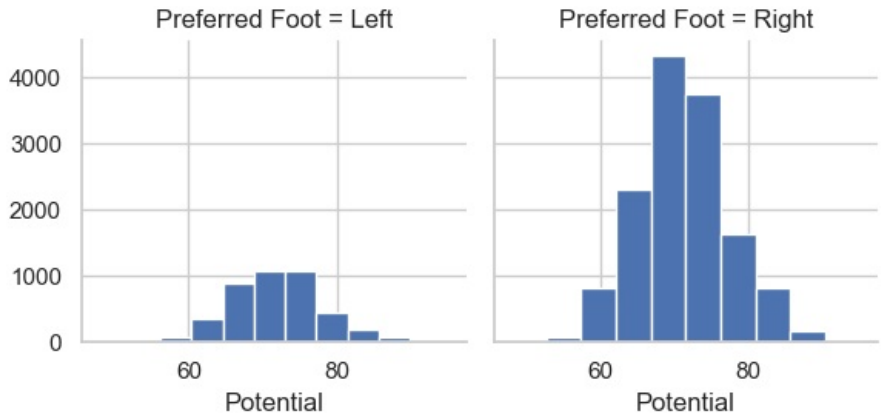
```
a =a.map(plt.hist,'Potential',color='y',bins=10)
```



In [57]:
```
a = sns.FacetGrid(data,col='Preferred Foot')
# scatter plot respect to height and weight, edgecolor white, add legeand
a = a.map(plt.scatter,'Height','Weight',edgecolor='w').add_legend()
```



In [58]:
```
a = sns.FacetGrid(data,col='Preferred Foot')
a= a.map(plt.hist,'Potential')
```

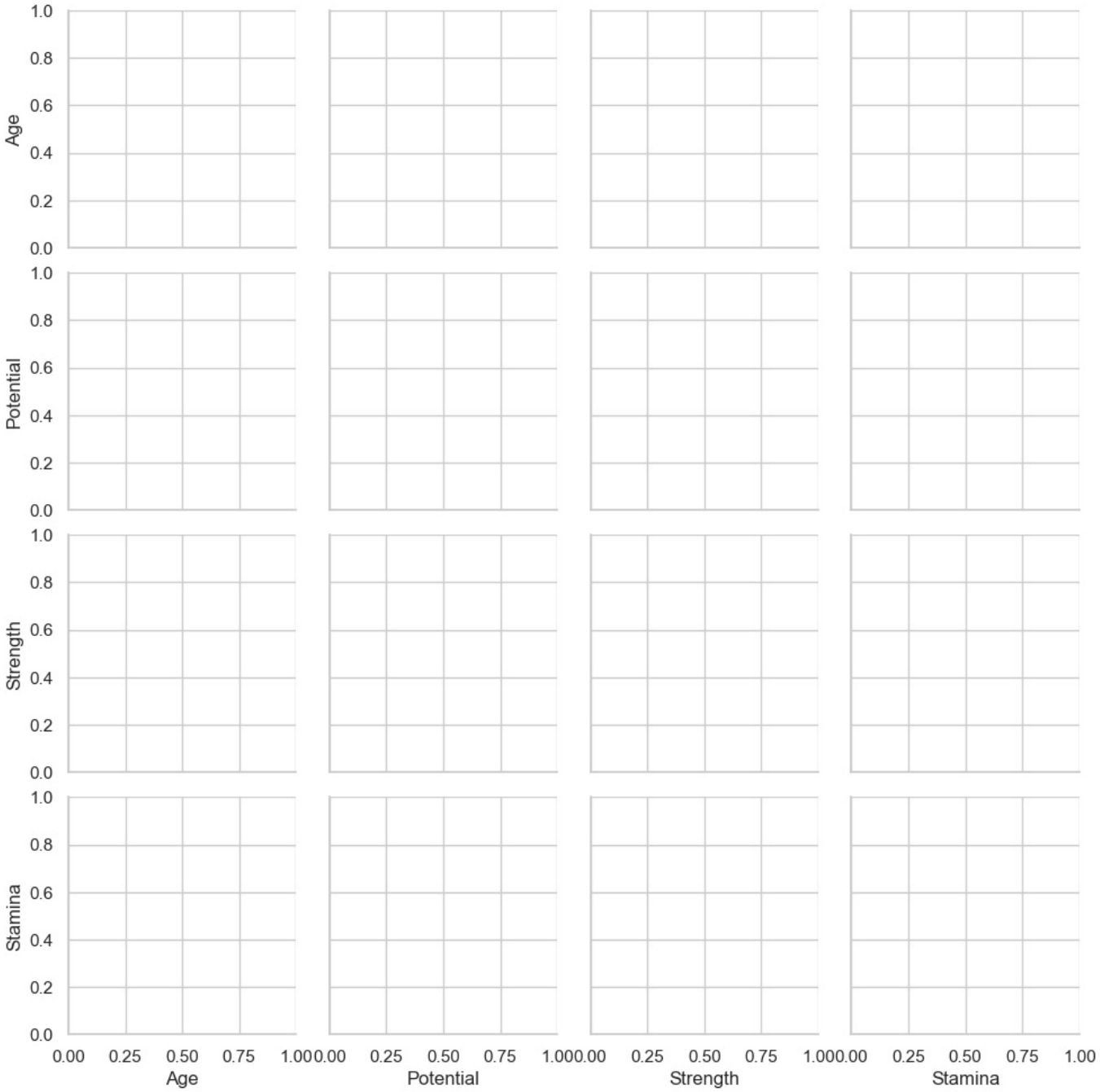


pairgrid

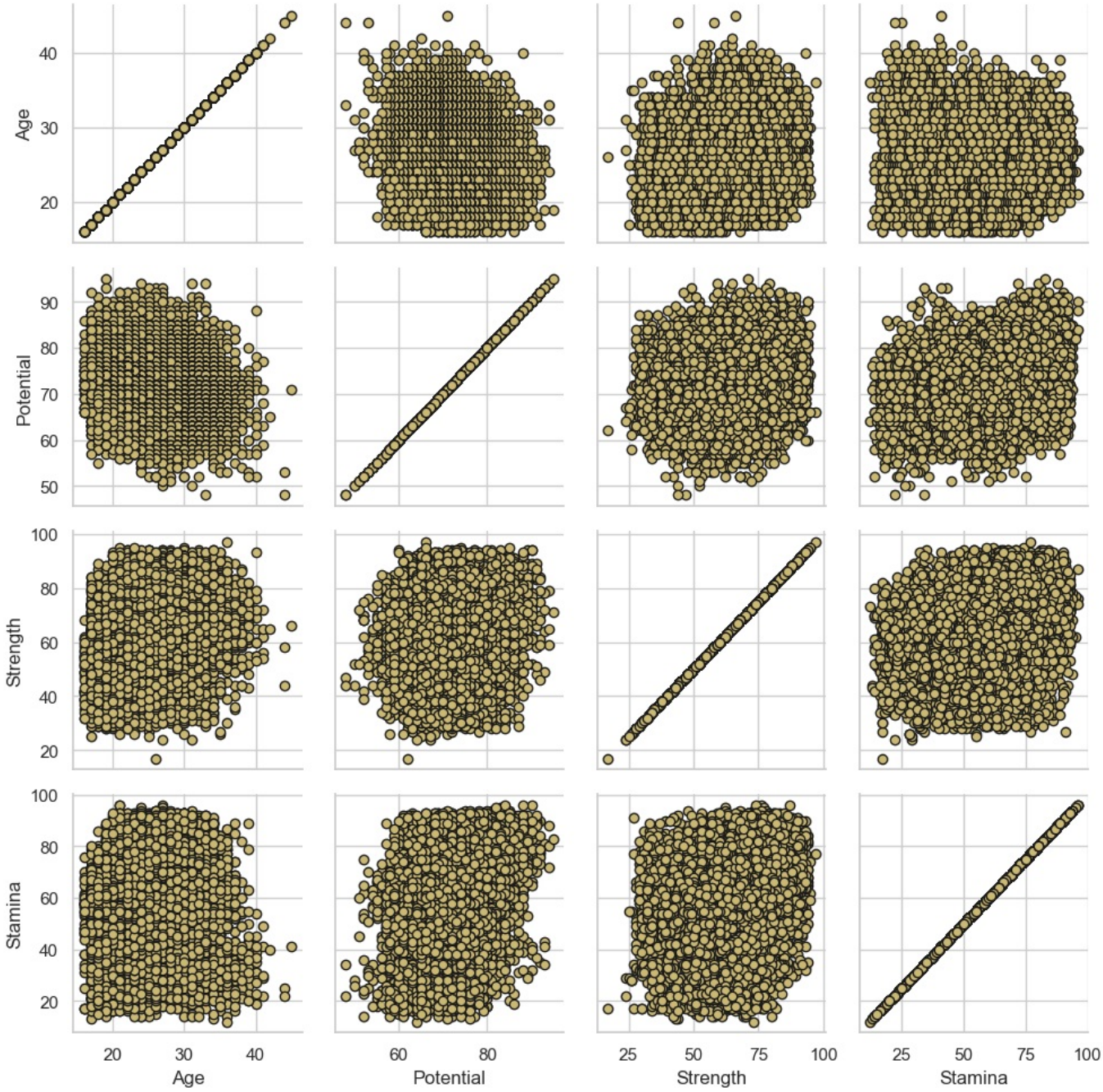
In [59]:
```
# make pair from the whole data frame
data1=data[['Age', 'Potential', 'Strength', 'Stamina', 'Preferred Foot']]
```

In [60]:
```
# pair grid plot view
sns.PairGrid(data1)
```

Out[60]:
```
<seaborn.axisgrid.PairGrid at 0x1d922e7bfe0>
```

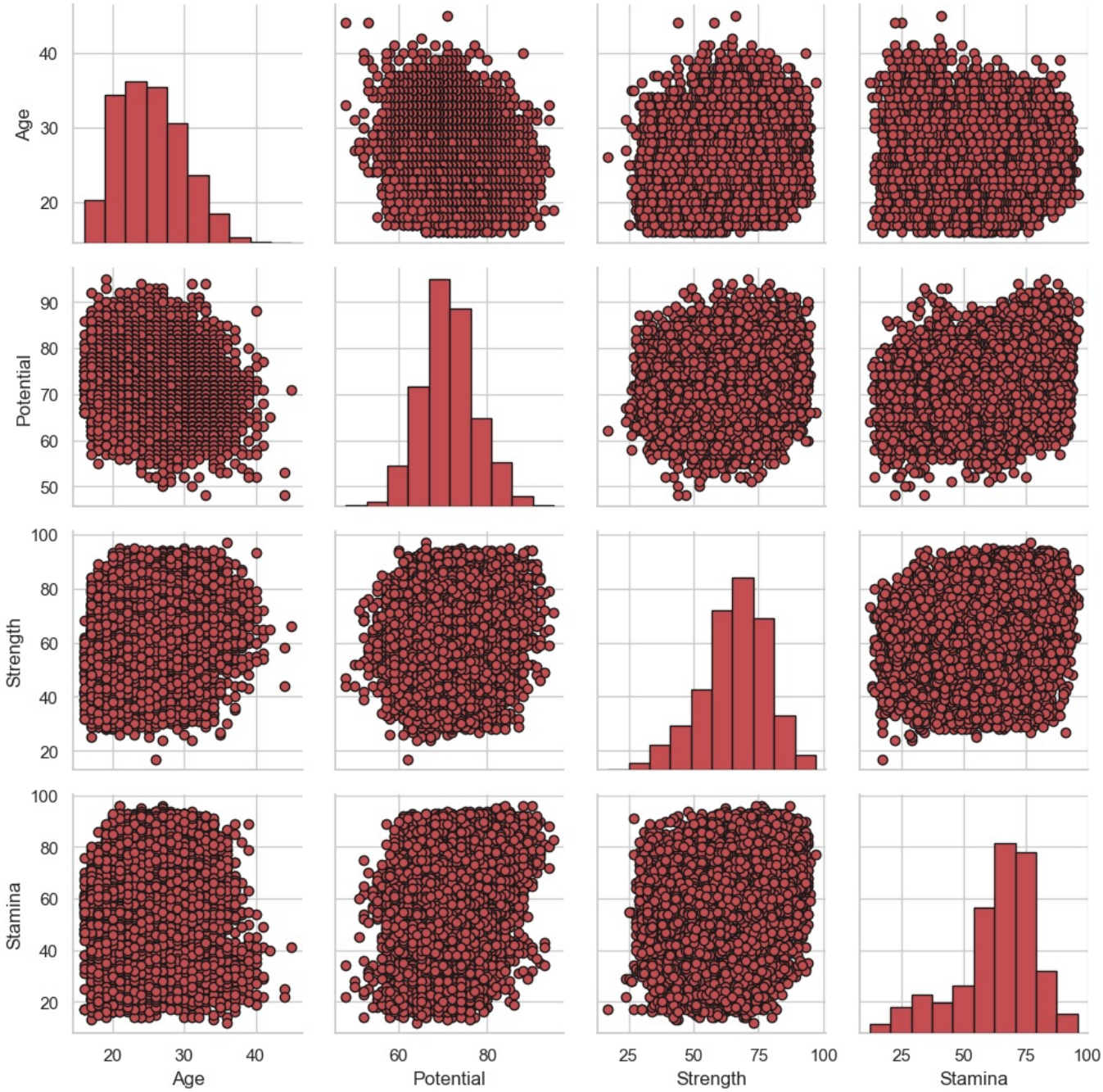


In [61]:
```
a = sns.PairGrid(data1)
# map scatter plot with the pair data, color yellow, edge color black
a=a.map(plt.scatter,color='y',edgecolor='k')
```



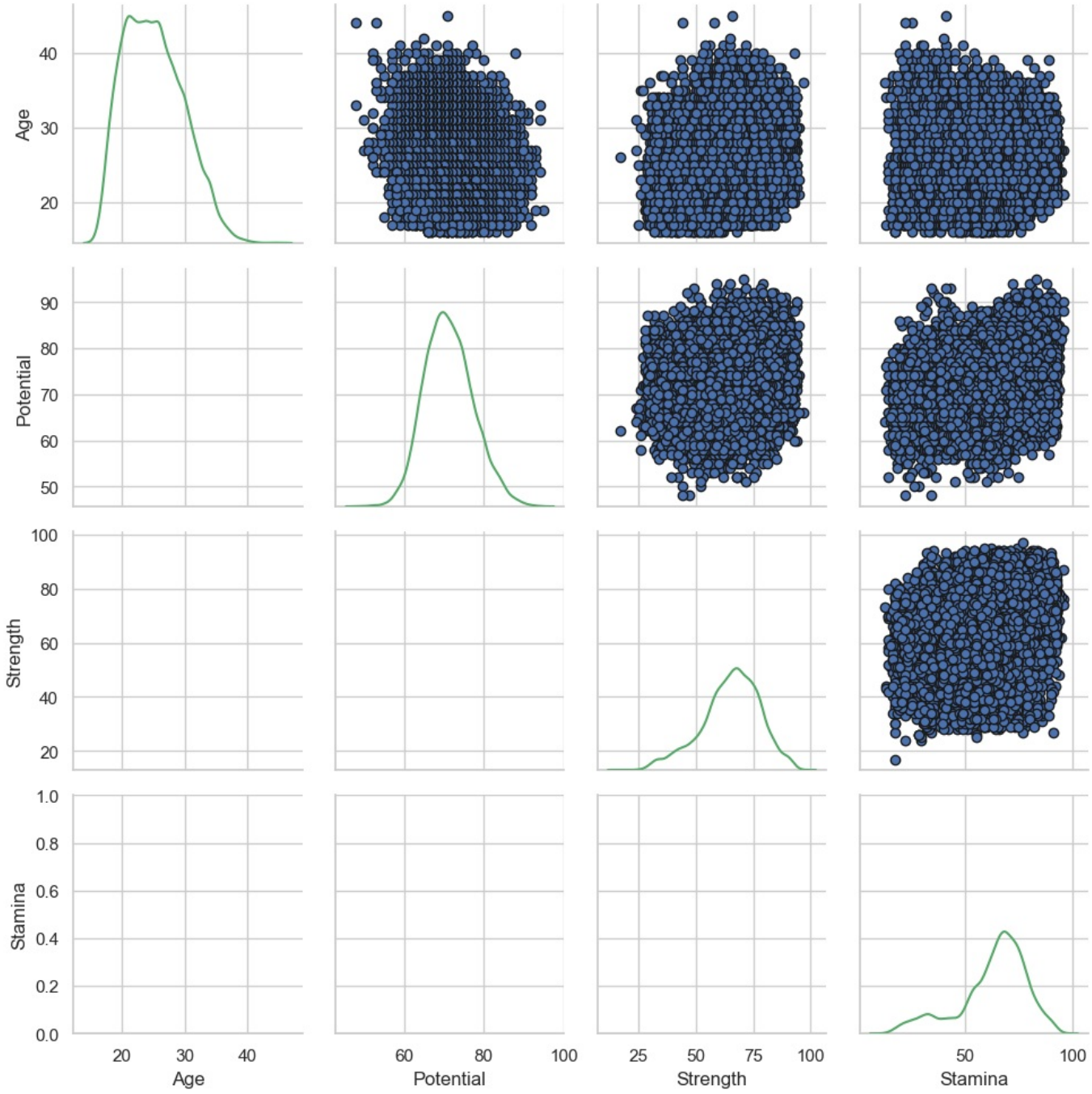
In [62]:
```
a = sns.PairGrid(data1)
#  diagonal matrices histogram,edgecolor black,color red
a=a.map_diag(plt.hist,edgecolor='k',color='r')
# off_digonal matrix are scatter plot edge color black,color red
a.map_offdiag(plt.scatter,edgecolor='k',color='r')
```

Out[62]: <seaborn.axisgrid.PairGrid at 0x1d91b312930>



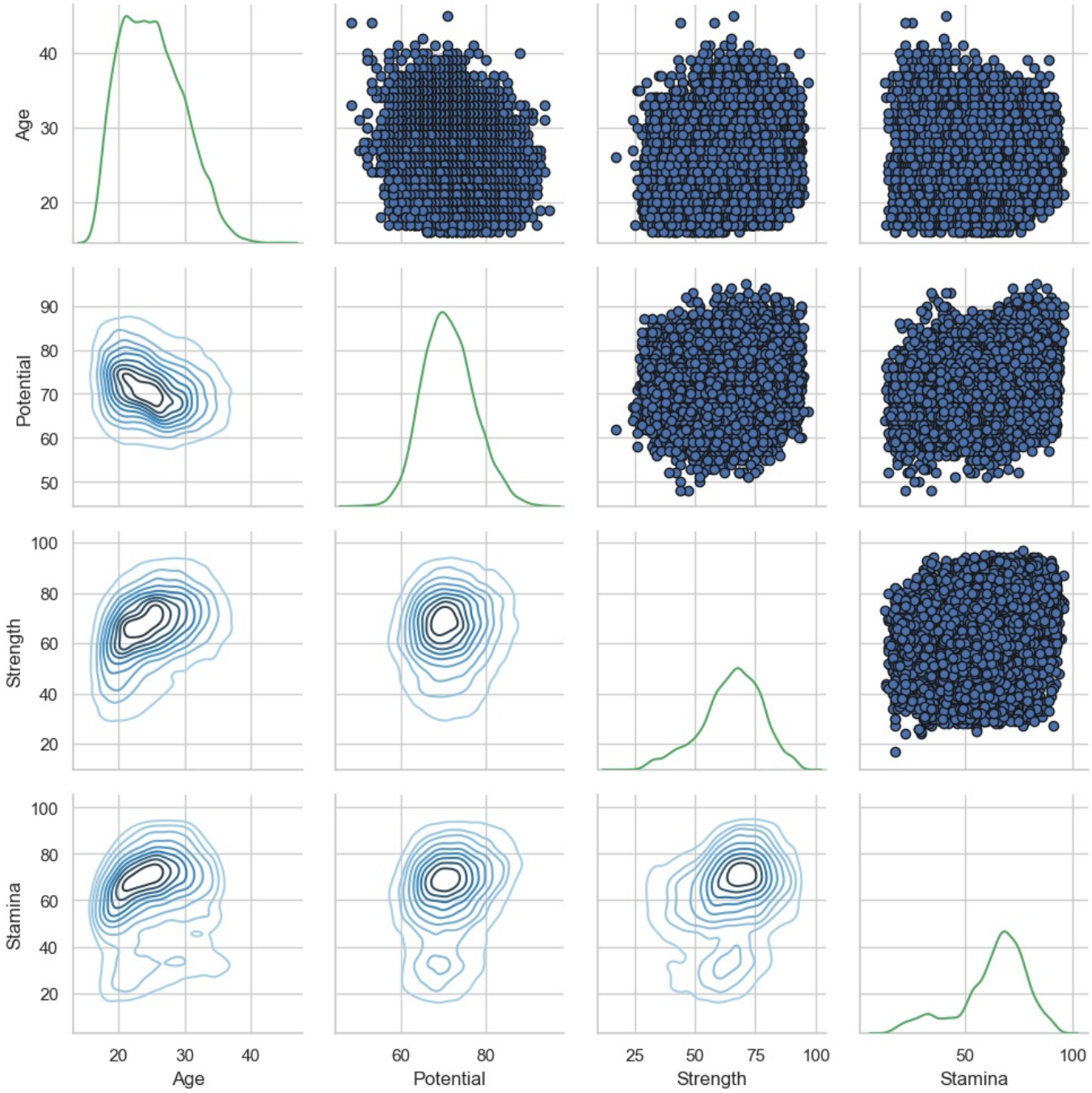
In [63]:
```
a=sns.PairGrid(data1)
# diagonal matrix are kdeplot, color green
a.map_diag(sns.kdeplot,color='g')
# upper to digonal matrix are scatter plot with edgecolor black
a.map_upper(plt.scatter,edgecolor='k')
```

Out[63]:
```
<seaborn.axisgrid.PairGrid at 0x1d9267db890>
```



In [64]:
```
a=sns.PairGrid(data1)
# digonal matrix kde plot color green
a.map_diag(sns.kdeplot,color='g')
# upper diagonal matrix scatter and edgecolor black
a.map_upper(plt.scatter,edgecolor='k')
# lower diagonal matrix are kde plot cmap blues
a.map_lower(sns.kdeplot,cmap='Blues_d')
```

Out[64]:
```
<seaborn.axisgrid.PairGrid at 0x1d9261b9190>
```

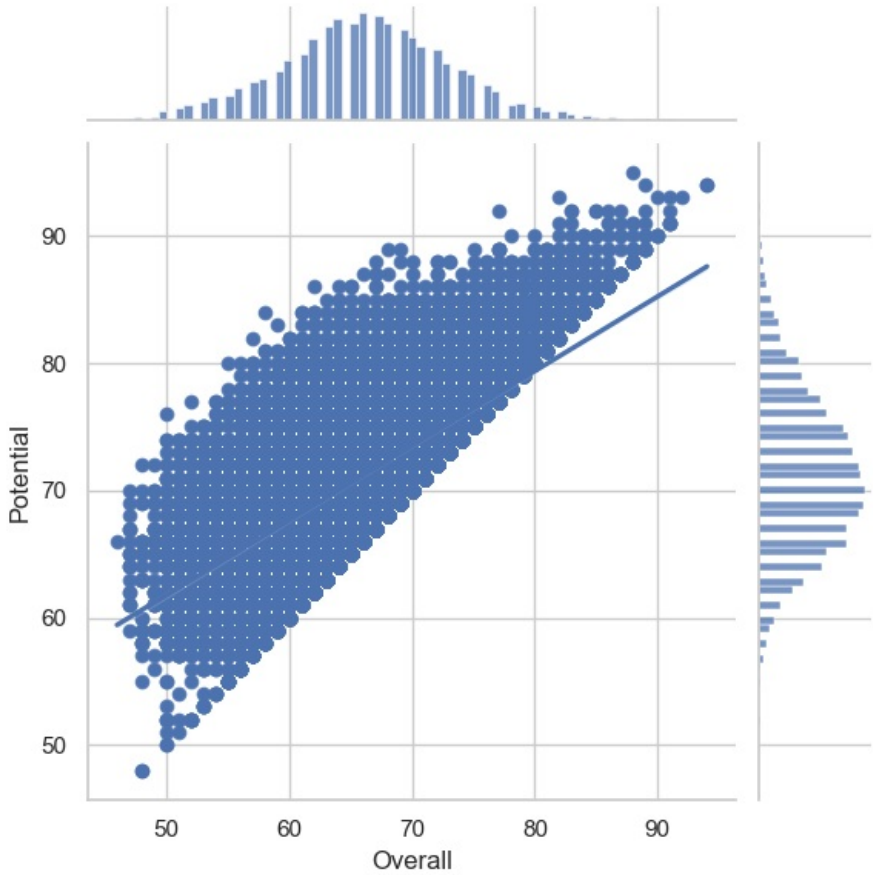


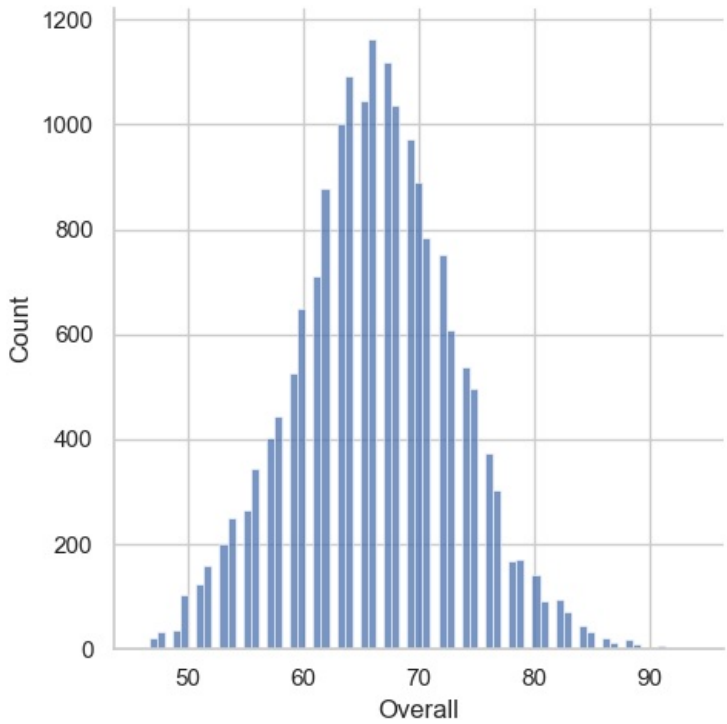
jointplot

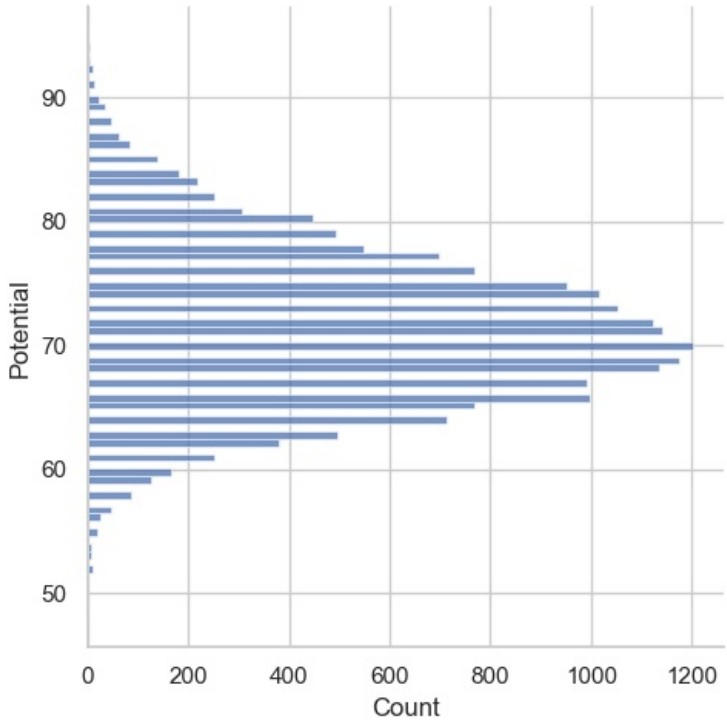
In [70]:
```
a=sns.jointplot(data,x="Overall",y='Potential')
a.plot(sns.regplot,sns.displot)
```

```
C:\Users\sunil\anaconda3\Lib\site-packages\seaborn\distributions.py:2125: UserWarning: `displot` is a figure-lev
el function and does not accept the ax= parameter. You may wish to try histplot.
  warnings.warn(msg, UserWarning)
C:\Users\sunil\anaconda3\Lib\site-packages\seaborn\distributions.py:2125: UserWarning: `displot` is a figure-lev
el function and does not accept the ax= parameter. You may wish to try histplot.
  warnings.warn(msg, UserWarning)
```

Out[70]: <seaborn.axisgrid.JointGrid at 0x1d9269636e0>

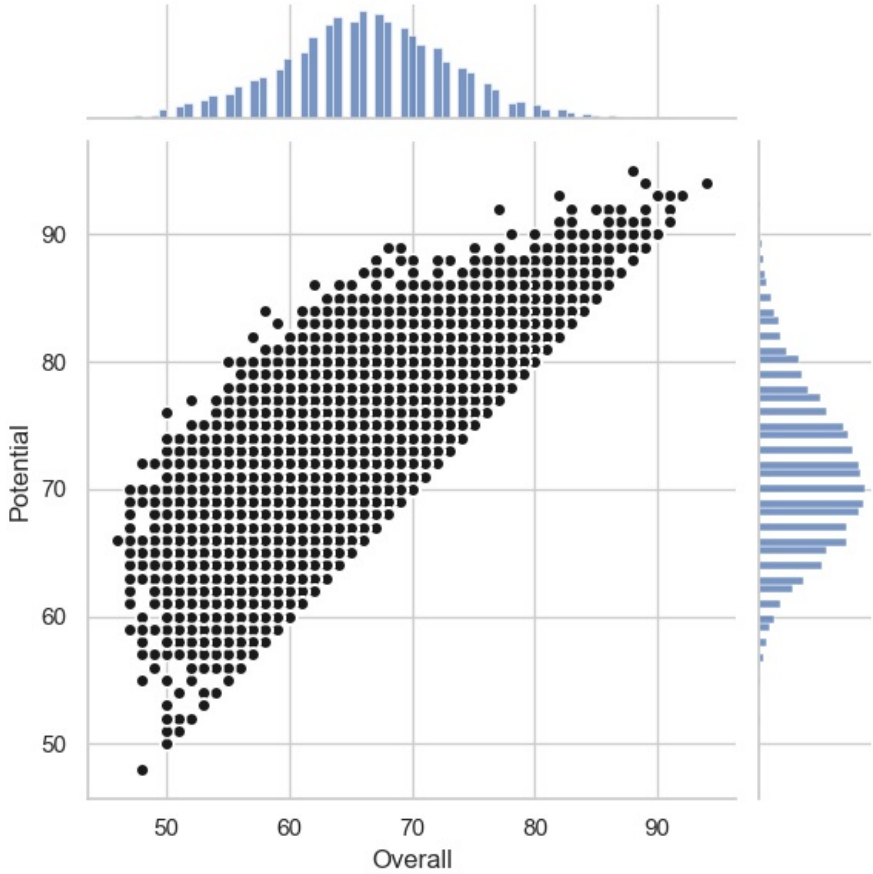






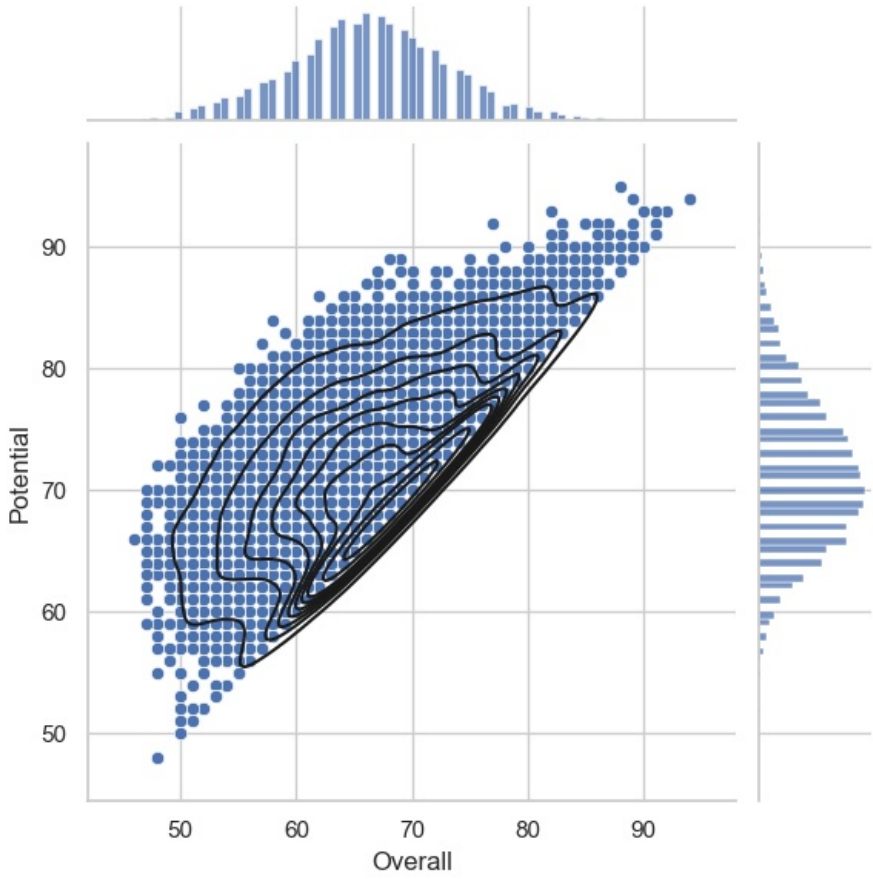
In [71]:
```
a=sns.jointplot(data,x="Overall",y='Potential')
a.plot_joint(plt.scatter,edgecolor='w',color='k')
```

Out[71]: <seaborn.axisgrid.JointGrid at 0x1d929b70aa0>



In [81]:
```
a=sns.jointplot(data,x="Overall",y='Potential')
a.plot_joint(sns.kdeplot, color='k')
```

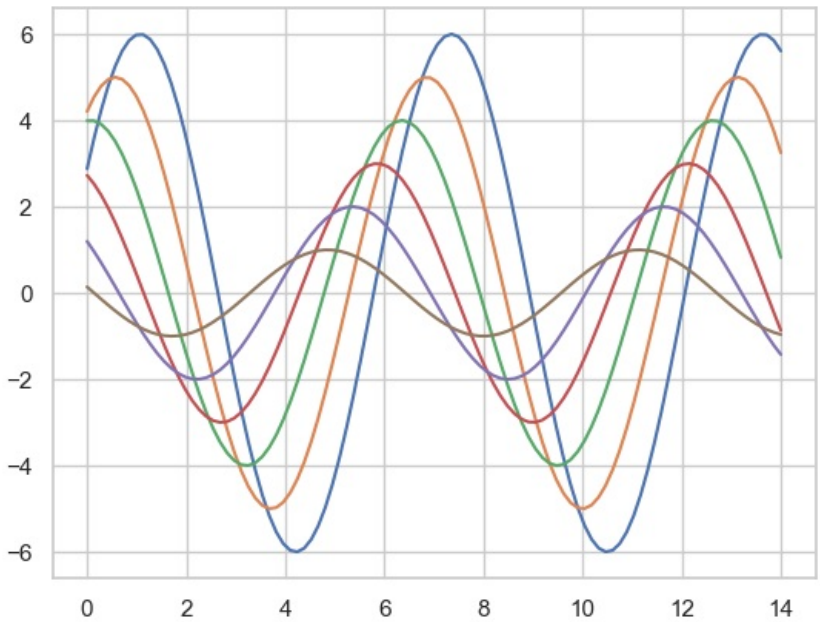
Out[81]: <seaborn.axisgrid.JointGrid at 0x1d92f48f9e0>



seaborn figure style

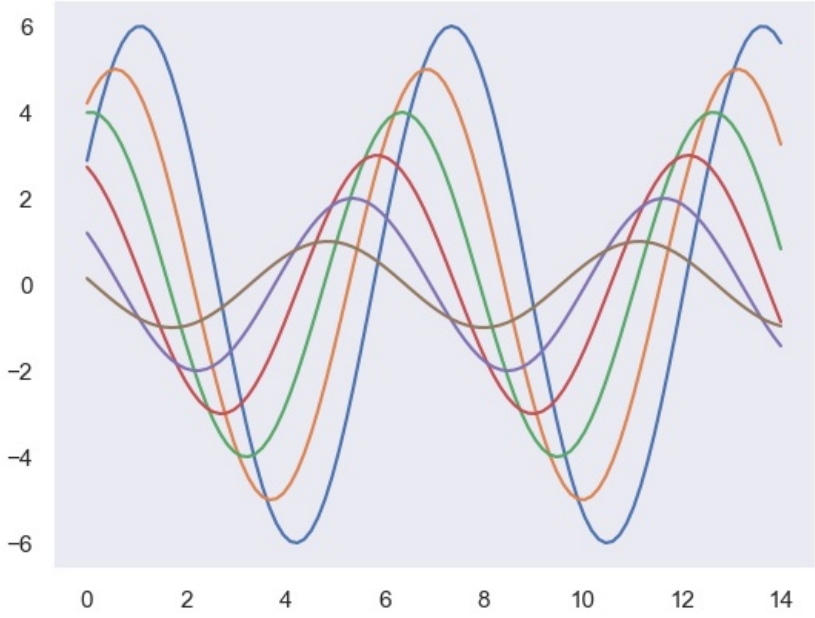
In [82]:
```
def sinplot(flip=1):
    x = np.linspace(0, 14, 100)
    for i in range(1, 7):
        plt.plot(x, np.sin(x + i * .5) * (7 - i) * flip)
```

In [85]:
```
sinplot()
```

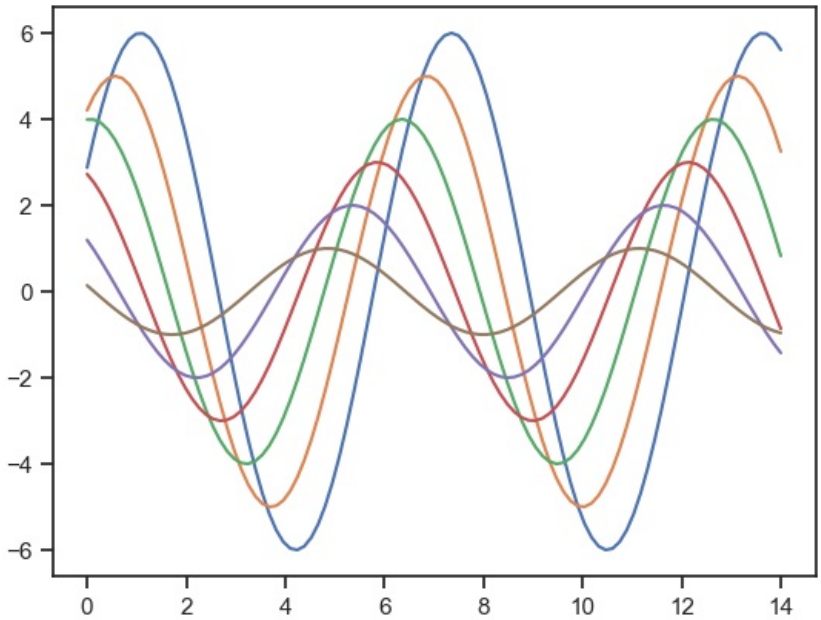


In [88]:
```
sns.set_style('dark')
```

```
sinplot()
```



In [89]:
```
sns.set_style('ticks')
sinplot()
```



Completed

In [ ]:

Loading [MathJax]/extensions/Safe.js