statistical-analysis

August 10, 2024

[47]:
```
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
import plotly.express as px

import warnings
warnings.filterwarnings('ignore')

pd.set_option('display.max_columns',None)
```

[10]:
```
df= pd.read_excel("Game.xlsx")
```

[11]:
```
df.head()
```

[11]:
```
    ID          Name  Age                                      Photo  \
0   16   Luis García   37   https://cdn.sofifa.org/players/4/19/16.png
1   41       Iniesta   34   https://cdn.sofifa.org/players/4/19/41.png
2   80  E. Belözoğlu   37   https://cdn.sofifa.org/players/4/19/80.png
3  164      G. Pinzi   37  https://cdn.sofifa.org/players/4/19/164.png
4  657    D. Vaughan   35  https://cdn.sofifa.org/players/4/19/657.png

  Nationality                                 Flag  Overall  Potential  \
0       Spain  https://cdn.sofifa.org/flags/45.png       71         71
1       Spain  https://cdn.sofifa.org/flags/45.png       86         86
2      Turkey  https://cdn.sofifa.org/flags/48.png       79         79
3       Italy  https://cdn.sofifa.org/flags/27.png       70         70
4       Wales  https://cdn.sofifa.org/flags/50.png       66         66

                   Club                                      Club Logo  \
0             KAS Eupen    https://cdn.sofifa.org/teams/2/light/2013.png
1           Vissel Kobe  https://cdn.sofifa.org/teams/2/light/101146.png
2  Medipol Başakşehir FK  https://cdn.sofifa.org/teams/2/light/101014.png
3                Padova  https://cdn.sofifa.org/teams/2/light/110912.png
4          Notts County    https://cdn.sofifa.org/teams/2/light/1937.png

    Value   Wage  Special Preferred Foot  International Reputation  Weak Foot  \
```

0	€750K	€6K	1906	Right	1.0	4.0
1	€21.5M	€21K	2058	Right	4.0	4.0
2	€4M	€23K	2047	Left	2.0	4.0
3	€240K	€2K	1882	Right	2.0	3.0
4	€150K	€4K	1781	Left	1.0	3.0

	Skill Moves	Work Rate	Body Type	Real Face	Position	Jersey Number	\
0	3.0	Medium/ Medium	Lean	No	RCM	10.0	
1	4.0	High/ Medium	Normal	Yes	LF	8.0	
2	4.0	Medium/ Medium	Stocky	No	CM	5.0	
3	3.0	Low/ High	Normal	No	LCM	11.0	
4	2.0	Medium/ High	Stocky	No	CDM	8.0	

	Joined	Loaned From	Contract Valid Until	Height	Weight	LS	ST	\
0	Jul 19, 2014	NaN	2019	5'10	143lbs	66+2	66+2	
1	Jul 16, 2018	NaN	2021	5'7	150lbs	74+3	74+3	
2	Jul 9, 2015	NaN	2019	5'7	159lbs	67+2	67+2	
3	Aug 31, 2017	NaN	2019	5'11	168lbs	63+2	63+2	
4	Jul 6, 2018	NaN	2019	5'6	154lbs	59+2	59+2	

	RS	LW	LF	CF	RF	RW	LAM	CAM	RAM	LM	LCM	CM	\
0	66+2	67+2	68+2	68+2	68+2	67+2	69+2	69+2	69+2	67+2	69+2	69+2	
1	74+3	82+3	81+3	81+3	81+3	82+3	85+3	85+3	85+3	82+3	83+3	83+3	
2	67+2	72+2	72+2	72+2	72+2	72+2	76+2	76+2	76+2	73+2	78+2	78+2	
3	63+2	64+2	64+2	64+2	64+2	64+2	65+2	65+2	65+2	64+2	66+2	66+2	
4	59+2	59+2	60+2	60+2	60+2	59+2	62+2	62+2	62+2	59+2	65+2	65+2	

	RCM	RM	LWB	LDM	CDM	RDM	RWB	LB	LCB	CB	RCB	RB	\
0	69+2	67+2	62+2	65+2	65+2	65+2	62+2	59+2	59+2	59+2	59+2	59+2	
1	83+3	82+3	71+3	73+3	73+3	73+3	71+3	68+3	63+3	63+3	63+3	68+3	
2	78+2	73+2	68+2	74+2	74+2	74+2	68+2	65+2	66+2	66+2	66+2	65+2	
3	66+2	64+2	68+2	69+2	69+2	69+2	68+2	67+2	69+2	69+2	69+2	67+2	
4	65+2	59+2	60+2	65+2	65+2	65+2	60+2	59+2	62+2	62+2	62+2	59+2	

	Crossing	Finishing	HeadingAccuracy	ShortPassing	Volleys	Dribbling	\
0	68.0	64.0	61.0	76.0	68.0	69.0	
1	77.0	70.0	54.0	90.0	74.0	90.0	
2	80.0	68.0	43.0	86.0	69.0	78.0	
3	63.0	53.0	62.0	69.0	61.0	66.0	
4	64.0	59.0	58.0	72.0	57.0	62.0	

	Curve	FKAccuracy	LongPassing	BallControl	Acceleration	SprintSpeed	\
0	79.0	79.0	71.0	71.0	56.0	50.0	
1	80.0	70.0	85.0	92.0	70.0	67.0	
2	83.0	80.0	87.0	82.0	54.0	38.0	
3	58.0	54.0	64.0	67.0	65.0	62.0	
4	68.0	67.0	68.0	68.0	33.0	33.0	

```
   Agility  Reactions  Balance  ShotPower  Jumping  Stamina  Strength  \
0     62.0       65.0     72.0       75.0     54.0     64.0      60.0
1     79.0       86.0     84.0       65.0     47.0     55.0      58.0
2     68.0       73.0     81.0       77.0     72.0     61.0      65.0
3     69.0       67.0     63.0       71.0     69.0     74.0      67.0
4     60.0       67.0     91.0       70.0     60.0     50.0      58.0

   LongShots  Aggression  Interceptions  Positioning  Vision  Penalties  \
0       71.0        71.0           71.0         72.0    73.0       75.0
1       71.0        58.0           66.0         81.0    93.0       71.0
2       76.0        87.0           66.0         63.0    86.0       85.0
3       65.0        75.0           70.0         61.0    65.0       65.0
4       66.0        74.0           63.0         55.0    64.0       66.0

   Composure  Marking  StandingTackle  SlidingTackle  GKDiving  GKHandling  \
0       79.0     70.0            43.0           40.0       9.0        12.0
1       89.0     67.0            57.0           56.0       6.0        13.0
2       80.0     65.0            67.0           60.0       7.0        11.0
3       68.0     74.0            71.0           66.0       5.0        15.0
4       82.0     59.0            66.0           62.0       6.0         9.0

   GKKicking  GKPositioning  GKReflexes Release Clause
0       13.0           11.0        11.0          €1.1M
1        6.0           13.0         7.0         €26.9M
2        7.0           14.0         8.0          €7.6M
3        7.0            6.0        13.0          €372K
4       10.0            9.0         7.0          €263K
```

[12]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 18207 entries, 0 to 18206
Data columns (total 88 columns):
 #   Column                    Non-Null Count  Dtype
---  ------                    --------------  -----
 0   ID                        18207 non-null  int64
 1   Name                      18207 non-null  object
 2   Age                       18207 non-null  int64
 3   Photo                     18207 non-null  object
 4   Nationality               18207 non-null  object
 5   Flag                      18207 non-null  object
 6   Overall                   18207 non-null  int64
 7   Potential                 18207 non-null  int64
 8   Club                      17966 non-null  object
 9   Club Logo                 18207 non-null  object
 10  Value                     18207 non-null  object
```

```
11  Wage                      18207 non-null  object
12  Special                   18207 non-null  int64
13  Preferred Foot            18159 non-null  object
14  International Reputation  18159 non-null  float64
15  Weak Foot                 18159 non-null  float64
16  Skill Moves               18159 non-null  float64
17  Work Rate                 18159 non-null  object
18  Body Type                 18159 non-null  object
19  Real Face                 18159 non-null  object
20  Position                  18147 non-null  object
21  Jersey Number             18147 non-null  float64
22  Joined                    16654 non-null  object
23  Loaned From               1264 non-null   object
24  Contract Valid Until      17918 non-null  object
25  Height                    18159 non-null  object
26  Weight                    18159 non-null  object
27  LS                        16122 non-null  object
28  ST                        16122 non-null  object
29  RS                        16122 non-null  object
30  LW                        16122 non-null  object
31  LF                        16122 non-null  object
32  CF                        16122 non-null  object
33  RF                        16122 non-null  object
34  RW                        16122 non-null  object
35  LAM                       16122 non-null  object
36  CAM                       16122 non-null  object
37  RAM                       16122 non-null  object
38  LM                        16122 non-null  object
39  LCM                       16122 non-null  object
40  CM                        16122 non-null  object
41  RCM                       16122 non-null  object
42  RM                        16122 non-null  object
43  LWB                       16122 non-null  object
44  LDM                       16122 non-null  object
45  CDM                       16122 non-null  object
46  RDM                       16122 non-null  object
47  RWB                       16122 non-null  object
48  LB                        16122 non-null  object
49  LCB                       16122 non-null  object
50  CB                        16122 non-null  object
51  RCB                       16122 non-null  object
52  RB                        16122 non-null  object
53  Crossing                  18159 non-null  float64
54  Finishing                 18159 non-null  float64
55  HeadingAccuracy           18159 non-null  float64
56  ShortPassing              18159 non-null  float64
57  Volleys                   18159 non-null  float64
58  Dribbling                 18159 non-null  float64
```

```
 59  Curve                       18159 non-null  float64
 60  FKAccuracy                  18159 non-null  float64
 61  LongPassing                 18159 non-null  float64
 62  BallControl                 18159 non-null  float64
 63  Acceleration                18159 non-null  float64
 64  SprintSpeed                 18159 non-null  float64
 65  Agility                     18159 non-null  float64
 66  Reactions                   18159 non-null  float64
 67  Balance                     18159 non-null  float64
 68  ShotPower                   18159 non-null  float64
 69  Jumping                     18159 non-null  float64
 70  Stamina                     18159 non-null  float64
 71  Strength                    18159 non-null  float64
 72  LongShots                   18159 non-null  float64
 73  Aggression                  18159 non-null  float64
 74  Interceptions               18159 non-null  float64
 75  Positioning                 18159 non-null  float64
 76  Vision                      18159 non-null  float64
 77  Penalties                   18159 non-null  float64
 78  Composure                   18159 non-null  float64
 79  Marking                     18159 non-null  float64
 80  StandingTackle              18159 non-null  float64
 81  SlidingTackle               18159 non-null  float64
 82  GKDiving                    18159 non-null  float64
 83  GKHandling                  18159 non-null  float64
 84  GKKicking                   18159 non-null  float64
 85  GKPositioning               18159 non-null  float64
 86  GKReflexes                  18159 non-null  float64
 87  Release Clause              16643 non-null  object
dtypes: float64(38), int64(5), object(45)
memory usage: 12.2+ MB
```

Objective and Insights These are the instructions that i coppied from excel file

Determine the outliers for Wages and mentioned the steps, process and logic

Analyze the distribution for potential column.

Difference between normal and student t distribution explain it using 'potential' column.

Difference between normal and standard normal distribution explain it using 'potential' column.

find the 95%, 90%, and 99%, confidence interval for 'Potential','wage','weight' column.

find the 95%, 90%, and 99%, confidence interval for 'Potential','wage','weight' column.

Proove Central Limit Theorom by using 'potential' column of the game_data.

Pls give any insgights by anlysing the data in your own and make PPT out of it (Currently we are not checking PPT skills so you can paste the graphs and write insights

I am going to load only the specific columns required to complete the tasks mentioned in the instructions and include 2-3 extra columns for additional analysis

[13]:
```
df.isnull().sum()/len(df)*100
```

[13]:
```
ID                  0.000000
Name                0.000000
Age                 0.000000
Photo               0.000000
Nationality         0.000000
                      …
GKHandling          0.263635
GKKicking           0.263635
GKPositioning       0.263635
GKReflexes          0.263635
Release Clause      8.590103
Length: 88, dtype: float64
```

[15]:
```
# Only these columns i will load from fame excel file
col = ['ID','Name', 'Age', 'Overall', 'Club', 'Value', 'Wage', 'Potential',␣
 ↪'Weight']
```

[16]:
```
df=pd.read_excel("Game.xlsx",usecols=col)
```

[17]:
```
df
```

[17]:
```
           ID          Name  Age  Overall  Potential                   Club  \
0          16   Luis García   37       71         71               KAS Eupen
1          41       Iniesta   34       86         86             Vissel Kobe
2          80  E. Belözoğlu   37       79         79  Medipol Başakşehir FK
3         164      G. Pinzi   37       70         70                  Padova
4         657    D. Vaughan   35       66         66            Notts County
…         …             …   …      …          …                     …
18202  246609    J. Requena   19       57         72       Newell's Old Boys
18203  246613     J. Zwarts   19       62         77               Feyenoord
18204  246616     José Uche   18       58         69               SD Huesca
18205  246617     Javi Mier   19       62         76             Real Oviedo
18206  246620      E. McCue   17       51         74          Houston Dynamo

        Value  Wage  Weight
0       €750K   €6K  143lbs
1      €21.5M  €21K  150lbs
2         €4M  €23K  159lbs
3       €240K   €2K  168lbs
4       €150K   €4K  154lbs
…         …    …     …
18202   €220K   €1K  159lbs
18203   €650K   €1K  163lbs
18204   €180K   €1K  161lbs
18205   €650K   €1K  143lbs
```

```
18206     €70K    €1K   185lbs

[18207 rows x 9 columns]
```

[18]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 18207 entries, 0 to 18206
Data columns (total 9 columns):
 #   Column     Non-Null Count  Dtype
---  ------     --------------  -----
 0   ID         18207 non-null  int64
 1   Name       18207 non-null  object
 2   Age        18207 non-null  int64
 3   Overall    18207 non-null  int64
 4   Potential  18207 non-null  int64
 5   Club       17966 non-null  object
 6   Value      18207 non-null  object
 7   Wage       18207 non-null  object
 8   Weight     18159 non-null  object
dtypes: int64(4), object(5)
memory usage: 1.3+ MB
```

[19]:
```
df.isnull().sum()/len(df)*100
```

[19]:
```
ID           0.000000
Name         0.000000
Age          0.000000
Overall      0.000000
Potential    0.000000
Club         1.323667
Value        0.000000
Wage         0.000000
Weight       0.263635
dtype: float64
```

1 Determine the outliers for Wages and mentioned the steps, process and logic

[20]:
```
df['Wage'].describe()
```

[20]:
```
count     18207
unique      144
top         €1K
freq       4900
Name: Wage, dtype: object
```

[21]:
```python
df['Wage'].head()
```

[21]:
```
0     €6K
1    €21K
2    €23K
3     €2K
4     €4K
Name: Wage, dtype: object
```

[25]:
```python
df['Wage']=df['Wage'].str.replace('€','')
df['Wage']=df['Wage'].str.replace('K','000')
df['Wage'] = df['Wage'].str.replace('M', '000000')
```

[30]:
```python
df['Wage']=df['Wage'].astype(float)
```

[31]:
```python
# outlier detection using IQR METHOD (INTER QUANTILE RANGE)
```

[32]:
```python
Q1 = df['Wage'].quantile(0.25)
Q3 = df['Wage'].quantile(0.75)
```

[33]:
```python
Q1
```

[33]:
```
1000.0
```

[34]:
```python
Q3
```

[34]:
```
9000.0
```

[36]:
```python
IQR= Q3-Q1
print(IQR)
```

```
8000.0
```

[37]:
```python
# upper limit and lower limit

upper_limit = Q3+1.5*(IQR)
print(upper_limit)

lower_limit = Q1-1.5*(IQR)
print(lower_limit)
```

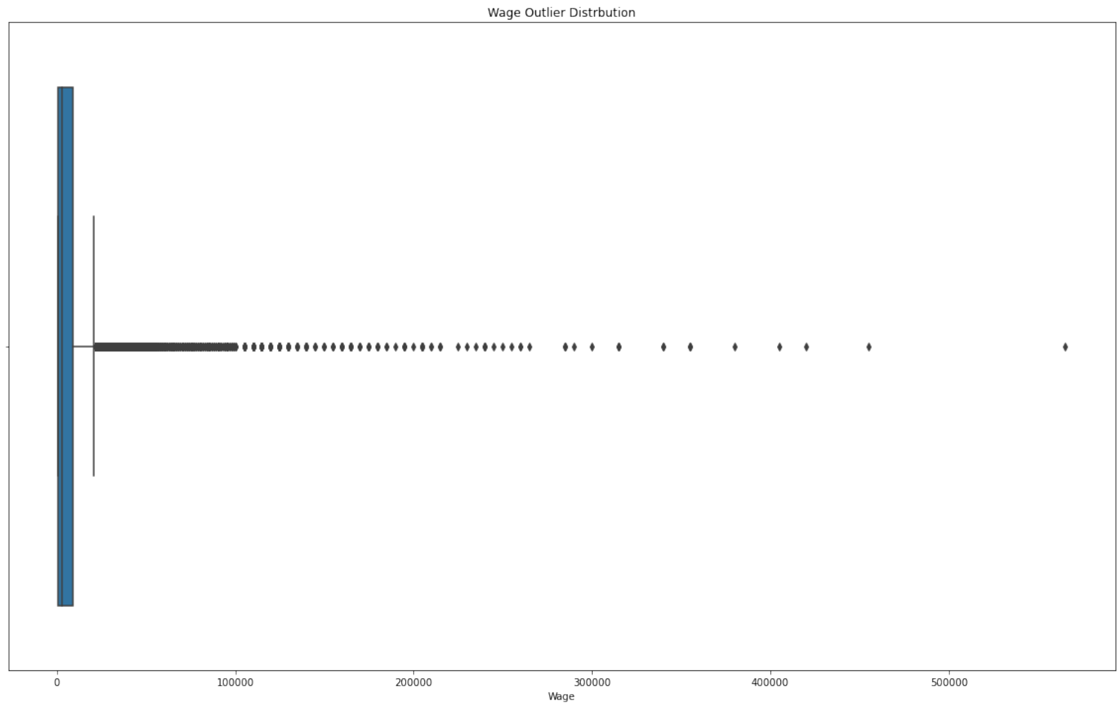
```
21000.0
-11000.0
```

[40]:
```python
outliers = df[(df['Wage']>upper_limit) | (df['Wage']<lower_limit) ]
```

[41]:
```python
outliers
```

[41]:
```
            ID             Name  Age  Overall  Potential  \
2           80      E. Belözoğlu   37       79         79
6         1179        G. Buffon   40       88         88
7         2147  M. Stekelenburg   35       73         73
17        9014        A. Robben   34       84         84
43       20775         Quaresma   34       84         84
…          …              …  …    …        …
17168  244544         B. Alıcı   21       73         80
17567  245221   O. Abdulrahman   26       77         78
17579  245238     Javi Sánchez   21       67         79
17606  245279         Reguilón   21       68         80
18010  246069          L. Rupp   27       76         76

                       Club   Value      Wage   Weight
2      Medipol Başakşehir FK     €4M   23000.0  159lbs
6        Paris Saint-Germain     €4M   77000.0  203lbs
7                    Everton   €950K   30000.0  203lbs
17         FC Bayern München  €15.5M  110000.0  176lbs
43               Beşiktaş JK  €15.5M   80000.0  148lbs
…                         …     …         …      …
17168         Fenerbahçe SK   €5.5M   33000.0  154lbs
17567              Al Hilal  €10.5M   39000.0  132lbs
17579           Real Madrid   €1.2M   24000.0  170lbs
17606           Real Madrid   €1.4M   28000.0  150lbs
18010     TSG 1899 Hoffenheim     €8M   32000.0  161lbs

[2031 rows x 9 columns]
```

[49]:
```
plt.figure(figsize=(20,12))
plt.title("Wage Outlier Distrbution")
sns.boxplot(df['Wage']);
```

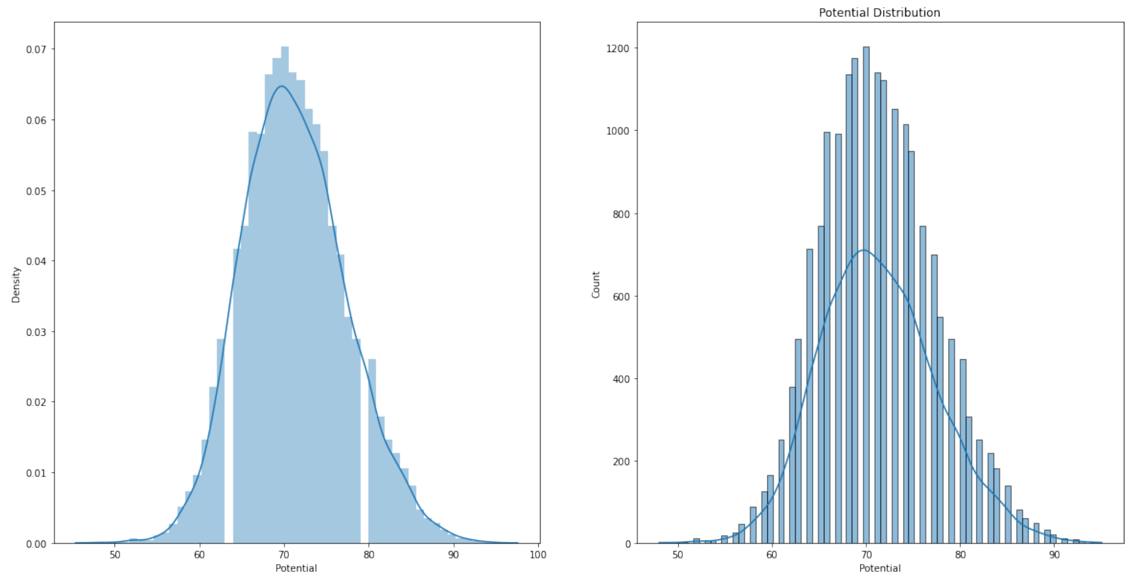


[53]:
```
print(outliers['Wage'].max())
print(outliers['Wage'].min())
```

```
565000.0
22000.0
```

2 2. Analyze the distribution for potential column.

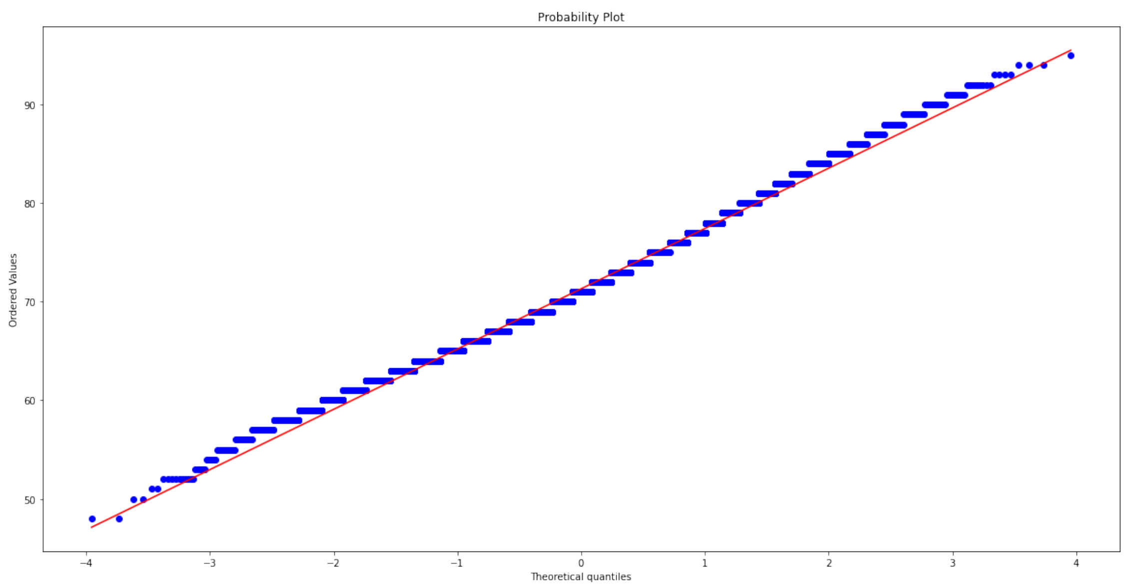
[73]:
```
plt.figure(figsize=(20,10))
plt.title("Potential Distribution")
plt.subplot(1,2,1)
sns.distplot(df['Potential'])

plt.subplot(1,2,2)
plt.title("Potential Distribution")
sns.histplot(df['Potential'],kde=True);
```



[78]:
```
from scipy import stats as st
```

[68]:
```
import pylab
```

[72]:
```
plt.figure(figsize=(20,10))
plt.title("Potential Distribution")
st.probplot(df['Potential'],dist='norm',plot=pylab);
```



[76]:
```
```

[76]:
```
0    70
dtype: int64
```

Conclusions:-

The 'Potential' column contains 18,207 data points.

The mean potential is approximately 71.31, with a standard deviation of around 6.14.

The minimum potential is 48, and the maximum potential is 95.

The histogram and density plot both exhibit a bell-shaped curve, indicating a normal distribution.

The Q-Q plot shows a nearly straight line, suggesting the data points follow a normal distribution.

Overall, the 'Potential' column appears to be normally distributed.

3 3 Difference between normal and student t distribution explain it using 'potential' column

Normal Distrbution

~~ normal distribution which is also known as Gaussian Distribution and it is a countinous probability distribution with a Bell shaped curve

~~ Mean ,median , mode of 'Potential' are nearly equals and that indicates it follow normal distribution

~~ Normal distribution should follow Emperical Rule

~~ In the 'Potential' column context, the normal distribution would describe how the 'Potential' values are distributed around their mean value.

~~ The majority of 'Potential' values would be concentrated around the mean, and the distribution would be symmetric.

Student t distribution

~~ The Student's t-distribution is also a continuous probability distribution, but it has heavier tails than the normal distribution.

~~ It is used when the sample size is small or when the population standard deviation is unknown.

~~ In the 'Potential' column context, the Student's t-distribution might be relevant if we are dealing with a small sample of 'Potential' values or if the population standard deviation is not known.

4 4. Difference between normal and standard normal distribution explain it using 'potential' column.

Normal Distrbution

~~ normal distribution which is also known as Gaussian Distribution and it is a countinous probability distribution with a Bell shaped curve

~~ Mean ,median , mode of ‘Potential’ are nearly equals and that indicates it follow normal distribution

~~ Normal distribution should follow Emperical Rule

~~ In the ‘Potential’ column context, the normal distribution would describe how the ‘Potential’ values are distributed around their mean value.

~~ The majority of ‘Potential’ values would be concentrated around the mean, and the distribution would be symmetric.

Standard Normal Distribution:

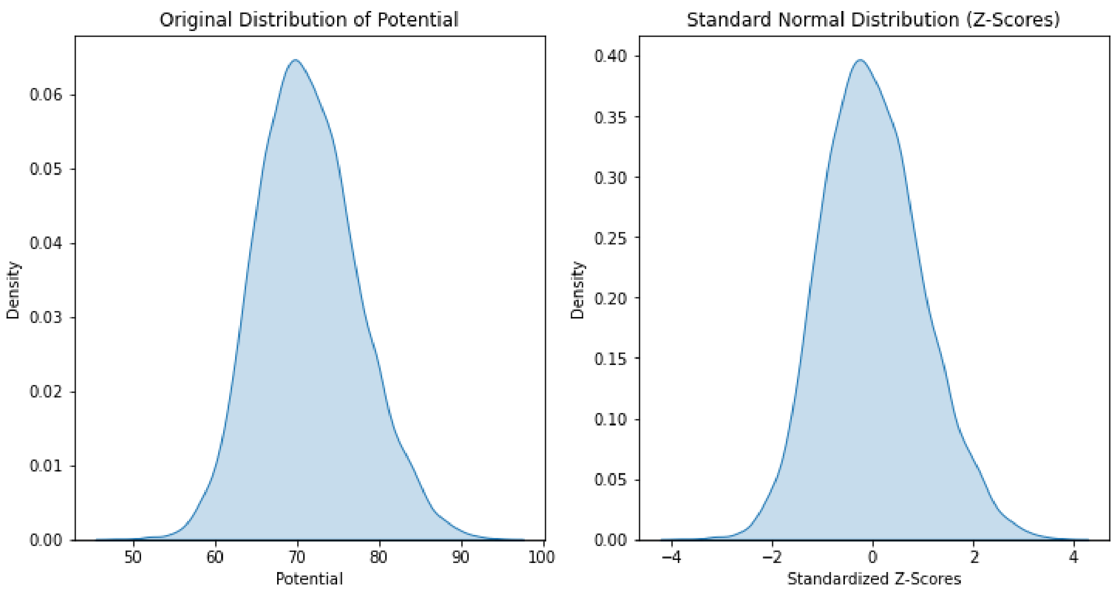
~~The standard normal distribution is a specific type of normal distribution with a mean of 0 and a standard deviation of 1.

~~To convert data from a normal distribution with any mean and standard deviation to the standard normal distribution, we can use a process called standardization.

~~In standardization, we use z-scores, which represent how many standard deviations each ‘Potential’ value is away from the mean of the original distribution.

~~By standardizing the data, the resulting dataset will have a mean of 0 and a standard deviation of 1.

[79]:
```
mean_potential = df['Potential'].mean()
std_potential = df['Potential'].std()

# Calculate z-scores for 'Potential' data
z_scores = st.zscore(df['Potential'])

# Plot density plot (KDE) of the 'Potential' data before standardization
plt.figure(figsize=(12, 6))
plt.subplot(1, 2, 1)
sns.kdeplot(df['Potential'], fill=True)
plt.xlabel('Potential')
plt.ylabel('Density')
plt.title('Original Distribution of Potential')

# Plot density plot (KDE) of the standardized 'Potential' data (Z-Scores)
plt.subplot(1, 2, 2)
sns.kdeplot(z_scores, fill=True)
plt.xlabel('Standardized Z-Scores')
plt.ylabel('Density')
plt.title('Standard Normal Distribution (Z-Scores)')
plt.show()
```



5 5 CENTRAL LIMIT THEOREM ON POTENTIAL

The central limit theorem says that the sampling distribution of the mean will always be normally distributed, as long as the sample size is large enough.

[82]:
```
# Define the sample sizes
sample_sizes = [5, 30, 50]

# Initialize a dictionary to store sample means for each sample size
sample_means_dict = {size: [] for size in sample_sizes}
sample_means_dict
```

[82]: {5: [], 30: [], 50: []}

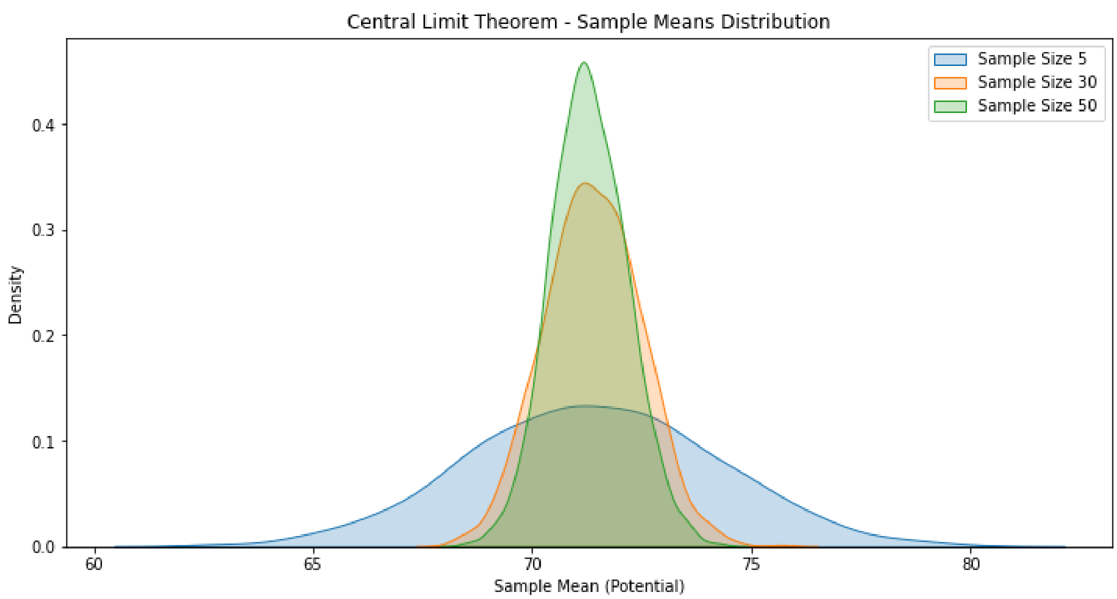
[83]:
```
# Perform sampling and calculate sample means for each sample size
for size in sample_sizes:
    for _ in range(1000):  # Taking 1000 random samples for each size
        sample = df['Potential'].sample(size, replace=True)  # Sampling with␣
↪replacement(1 data points may be select two times)
        sample_mean = np.mean(sample)
        sample_means_dict[size].append(sample_mean)

pd.DataFrame(sample_means_dict)
```

[83]:

	5	30	50
0	70.6	70.200000	70.56
1	68.6	69.900000	70.62

```
2    72.6  71.900000  70.78
3    74.6  70.400000  70.50
4    71.0  72.166667  71.42
..    …         …     …
995  76.4  72.833333  69.16
996  73.4  70.466667  72.10
997  70.6  74.700000  72.54
998  71.6  71.066667  70.62
999  75.2  70.733333  70.16

[1000 rows x 3 columns]
```

[84]:
```python
# Plot KDE plots of the sample means for each sample size
plt.figure(figsize=(12, 6))
for size in sample_sizes:
    sns.kdeplot(sample_means_dict[size], fill=True, label=f'Sample Size {size}')
plt.xlabel('Sample Mean (Potential)')
plt.ylabel('Density')
plt.title('Central Limit Theorem - Sample Means Distribution')
plt.legend()
plt.show()
```



We can see clearly here that the distribution approaches normal as sample size gets larger. It is evident from the graphs that as we keep on increasing the sample size from 5 to 50 the histogram tends to take the shape of a normal distribution.

6 Getting top 10 players according to potential

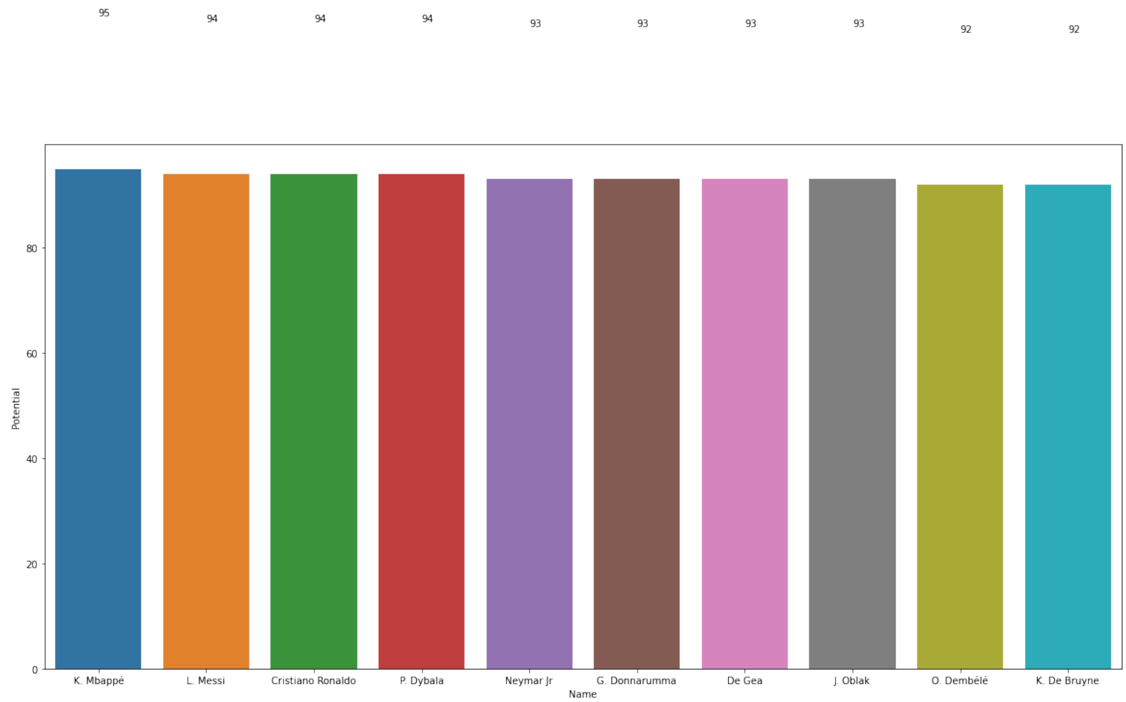
[88]:
```
High_performance_player = df.sort_values(by='Potential',ascending=False)
```

[91]:
```
H_P_P=High_performance_player.nlargest(10,'Potential')
```

[92]:
```
H_P_P
```

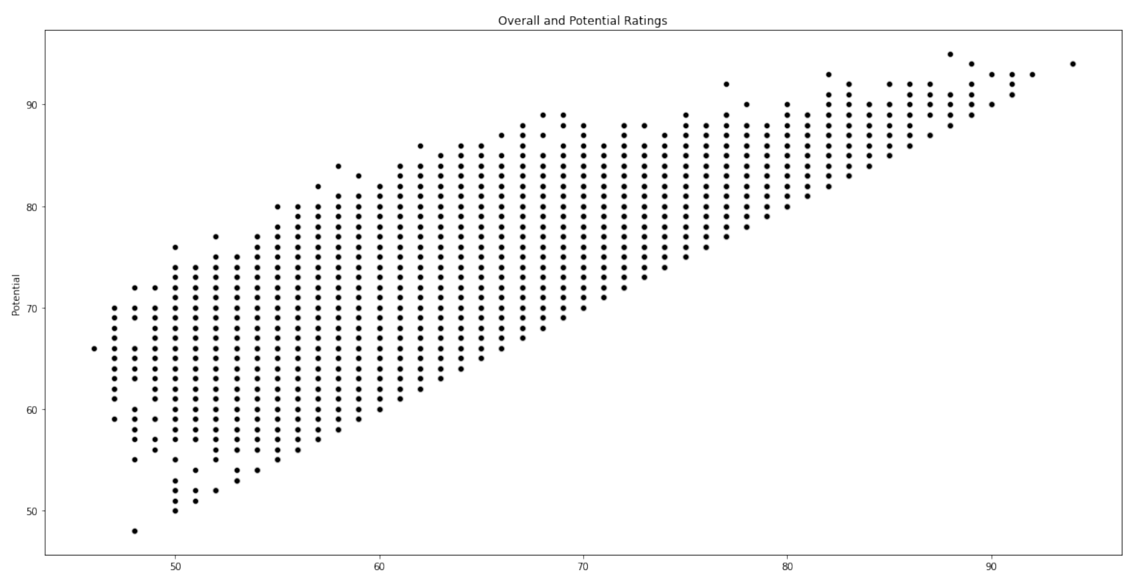
[92]:
```
            ID               Name  Age  Overall  Potential  \
12046   231747          K. Mbappé   19       88         95
699     158023           L. Messi   31       94         94
45       20801  Cristiano Ronaldo   33       94         94
6832    211110          P. Dybala   24       89         94
3255    190871          Neymar Jr   26       92         93
11717   230621       G. Donnarumma   19       82         93
3591    193080             De Gea   27       91         93
4562    200389           J. Oblak   25       90         93
11947   231443         O. Dembélé   21       83         92
3576    192985       K. De Bruyne   27       91         92

                     Club    Value      Wage  Weight
12046   Paris Saint-Germain     €81M  100000.0  161lbs
699            FC Barcelona  €110.5M  565000.0  159lbs
45                 Juventus     €77M  405000.0  183lbs
6832               Juventus     €89M  205000.0  165lbs
3255    Paris Saint-Germain  €118.5M  290000.0  150lbs
11717                 Milan     €29M   23000.0  198lbs
3591      Manchester United     €72M  260000.0  168lbs
4562        Atlético Madrid     €68M   94000.0  192lbs
11947          FC Barcelona     €40M  155000.0  148lbs
3576        Manchester City    €102M  355000.0  154lbs
```

[110]:
```
plt.figure(figsize=(20,10))
sns.barplot(H_P_P['Name'],H_P_P['Potential'],data=H_P_P);
for i , v in enumerate(H_P_P['Potential']):
    plt.text(x=i,y=v+29,s=f"{v}")
```



7 Comparison of Overall and Potential Ratings

[124]:
```
plt.figure(figsize=(20,10))
plt.title("Overall and Potential Ratings")
sns.scatterplot(df['Overall'],df['Potential'],color='black',data=df)
plt.plot();
```



player who have high potential also have high overall performance!