European Soccer Data Analysis

May 7, 2019

1 Project: Investigate European Soccer Datasets

1.1 Table of Contents

Introduction

World Cup is a huge soccer summit every four years. Although I am not a huge soccer fan of any player or team, I still enjoy watching world cup and buying sport lotteries. This dataset gives me a broad view of teams and players of European Leagues, which is almost 90% of best soccer players around the world. I am interested in investigating these questions listed below that might be helpful for me to make better predictions of next World Cup matches.

1. What are the most powerful skills that a good soccer player must have?

2. Do star players make a huge difference to match results?
3. Are match results related to the sum of overall rating of players?

```
In [1]: import pandas as pd
        import numpy as np
        from numpy.polynomial.polynomial import polyfit
        import matplotlib.pyplot as plt
        import seaborn as sns
        import sqlite3
        from pandas.plotting import scatter_matrix
        %matplotlib inline
```

Data Wrangling

1.1.1 General Properties and Data Cleaning

Read all tables from the sqlite database and merge some tables that are relatively small.

```
In [2]: # Load your data and print out a few lines. Perform operations to inspect data
        #   types and look for instances of missing or possibly errant data.
```

```
# Create your connection.
cnx = sqlite3.connect('database.sqlite')

df_country = pd.read_sql_query("SELECT * FROM Country", cnx)
df_league = pd.read_sql_query("SELECT * FROM League", cnx)
df_match = pd.read_sql_query("SELECT * FROM Match", cnx)
df_player = pd.read_sql_query("SELECT * FROM Player", cnx)
df_player_attributes = pd.read_sql_query("SELECT * FROM Player_Attributes", cnx)
df_team = pd.read_sql_query("SELECT * FROM Team", cnx)
df_team_attributes = pd.read_sql_query("SELECT * FROM Team_Attributes", cnx)
```

1.1.2 Combine country and league tables since they have the same id numbers

In [3]:
```
df_country_league =  df_country.merge(df_league, left_on = 'id', right_on = 'country_id
df_country_league.rename(index = str, columns = {'name_x':'country_name','name_y':'leag
df_country_league
```

Out[3]:
```
   country_name  country_id               league_name
0       Belgium           1    Belgium Jupiler League
1       England        1729    England Premier League
2        France        4769            France Ligue 1
3       Germany        7809     Germany 1. Bundesliga
4         Italy       10257             Italy Serie A
5   Netherlands       13274    Netherlands Eredivisie
6        Poland       15722        Poland Ekstraklasa
7      Portugal       17642  Portugal Liga ZON Sagres
8      Scotland       19694   Scotland Premier League
9         Spain       21518           Spain LIGA BBVA
10  Switzerland       24558  Switzerland Super League
```

In [4]:
```
df_match.head(1)
```

Out[4]:
```
   id  country_id  league_id     season  stage                 date  \
0   1           1          1  2008/2009      1  2008-08-17 00:00:00

   match_api_id  home_team_api_id  away_team_api_id  home_team_goal  ...  SJA  \
0        492473              9987              9993               1  ...  4.0

    VCH  VCD  VCA   GBH   GBD  GBA   BSH  BSD  BSA
0  1.65  3.4  4.5  1.78  3.25  4.0  1.73  3.4  4.2

[1 rows x 115 columns]
```

I don't know the soccer terminologies at the end of this table, so I decided to ignore that part and only have match date, season, result and player ids in the match table. Merging the `country_league` table to the `match` table makes it easier to investigate the relationship of leagues/countries with match results.

```
In [5]: df = df_match.iloc[:,[0,1,3,5,6,7,8,9,10,55,56,57,58,59,60,61,62,63,64,65,66,67,68,69,
        df_match = df_country_league.merge(df,on = 'country_id')
```

```
In [7]: df_match.head()
```

```
Out[7]:   country_name  country_id              league_name  id     season  \
        0      Belgium           1  Belgium Jupiler League   1  2008/2009
        1      Belgium           1  Belgium Jupiler League   2  2008/2009
        2      Belgium           1  Belgium Jupiler League   3  2008/2009
        3      Belgium           1  Belgium Jupiler League   4  2008/2009
        4      Belgium           1  Belgium Jupiler League   5  2008/2009

                          date  match_api_id  home_team_api_id  away_team_api_id  \
        0  2008-08-17 00:00:00        492473              9987              9993
        1  2008-08-16 00:00:00        492474             10000              9994
        2  2008-08-16 00:00:00        492475              9984              8635
        3  2008-08-17 00:00:00        492476              9991              9998
        4  2008-08-16 00:00:00        492477              7947              9985

           home_team_goal  ...  away_player_2  away_player_3  away_player_4  \
        0               1  ...            NaN            NaN            NaN
        1               0  ...            NaN            NaN            NaN
        2               0  ...            NaN            NaN            NaN
        3               5  ...            NaN            NaN            NaN
        4               1  ...            NaN            NaN            NaN

           away_player_5  away_player_6  away_player_7  away_player_8  away_player_9  \
        0            NaN            NaN            NaN            NaN            NaN
        1            NaN            NaN            NaN            NaN            NaN
        2            NaN            NaN            NaN            NaN            NaN
        3            NaN            NaN            NaN            NaN            NaN
        4            NaN            NaN            NaN            NaN            NaN

           away_player_10  away_player_11
        0             NaN             NaN
        1             NaN             NaN
        2             NaN             NaN
        3             NaN             NaN
        4             NaN             NaN

        [5 rows x 33 columns]
```

```
In [8]: df_match.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 25979 entries, 0 to 25978
Data columns (total 33 columns):
country_name        25979 non-null object
country_id          25979 non-null int64
```

```
league_name         25979 non-null object
id                  25979 non-null int64
season              25979 non-null object
date                25979 non-null object
match_api_id        25979 non-null int64
home_team_api_id    25979 non-null int64
away_team_api_id    25979 non-null int64
home_team_goal      25979 non-null int64
away_team_goal      25979 non-null int64
home_player_1       24755 non-null float64
home_player_2       24664 non-null float64
home_player_3       24698 non-null float64
home_player_4       24656 non-null float64
home_player_5       24663 non-null float64
home_player_6       24654 non-null float64
home_player_7       24752 non-null float64
home_player_8       24670 non-null float64
home_player_9       24706 non-null float64
home_player_10      24543 non-null float64
home_player_11      24424 non-null float64
away_player_1       24745 non-null float64
away_player_2       24701 non-null float64
away_player_3       24686 non-null float64
away_player_4       24658 non-null float64
away_player_5       24644 non-null float64
away_player_6       24666 non-null float64
away_player_7       24744 non-null float64
away_player_8       24638 non-null float64
away_player_9       24651 non-null float64
away_player_10      24538 non-null float64
away_player_11      24425 non-null float64
dtypes: float64(22), int64(7), object(4)
memory usage: 6.7+ MB
```

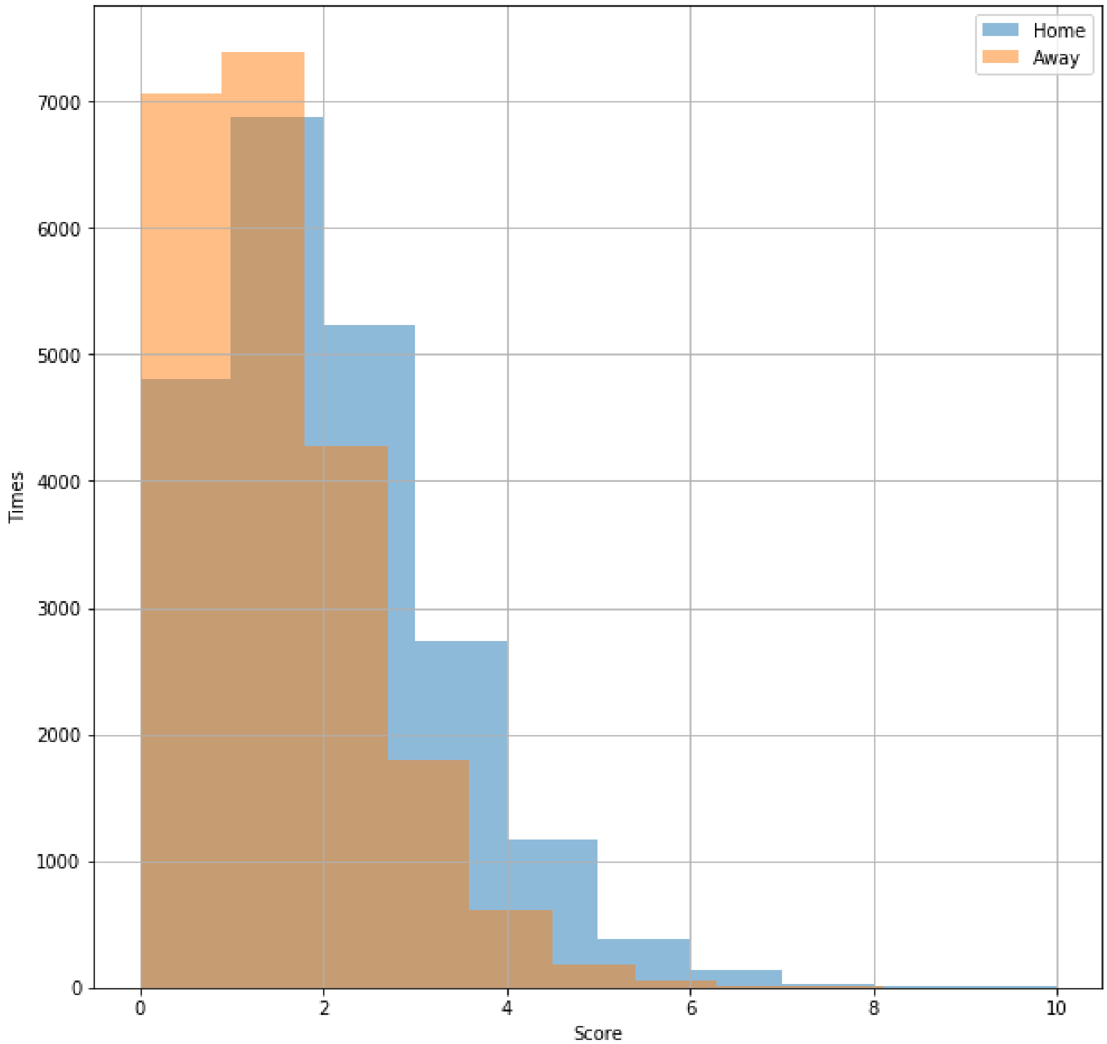
We have some missing player list for some matches and the value can't be filled with average or mean value since they are player ID. I decided to drop those matches with missing values.

```
In [9]: df_match.dropna(axis=0, inplace = True)
```

```
In [11]: def plotting(col_name, label, size):
             col_name.hist(figsize = size, alpha = 0.5, label = label)

         plotting(df_match.home_team_goal, 'Home',(10,10))
         plotting(df_match.away_team_goal, 'Away',(10,10))
         plt.legend()
         plt.xlabel('Score')
         plt.ylabel('Times')
```

```
Out[11]: Text(0, 0.5, 'Times')
```



Home teams are typically getting more scores than away teams.

Checked there's no missing value in the dataframe anymore. The other important dataframe for my analysis is the player dataframe. I am planning to combine the player dataframe with player attribute dataframe since they are relevent and easy to be combined.

```
In [12]: df_player.head()
```

```
Out[12]:    id  player_api_id         player_name  player_fifa_api_id  \
         0   1         505942  Aaron Appindangoye              218353
         1   2         155782     Aaron Cresswell              189615
         2   3         162549         Aaron Doran              186170
         3   4          30572       Aaron Galindo              140161
         4   5          23780        Aaron Hughes               17725
```

```
             birthday  height  weight
0  1992-02-29 00:00:00  182.88     187
1  1989-12-15 00:00:00  170.18     146
2  1991-05-13 00:00:00  170.18     163
3  1982-05-08 00:00:00  182.88     198
4  1979-11-08 00:00:00  182.88     154
```

```
In [13]: df_player_attributes.head()
```

```
Out[13]:    id  player_fifa_api_id  player_api_id                 date  overall_rating  \
         0   1              218353         505942  2016-02-18 00:00:00            67.0
         1   2              218353         505942  2015-11-19 00:00:00            67.0
         2   3              218353         505942  2015-09-21 00:00:00            62.0
         3   4              218353         505942  2015-03-20 00:00:00            61.0
         4   5              218353         505942  2007-02-22 00:00:00            61.0

            potential preferred_foot attacking_work_rate defensive_work_rate  crossing  \
         0       71.0          right              medium              medium      49.0
         1       71.0          right              medium              medium      49.0
         2       66.0          right              medium              medium      49.0
         3       65.0          right              medium              medium      48.0
         4       65.0          right              medium              medium      48.0

            ...  vision  penalties  marking  standing_tackle  sliding_tackle  \
         0  ...    54.0       48.0     65.0             69.0            69.0
         1  ...    54.0       48.0     65.0             69.0            69.0
         2  ...    54.0       48.0     65.0             66.0            69.0
         3  ...    53.0       47.0     62.0             63.0            66.0
         4  ...    53.0       47.0     62.0             63.0            66.0

            gk_diving  gk_handling  gk_kicking  gk_positioning  gk_reflexes
         0        6.0         11.0        10.0             8.0          8.0
         1        6.0         11.0        10.0             8.0          8.0
         2        6.0         11.0        10.0             8.0          8.0
         3        5.0         10.0         9.0             7.0          7.0
         4        5.0         10.0         9.0             7.0          7.0

         [5 rows x 42 columns]
```

```
In [14]: df_player_full = df_player_attributes.merge(df_player, on = 'player_api_id')
         df_player_full.drop(['id_y', 'player_fifa_api_id_y'], axis = 1,inplace = True)
         df_player_full.rename(index = str, columns={'id_x':'id','player_fifa_api_id_x':'playe
```

```
In [15]: df_player_full.isna().sum()
```

```
Out[15]: id                      0
         player_fifa_api_id      0
         player_api_id           0
         date                    0
```

```
overall_rating          836
potential               836
preferred_foot          836
attacking_work_rate    3230
defensive_work_rate     836
crossing                836
finishing               836
heading_accuracy        836
short_passing           836
volleys                2713
dribbling               836
curve                  2713
free_kick_accuracy      836
long_passing            836
ball_control            836
acceleration            836
sprint_speed            836
agility                2713
reactions               836
balance                2713
shot_power              836
jumping                2713
stamina                 836
strength                836
long_shots              836
aggression              836
interceptions           836
positioning             836
vision                 2713
penalties               836
marking                 836
standing_tackle         836
sliding_tackle         2713
gk_diving               836
gk_handling             836
gk_kicking              836
gk_positioning          836
gk_reflexes             836
player_name               0
birthday                  0
height                    0
weight                    0
dtype: int64
```

Since I want to evaluate the relationship between overall rating and skills, if the overall rating value is missing for a player, it's a useless data point. I am going to delete those rows with overall rating missing values.

```
In [16]: df_player_full = df_player_full[df_player_full.overall_rating.isna() == 0]
```

In [17]: df_player_full.isna().sum()

Out[17]:
```
id                        0
player_fifa_api_id        0
player_api_id             0
date                      0
overall_rating            0
potential                 0
preferred_foot            0
attacking_work_rate    2394
defensive_work_rate       0
crossing                  0
finishing                 0
heading_accuracy          0
short_passing             0
volleys                1877
dribbling                 0
curve                  1877
free_kick_accuracy        0
long_passing              0
ball_control              0
acceleration              0
sprint_speed              0
agility                1877
reactions                 0
balance                1877
shot_power                0
jumping                1877
stamina                   0
strength                  0
long_shots                0
aggression                0
interceptions             0
positioning               0
vision                 1877
penalties                 0
marking                   0
standing_tackle           0
sliding_tackle         1877
gk_diving                 0
gk_handling               0
gk_kicking                0
gk_positioning            0
gk_reflexes               0
player_name               0
birthday                  0
height                    0
weight                    0
```

```
dtype: int64
```

There are still eight features in the dataframe with missing values. It's easier for me to delete those columns and only consider other features as relevant features to the overall rating.

```
In [18]: df_player_full.drop(['attacking_work_rate','volleys','curve','agility','balance','jum
```

```
In [19]: df_player_full.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 183142 entries, 0 to 183977
Data columns (total 38 columns):
id                      183142 non-null int64
player_fifa_api_id      183142 non-null int64
player_api_id           183142 non-null int64
date                    183142 non-null object
overall_rating          183142 non-null float64
potential               183142 non-null float64
preferred_foot          183142 non-null object
defensive_work_rate     183142 non-null object
crossing                183142 non-null float64
finishing               183142 non-null float64
heading_accuracy        183142 non-null float64
short_passing           183142 non-null float64
dribbling               183142 non-null float64
free_kick_accuracy      183142 non-null float64
long_passing            183142 non-null float64
ball_control            183142 non-null float64
acceleration            183142 non-null float64
sprint_speed            183142 non-null float64
reactions               183142 non-null float64
shot_power              183142 non-null float64
stamina                 183142 non-null float64
strength                183142 non-null float64
long_shots              183142 non-null float64
aggression              183142 non-null float64
interceptions           183142 non-null float64
positioning             183142 non-null float64
penalties               183142 non-null float64
marking                 183142 non-null float64
standing_tackle         183142 non-null float64
gk_diving               183142 non-null float64
gk_handling             183142 non-null float64
gk_kicking              183142 non-null float64
gk_positioning          183142 non-null float64
gk_reflexes             183142 non-null float64
player_name             183142 non-null object
birthday                183142 non-null object
height                  183142 non-null float64
```

```
weight                  183142 non-null int64
dtypes: float64(29), int64(4), object(5)
memory usage: 54.5+ MB
```

In [20]: df_player_full.describe()

```
Out[20]:                    id  player_fifa_api_id  player_api_id  overall_rating  \
         count  183142.000000       183142.000000  183142.000000   183142.000000
         mean    91978.031265       165826.723040  136294.314139       68.600015
         std     53116.611471        53782.559432  137080.717171        7.041139
         min         1.000000            2.000000    2625.000000       33.000000
         25%     45985.250000       155885.000000   34952.000000       64.000000
         50%     91958.500000       183527.000000   78411.000000       69.000000
         75%    137972.750000       199912.000000  191616.000000       73.000000
         max    183978.000000       234141.000000  750584.000000       94.000000

                    potential       crossing      finishing  heading_accuracy  \
         count  183142.000000  183142.000000  183142.000000     183142.000000
         mean       73.460353      55.086883      49.921078         57.266023
         std         6.592271      17.242135      19.038705         16.488905
         min        39.000000       1.000000       1.000000          1.000000
         25%        69.000000      45.000000      34.000000         49.000000
         50%        74.000000      59.000000      53.000000         60.000000
         75%        78.000000      68.000000      65.000000         68.000000
         max        97.000000      95.000000      97.000000         98.000000

                short_passing      dribbling  ...      penalties        marking  \
         count  183142.000000  183142.000000  ...  183142.000000  183142.000000
         mean       62.429672      59.175154  ...      55.003986      46.772242
         std        14.194068      17.744688  ...      15.546519      21.227667
         min         3.000000       1.000000  ...       2.000000       1.000000
         25%        57.000000      52.000000  ...      45.000000      25.000000
         50%        65.000000      64.000000  ...      57.000000      50.000000
         75%        72.000000      72.000000  ...      67.000000      66.000000
         max        97.000000      97.000000  ...      96.000000      96.000000

                standing_tackle      gk_diving    gk_handling     gk_kicking  \
         count    183142.000000  183142.000000  183142.000000  183142.000000
         mean         50.351257      14.704393      16.063612      20.998362
         std          21.483706      16.865467      15.867382      21.452980
         min           1.000000       1.000000       1.000000       1.000000
         25%          29.000000       7.000000       8.000000       8.000000
         50%          56.000000      10.000000      11.000000      12.000000
         75%          69.000000      13.000000      15.000000      15.000000
         max          95.000000      94.000000      93.000000      97.000000

                gk_positioning    gk_reflexes         height         weight
```

```
count   183142.000000  183142.000000  183142.000000  183142.000000
mean        16.132154      16.441439     181.875925     168.769463
std         16.099175      17.198155       6.394896      15.088820
min          1.000000       1.000000     157.480000     117.000000
25%          8.000000       8.000000     177.800000     159.000000
50%         11.000000      11.000000     182.880000     168.000000
75%         15.000000      15.000000     185.420000     179.000000
max         96.000000      96.000000     208.280000     243.000000

[8 rows x 33 columns]
```

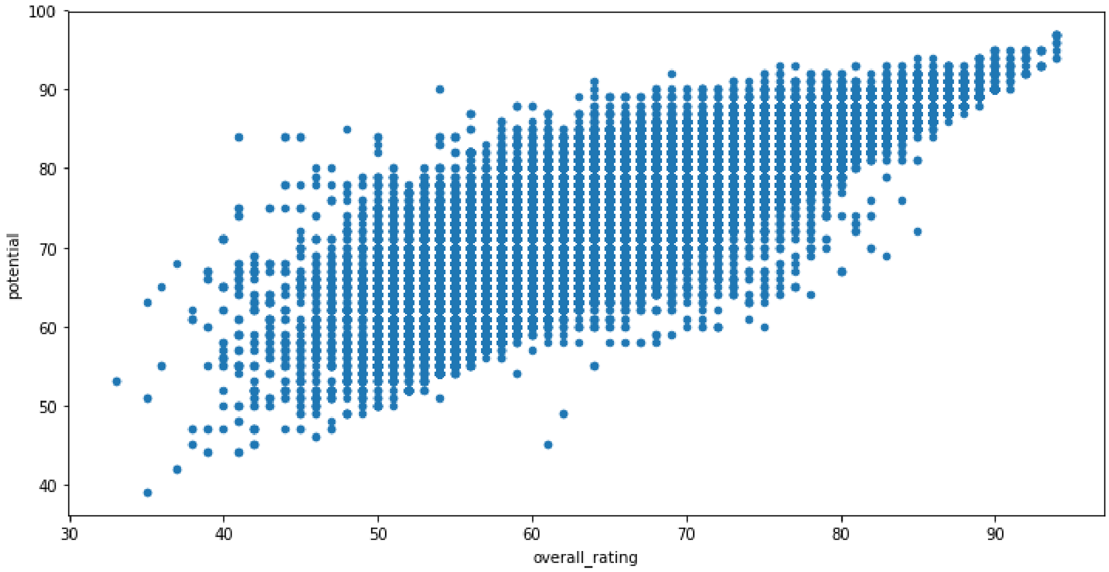
In [21]:
```
df_player_full.corr().overall_rating
```

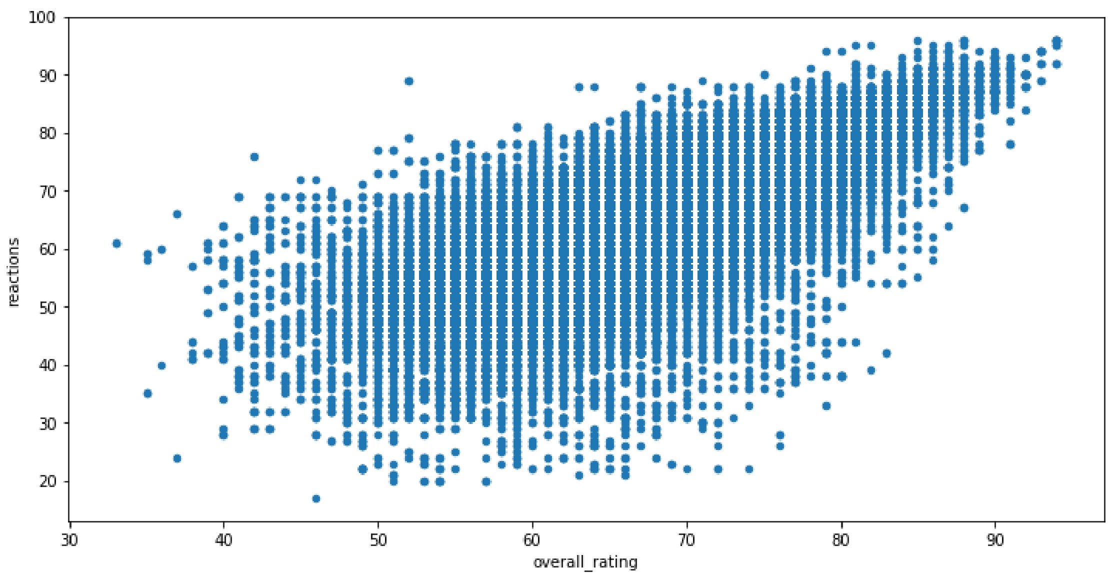
Out[21]:
```
id                    -0.002875
player_fifa_api_id    -0.274089
player_api_id         -0.322389
overall_rating         1.000000
potential              0.766757
crossing               0.357699
finishing              0.329298
heading_accuracy       0.314099
short_passing          0.458361
dribbling              0.354324
free_kick_accuracy     0.349592
long_passing           0.435018
ball_control           0.444257
acceleration           0.245655
sprint_speed           0.254841
reactions              0.769246
shot_power             0.427996
stamina                0.327456
strength               0.318661
long_shots             0.392382
aggression             0.323934
interceptions          0.250370
positioning            0.370019
penalties              0.393189
marking                0.133377
standing_tackle        0.165349
gk_diving              0.027976
gk_handling            0.004410
gk_kicking             0.025682
gk_positioning         0.005709
gk_reflexes            0.005687
height                -0.003475
weight                 0.064396
Name: overall_rating, dtype: float64
```

In [22]:
```
df_player_full.plot('overall_rating', 'potential', kind= 'scatter',figsize = (12,6))
```

```
df_player_full.plot('overall_rating','reactions', kind = 'scatter',figsize = (12,6))
```

Out[22]: <matplotlib.axes._subplots.AxesSubplot at 0x1a1c60ef60>





Overall_rating has strong relationship with Potentials and Reactions. Both features correlation coefficients with Overall_rating is higher than 0.7.

In [23]: df_team.head()

```
Out[23]:    id  team_api_id  team_fifa_api_id     team_long_name team_short_name
        0   1         9987             673.0           KRC Genk             GEN
        1   2         9993             675.0       Beerschot AC             BAC
        2   3        10000           15005.0   SV Zulte-Waregem             ZUL
        3   4         9994            2007.0   Sporting Lokeren             LOK
        4   5         9984            1750.0  KSV Cercle Brugge             CEB
```

```
In [24]: df_team_attributes.head()
```

```
Out[24]:    id  team_fifa_api_id  team_api_id                 date  buildUpPlaySpeed  \
        0   1               434         9930  2010-02-22 00:00:00                60
        1   2               434         9930  2014-09-19 00:00:00                52
        2   3               434         9930  2015-09-10 00:00:00                47
        3   4                77         8485  2010-02-22 00:00:00                70
        4   5                77         8485  2011-02-22 00:00:00                47

          buildUpPlaySpeedClass  buildUpPlayDribbling buildUpPlayDribblingClass  \
        0              Balanced                   NaN                    Little
        1              Balanced                  48.0                    Normal
        2              Balanced                  41.0                    Normal
        3                  Fast                   NaN                    Little
        4              Balanced                   NaN                    Little

           buildUpPlayPassing buildUpPlayPassingClass  ... chanceCreationShooting  \
        0                  50                   Mixed  ...                     55
        1                  56                   Mixed  ...                     64
        2                  54                   Mixed  ...                     64
        3                  70                    Long  ...                     70
        4                  52                   Mixed  ...                     52

           chanceCreationShootingClass chanceCreationPositioningClass  \
        0                       Normal                      Organised
        1                       Normal                      Organised
        2                       Normal                      Organised
        3                         Lots                      Organised
        4                       Normal                      Organised

           defencePressure defencePressureClass  defenceAggression  \
        0               50               Medium                 55
        1               47               Medium                 44
        2               47               Medium                 44
        3               60               Medium                 70
        4               47               Medium                 47

          defenceAggressionClass defenceTeamWidth  defenceTeamWidthClass  \
        0                  Press               45                 Normal
        1                  Press               54                 Normal
        2                  Press               54                 Normal
```

```
3                 Double           70                  Wide
4                  Press           52                Normal

  defenceDefenderLineClass
0                    Cover
1                    Cover
2                    Cover
3                    Cover
4                    Cover

[5 rows x 25 columns]
```

In [25]:
```
df_team_full = df_team.merge(df_team_attributes, on = 'team_api_id')
df_team_full.drop(['id_y','team_fifa_api_id_y'],axis = 1, inplace = True )
df_team_full.rename(columns = {'id_x':'id','team_fifa_api_id_x':'team_fifa_api_id'},i
```

In [26]:
```
df_team_full.info()
```

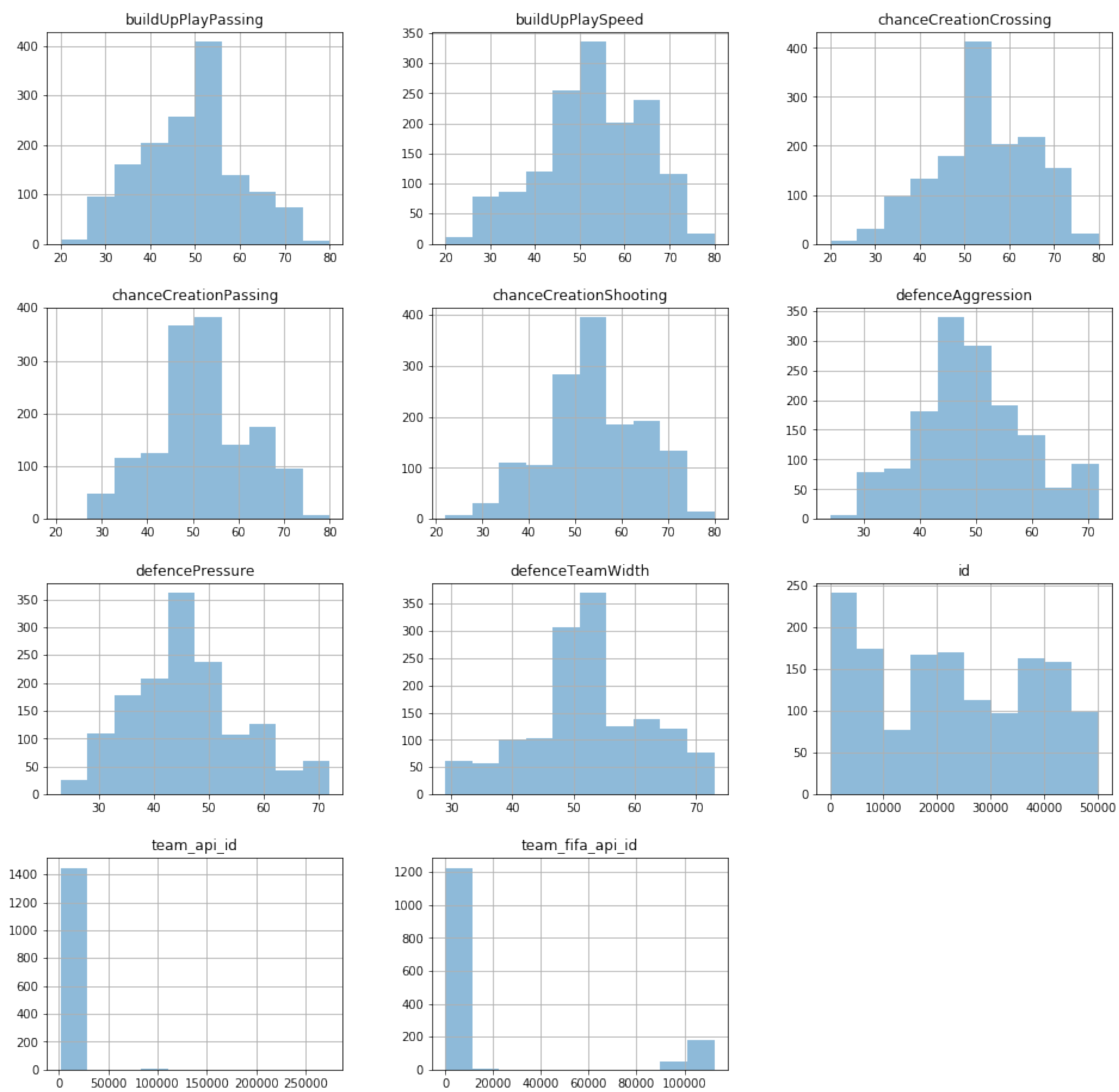
```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 1458 entries, 0 to 1457
Data columns (total 27 columns):
id                                  1458 non-null int64
team_api_id                         1458 non-null int64
team_fifa_api_id                    1458 non-null float64
team_long_name                      1458 non-null object
team_short_name                     1458 non-null object
date                                1458 non-null object
buildUpPlaySpeed                    1458 non-null int64
buildUpPlaySpeedClass               1458 non-null object
buildUpPlayDribbling                489 non-null float64
buildUpPlayDribblingClass           1458 non-null object
buildUpPlayPassing                  1458 non-null int64
buildUpPlayPassingClass             1458 non-null object
buildUpPlayPositioningClass         1458 non-null object
chanceCreationPassing               1458 non-null int64
chanceCreationPassingClass          1458 non-null object
chanceCreationCrossing              1458 non-null int64
chanceCreationCrossingClass         1458 non-null object
chanceCreationShooting              1458 non-null int64
chanceCreationShootingClass         1458 non-null object
chanceCreationPositioningClass      1458 non-null object
defencePressure                     1458 non-null int64
defencePressureClass                1458 non-null object
defenceAggression                   1458 non-null int64
defenceAggressionClass              1458 non-null object
defenceTeamWidth                    1458 non-null int64
defenceTeamWidthClass               1458 non-null object
defenceDefenderLineClass            1458 non-null object
```

```
dtypes: float64(2), int64(10), object(15)
memory usage: 318.9+ KB
```

Only `buildUpPlayDribbling` is a feature with a lot of missing values, so I decided to drop this column.

```
In [27]: df_team_full.drop('buildUpPlayDribbling',axis = 1, inplace = True)
```

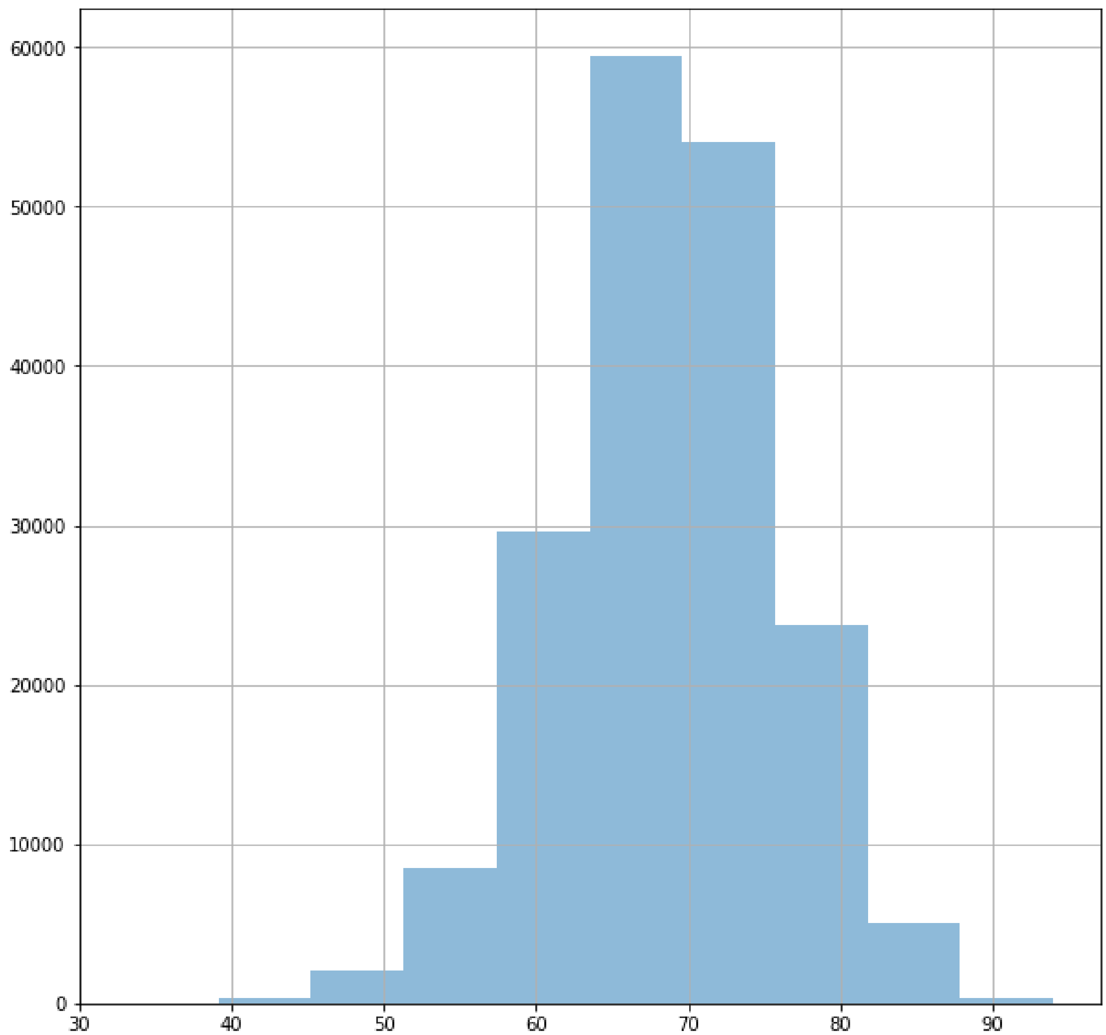
```
In [28]: plotting(df_team_full, '', (16,16))
```



Finally cleaned up all dataframes and the final dataframes I am going to use for analysis are: 1. `df_match` 2. `df_player_full` 3. `df_team_full`
Exploratory Data Analysis

1.1.3 Research Question 1 What are the most powerful skills that a high performance soccer player must have?

```
In [29]: plotting(df_player_full.overall_rating, '', (10,10))
```




```
In [30]: df_player_full.overall_rating.describe()
```

```
Out[30]: count    183142.000000
         mean         68.600015
         std           7.041139
         min          33.000000
         25%          64.000000
         50%          69.000000
         75%          73.000000
         max          94.000000
         Name: overall_rating, dtype: float64
```

I defined those players have overall ratings higher than two standard deviations than mean (top 5%) are hight performance players.

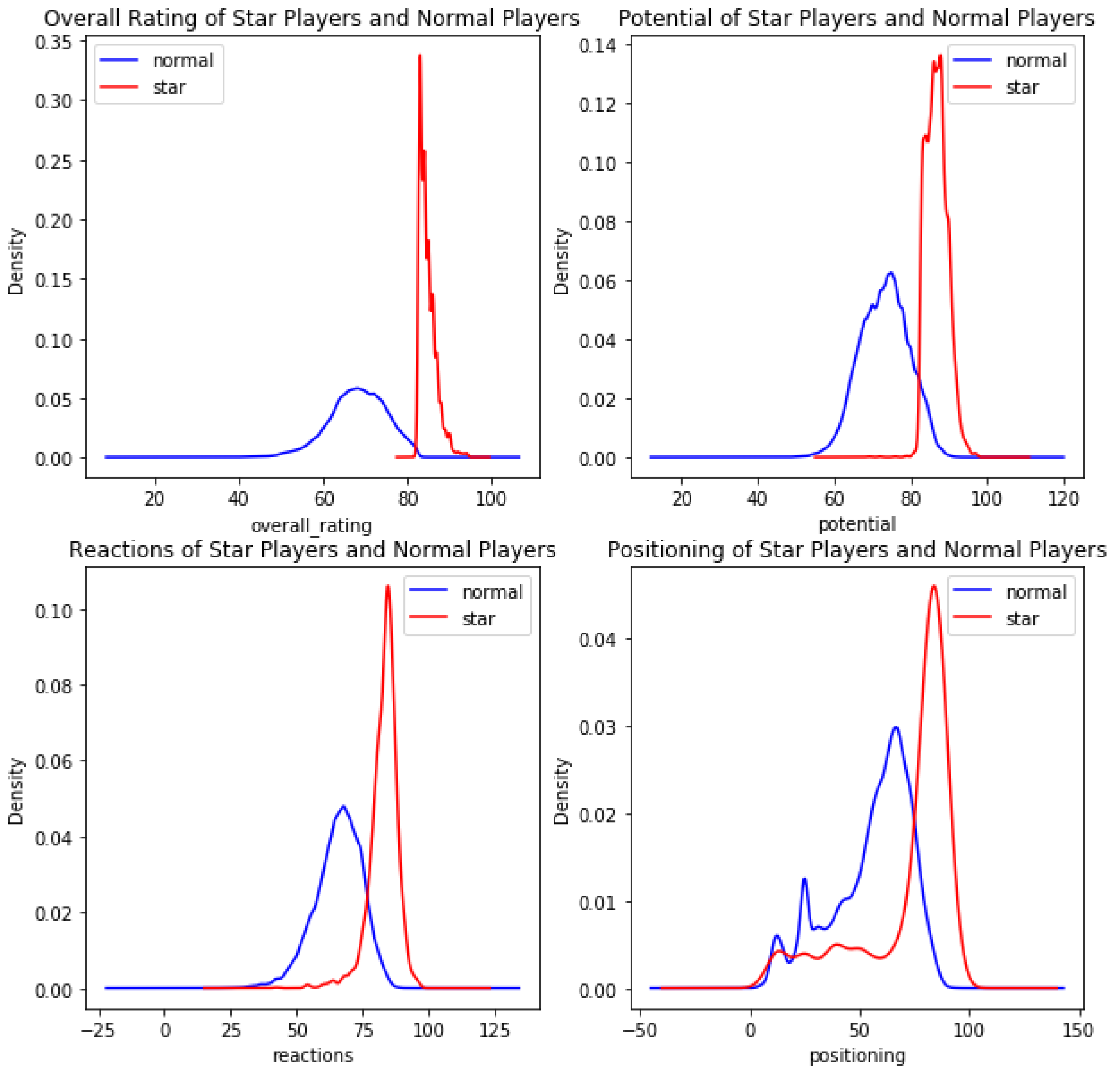
```
In [31]: bin_edges = [0,df_player_full.overall_rating.mean()+2*df_player_full.overall_rating.st
         bin_names = ['normal','star']
         df_player_full['level']=pd.cut(df_player_full['overall_rating'],bin_edges, labels = bi

In [32]: df_player_full.groupby('level').mean().T
         df_star = df_player_full.groupby('level').mean().T.star

In [289]: plt.subplot(2,2,1)

          df_player_full[df_player_full.level == 'normal'].overall_rating.plot.kde(color = 'b'
          df_player_full[df_player_full.level == 'star'].overall_rating.plot.kde(color = 'r',la
          plt.xlabel('overall_rating')
          plt.title('Overall Rating of Star Players and Normal Players')
          plt.legend()
          plt.subplot(2,2,2)
          df_player_full[df_player_full.level == 'normal'].potential.plot.kde(color = 'b',label
          df_player_full[df_player_full.level == 'star'].potential.plot.kde(color = 'r',label =
          plt.xlabel('potential')
          plt.title('Potential of Star Players and Normal Players')
          plt.legend()
          plt.subplot(2,2,3)
          df_player_full[df_player_full.level == 'normal'].reactions.plot.kde(color = 'b',label
          df_player_full[df_player_full.level == 'star'].reactions.plot.kde(color = 'r',label =
          plt.xlabel('reactions')
          plt.title('Reactions of Star Players and Normal Players')
          plt.legend()
          plt.subplot(2,2,4)
          df_player_full[df_player_full.level == 'normal'].positioning.plot.kde(color = 'b',lab
          df_player_full[df_player_full.level == 'star'].positioning.plot.kde(color = 'r',label
          plt.xlabel('positioning')
          plt.title('Positioning of Star Players and Normal Players')
          plt.legend()

Out[289]: <matplotlib.legend.Legend at 0x1a3ab92c18>
```



From the table above, we found that height and weight are not an important features to be a star player. Body fitness is not a way to predict soccer players' overall performance. I found that the above three features in the plots, `potential`, `reactions` and `positioning` are the most distinguishable features. Since the sample size of normal soccer players is much higher than star soccer players, so histagram is not a good plot to show the difference between these groups. I used `plot.kde` to show probability density function. You can see the density shape of all plots are still similar to bell shapes and peaks of `star` and `normal` group are far from each other, so these three features are great features for people to predict a soccer player's overall_rating.

1.1.4 Research Question 2 Do star players make a huge difference to match results? (Cristiano Ronaldo Specifically)

My boyfriend is a big fan of Cristiano Ronaldo, so when he knew I am going to work on this project, he asked me to research this question for him. He wants to know how powerful Cristiano Ronaldo is and how he can improve the performance of a team.

First, I want to do more research about Cristiano Ronaldo.

```
In [37]: df_ronaldo = pd.DataFrame([df_player_full[df_player_full.player_name == 'Cristiano Ro
                                   index = ['Ronaldo','Star Player Avg'])
         df_ronaldo.T
```

Out[37]:

	Ronaldo	Star Player Avg
id	33343.00	93448.109291
player_fifa_api_id	20801.00	119766.052632
player_api_id	30893.00	48608.437701
overall_rating	91.28	84.827605
potential	93.48	86.859023
crossing	83.88	66.740870
finishing	91.12	65.632922
heading_accuracy	85.52	65.759130
short_passing	82.28	75.732546
dribbling	92.64	72.385875
free_kick_accuracy	81.64	64.106069
long_passing	71.72	69.623523
ball_control	93.96	77.223953
acceleration	91.64	75.583244
sprint_speed	93.76	75.215897
reactions	88.16	82.922395
shot_power	92.76	73.809613
stamina	87.60	74.165951
strength	78.68	73.113319
long_shots	89.88	67.735768
aggression	61.28	67.037863
interceptions	35.64	60.352846
positioning	86.48	71.954887
penalties	83.60	72.106337
marking	22.12	45.798067
standing_tackle	30.84	51.929914
gk_diving	7.48	17.213749
gk_handling	12.96	19.016380
gk_kicking	28.44	29.403867
gk_positioning	15.16	19.122718
gk_reflexes	12.76	19.771214
height	185.42	181.780516
weight	176.00	171.952739

His average overall rating 91.28, which is super high. All his performance matrices are much high than the top 5% players' average. For example, `potential`, `crossing`, `finishing` and `heading_accuracy`... etc. No doubt he is one of the most legendary player in the history.

Let's take a look at his winning percentage of all his matches by searching for his `player_api_id` in the match player list.

```
In [38]: df_match.head().T
```

Out[38]:

```
                                   145                     153  \
country_name                   Belgium                 Belgium
country_id                           1                       1
league_name     Belgium Jupiler League  Belgium Jupiler League
id                                 146                     154
season                       2008/2009               2008/2009
date               2009-02-27 00:00:00     2009-03-08 00:00:00
match_api_id                    493017                  493025
home_team_api_id                  8203                    9984
away_team_api_id                  9987                    8342
home_team_goal                       2                       1
away_team_goal                       1                       3
home_player_1                    38327                   36835
home_player_2                    67950                   37047
home_player_3                    67958                   37021
home_player_4                    67959                   37051
home_player_5                    37112                  104386
home_player_6                    36393                   32863
home_player_7                   148286                   37957
home_player_8                    67898                   37909
home_player_9                   164352                   38357
home_player_10                   38801                   37065
home_player_11                   26502                   78462
away_player_1                    37937                   37990
away_player_2                    38293                   21812
away_player_3                   148313                   11736
away_player_4                   104411                   37858
away_player_5                   148314                   38366
away_player_6                    37202                   37983
away_player_7                    43158                   39578
away_player_8                     9307                   38336
away_player_9                    42153                   52280
away_player_10                   32690                   27423
away_player_11                   38782                   38440
result                             Win                    Lose

                                   155                     162  \
country_name                   Belgium                 Belgium
country_id                           1                       1
league_name     Belgium Jupiler League  Belgium Jupiler League
id                                 156                     163
season                       2008/2009               2008/2009
date               2009-03-07 00:00:00     2009-03-13 00:00:00
match_api_id                    493027                  493034
home_team_api_id                  8635                    8203
away_team_api_id                 10000                    8635
home_team_goal                       2                       2
away_team_goal                       0                       1
```

home_player_1	34480	38327
home_player_2	38388	67950
home_player_3	26458	67958
home_player_4	13423	38801
home_player_5	38389	67898
home_player_6	30949	37112
home_player_7	38393	67959
home_player_8	38253	148286
home_player_9	38383	164352
home_player_10	38778	33657
home_player_11	37069	26502
away_player_1	37900	34480
away_player_2	37886	38388
away_player_3	37903	38389
away_player_4	37889	31316
away_player_5	94030	164694
away_player_6	37893	30949
away_player_7	37981	38378
away_player_8	131531	38383
away_player_9	130027	38393
away_player_10	38231	38253
away_player_11	131530	37069
result	Win	Win

	168
country_name	Belgium
country_id	1
league_name	Belgium Jupiler League
id	169
season	2008/2009
date	2009-03-14 00:00:00
match_api_id	493040
home_team_api_id	10000
away_team_api_id	9999
home_team_goal	0
away_team_goal	0
home_player_1	37900
home_player_2	37886
home_player_3	37100
home_player_4	37903
home_player_5	37889
home_player_6	37893
home_player_7	37981
home_player_8	131531
home_player_9	131530
home_player_10	38231
home_player_11	130027
away_player_1	38318

```
away_player_2                        38247
away_player_3                        16387
away_player_4                        94288
away_player_5                        94284
away_player_6                        45832
away_player_7                        26669
away_player_8                        33671
away_player_9                       163670
away_player_10                       37945
away_player_11                       33622
result                                 Tie
```

In [39]:
```
df_match.loc[df_match.home_team_goal > df_match.away_team_goal, 'result'] = 'Win'
df_match.loc[df_match.home_team_goal < df_match.away_team_goal, 'result'] = 'Lose'
df_match.loc[df_match.home_team_goal == df_match.away_team_goal, 'result'] = 'Tie'
```

In [40]:
```
df_ronaldo_home = df_match[(df_match.home_player_1 == 30893) |
         (df_match.home_player_2 == 30893) |
         (df_match.home_player_3 == 30893) |
         (df_match.home_player_4 == 30893) |
         (df_match.home_player_5 == 30893) |
         (df_match.home_player_6 == 30893) |
         (df_match.home_player_7 == 30893) |
         (df_match.home_player_8 == 30893) |
         (df_match.home_player_9 == 30893) |
         (df_match.home_player_10 == 30893) |
         (df_match.home_player_11 == 30893)]

df_home_sum = df_ronaldo_home.groupby('result').id.count()
df_home_sum
```

Out[40]:
```
result
Lose      8
Tie      10
Win     107
Name: id, dtype: int64
```

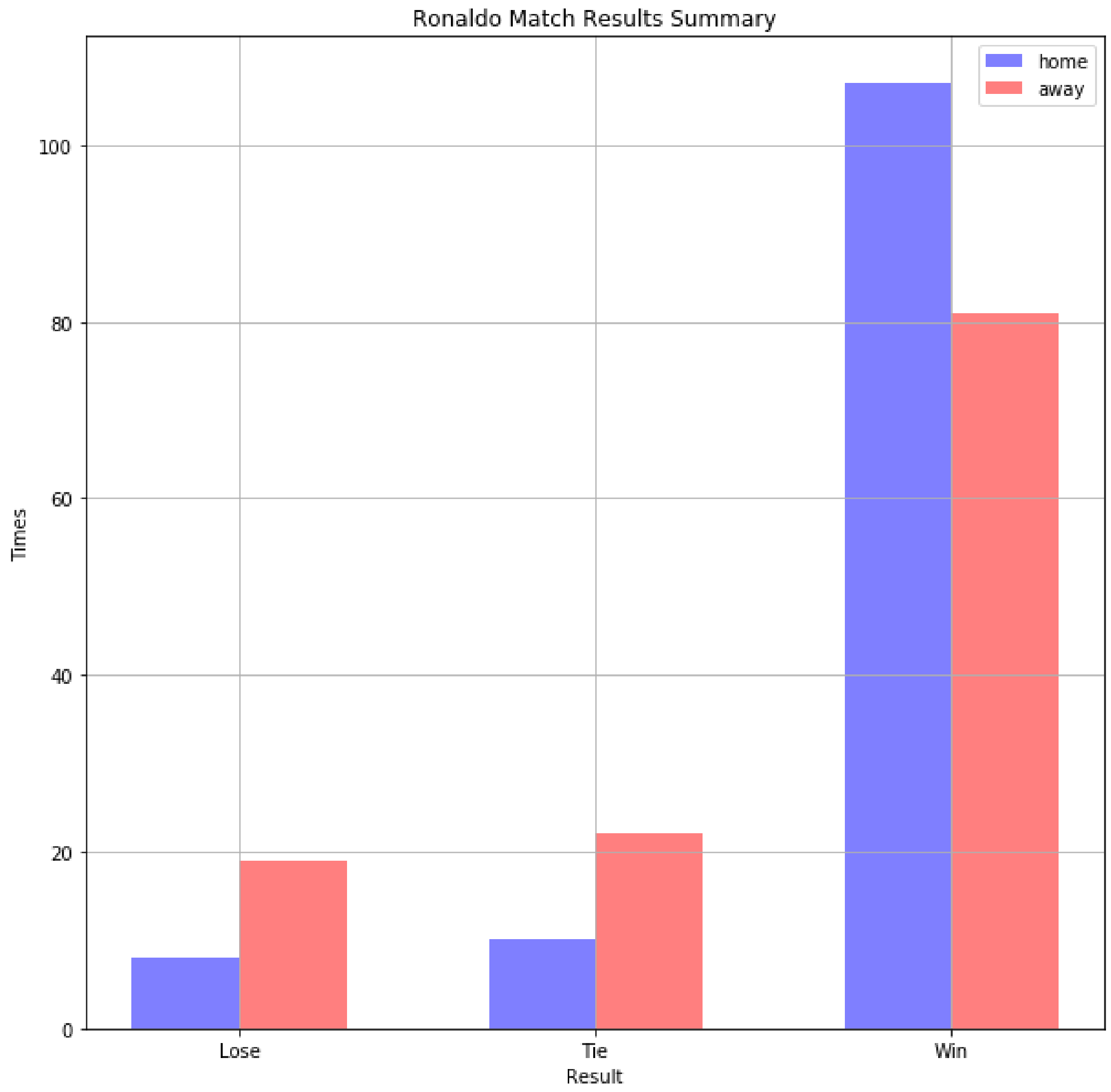
In [42]:
```
df_ronaldo_away = df_match[(df_match.away_player_1 == 30893) |
         (df_match.away_player_2 == 30893) |
         (df_match.away_player_3 == 30893) |
         (df_match.away_player_4 == 30893) |
         (df_match.away_player_5 == 30893) |
         (df_match.away_player_6 == 30893) |
         (df_match.away_player_7 == 30893) |
         (df_match.away_player_8 == 30893) |
         (df_match.away_player_9 == 30893) |
         (df_match.away_player_10 == 30893) |
         (df_match.away_player_11 == 30893)]
```

```
df_away_sum = df_ronaldo_away.groupby('result').id.count()
df_away_sum.rename(index = {'Lose':'Win','Win':'Lose'}, inplace = True)
df_ronaldo_result = pd.DataFrame([df_home_sum,df_away_sum],index = ['home','away'])
```

In [58]:
```
def multi_bar_chart(ind,width,df,label,color,size, mul):
    f, ax = plt.subplots(figsize = size)
    plt.bar(ind,df.iloc[0,:]*mul[0],width,label = label[0], color = color[0],alpha = (
    plt.bar(ind+width,df.iloc[1,:]*mul[1],width,label = label[1],color = color[1], alp
    plt.legend()
    plt.xticks(ind+width/2, labels = ['Lose','Tie','Win'])
    plt.xlabel('Result')
    plt.ylabel('Times')
    plt.grid(True)


multi_bar_chart(ind = np.arange(3),width = 0.3, df= df_ronaldo_result, label = ['home
plt.title('Ronaldo Match Results Summary')
```

Out[58]:
```
Text(0.5, 1.0, 'Ronaldo Match Results Summary')
```



Within all games Ronaldo played from 2008 to 2016, he won most of the games and his winning rate of home games is high than away games.

```
In [47]: pd.DataFrame([df_ronaldo_home.groupby(['home_team_api_id','season']).id.count(), df_ro
                      index = ['Home','Away'])
```

```
Out[47]: home_team_api_id     8633                                                          \
         season           2009/2010 2010/2011 2011/2012 2012/2013 2013/2014 2014/2015
         Home                    11        14        19        16        13        16
         Away                    12        15        18        14        16        17

         home_team_api_id               10260
         season           2015/2016 2008/2009
         Home                    19        17
         Away                    17        13
```

From the data, I know he has served two teams during 2008 to 2016 time period, so I am going to compare the winning rate of 2008/2009 of `home_team_api_id` = 10260 and the remaining seasons of `home_team_api_id` = 8633.

```
In [48]: Ronaldo_2008 = df_ronaldo_home[df_ronaldo_home.season == '2008/2009'].groupby('result

In [49]: Ronaldo_09to16 = df_ronaldo_home[df_ronaldo_home.season != '2008/2009'].groupby('resul

In [50]: pd.DataFrame([Ronaldo_2008/Ronaldo_2008.sum(),Ronaldo_09to16/Ronaldo_09to16.sum()],inc
```

```
Out[50]: result             Lose       Tie       Win
         Team id 10260  0.300000  0.133333  0.566667
         Team id 8633   0.368664  0.129032  0.502304
```

His winning rate while serving different teams are similar, so I won't treat team as an important factor and have separate analysis for the two scienarios. The above plot `Ronaldo Match Results Summary` can represent his performance of the whole time period.

```
In [67]: df_no_ronaldo_home = df_match[(df_match.home_player_1 != 30893) &
                   (df_match.home_player_2 != 30893) &
                   (df_match.home_player_3 != 30893) &
                   (df_match.home_player_4 != 30893) &
                   (df_match.home_player_5 != 30893) &
                   (df_match.home_player_6 != 30893) &
                   (df_match.home_player_7 != 30893) &
                   (df_match.home_player_8 != 30893) &
                   (df_match.home_player_9 != 30893) &
                   (df_match.home_player_10 != 30893) &
                   (df_match.home_player_11 != 30893) &
                   (((df_match.home_team_api_id == 10260) &
                   (df_match.season == '2008/2009')) |
                   ((df_match.home_team_api_id == 8633) &
                   (df_match.season != '2008/2009')))]

In [66]: df_no_ronaldo_away = df_match[(df_match.away_player_1 != 30893) &
                   (df_match.away_player_2 != 30893) &
                   (df_match.away_player_3 != 30893) &
                   (df_match.away_player_4 != 30893) &
                   (df_match.away_player_5 != 30893) &
                   (df_match.away_player_6 != 30893) &
                   (df_match.away_player_7 != 30893) &
                   (df_match.away_player_8 != 30893) &
                   (df_match.away_player_9 != 30893) &
                   (df_match.away_player_10 != 30893) &
                   (df_match.away_player_11 != 30893) &
                   (((df_match.away_team_api_id == 10260) &
                   (df_match.season == '2008/2009')) |
                   ((df_match.away_team_api_id == 8633) &
                   (df_match.season != '2008/2009')))]
```

In [53]:
```
df_no_ronaldo_home_result = df_no_ronaldo_home.groupby('result').id.count()
```

In [54]:
```
df_no_ronaldo_away_result = df_no_ronaldo_away.groupby('result').id.count().rename(inc
```

In [55]:
```
df_no_ronaldo_result = pd.DataFrame([df_no_ronaldo_home_result , df_no_ronaldo_away_re
df_no_ronaldo_result
```

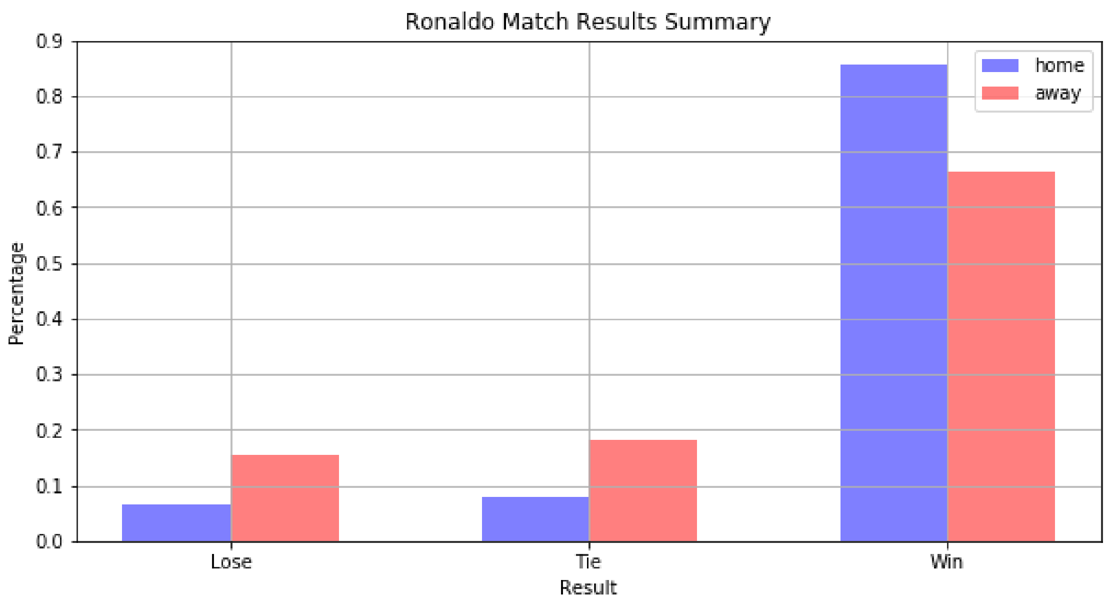
Out[55]:

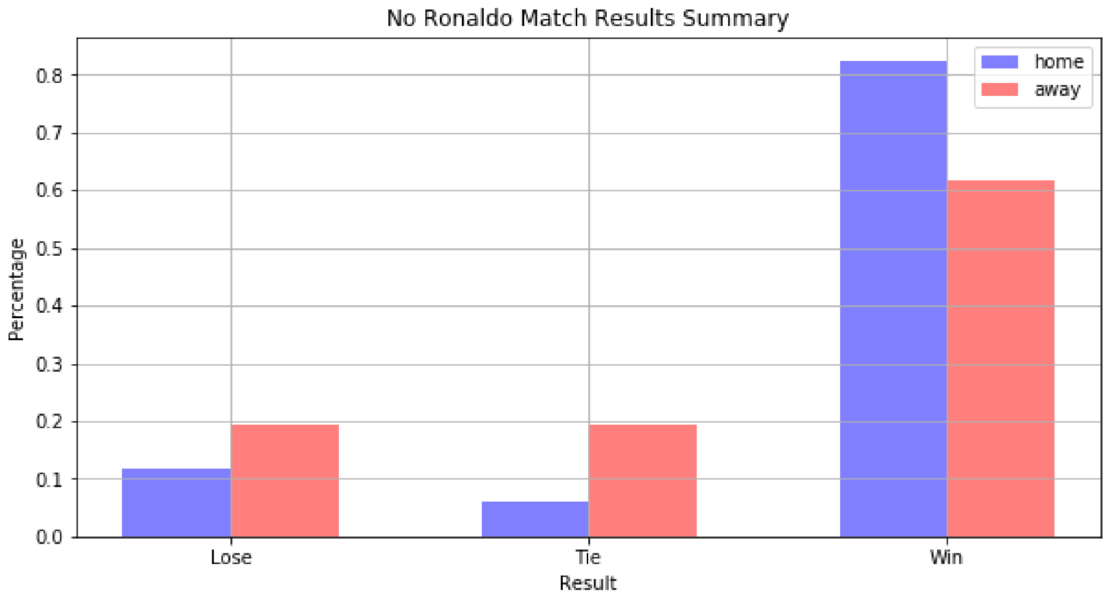
	Lose	Tie	Win
home	2	1	14
away	5	5	16

In [62]:
```
multi_bar_chart(ind=np.arange(3),width=0.3,
                df=df_ronaldo_result,label=['home','away'],
                color=['b','r'],size = (10,5),
                mul = [1/df_ronaldo_result.iloc[0,:].sum(),1/df_ronaldo_result.iloc[1
plt.title('Ronaldo Match Results Summary')
plt.xlabel('Result')
plt.ylabel('Percentage')

multi_bar_chart(ind=np.arange(3),width=0.3,
                df=df_no_ronaldo_result,label=['home','away'],
                color=['b','r'],size = (10,5),
                mul = [1/df_no_ronaldo_result.iloc[0,:].sum(),1/df_no_ronaldo_result.i
plt.title('No Ronaldo Match Results Summary')
plt.xlabel('Result')
plt.ylabel('Percentage')
```

Out[62]: Text(0, 0.5, 'Percentage')





1.1.5 Research Question 3 3. Are match results related to the sum of overall rating of players?

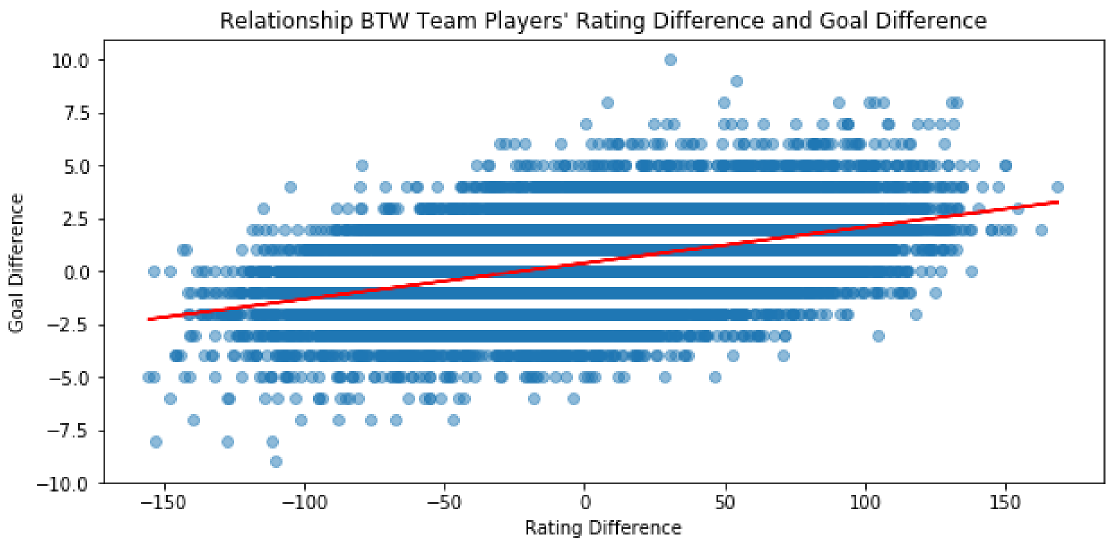
```
In [64]: df_rating = df_player_full.groupby('player_api_id').overall_rating.mean()
         df_rating_dict = df_rating.to_dict()
```

```
In [65]: df_match['home_1']=df_match.home_player_1.map(df_rating_dict)
         df_match['home_2']=df_match.home_player_2.map(df_rating_dict)
         df_match['home_3']=df_match.home_player_3.map(df_rating_dict)
         df_match['home_4']=df_match.home_player_4.map(df_rating_dict)
         df_match['home_5']=df_match.home_player_5.map(df_rating_dict)
         df_match['home_6']=df_match.home_player_6.map(df_rating_dict)
         df_match['home_7']=df_match.home_player_7.map(df_rating_dict)
         df_match['home_8']=df_match.home_player_8.map(df_rating_dict)
         df_match['home_9']=df_match.home_player_9.map(df_rating_dict)
         df_match['home_10']=df_match.home_player_10.map(df_rating_dict)
         df_match['home_11']=df_match.home_player_11.map(df_rating_dict)
         df_match['away_1']=df_match.away_player_1.map(df_rating_dict)
         df_match['away_2']=df_match.away_player_2.map(df_rating_dict)
         df_match['away_3']=df_match.away_player_3.map(df_rating_dict)
         df_match['away_4']=df_match.away_player_4.map(df_rating_dict)
         df_match['away_5']=df_match.away_player_5.map(df_rating_dict)
         df_match['away_6']=df_match.away_player_6.map(df_rating_dict)
         df_match['away_7']=df_match.away_player_7.map(df_rating_dict)
         df_match['away_8']=df_match.away_player_8.map(df_rating_dict)
         df_match['away_9']=df_match.away_player_9.map(df_rating_dict)
         df_match['away_10']=df_match.away_player_10.map(df_rating_dict)
         df_match['away_11']=df_match.away_player_11.map(df_rating_dict)
```

```
df_match['home_rating']=df_match.iloc[:,34:44].sum(axis =1)
df_match['away_rating']=df_match.iloc[:,45:55].sum(axis =1)
df_match['goal_diff']=df_match.home_team_goal - df_match.away_team_goal
df_match['rating_diff']=df_match.home_rating - df_match.away_rating
```

In [292]:
```
b, m = polyfit(df_match.rating_diff, df_match.goal_diff, 1)
plt.subplots(figsize = (10,10))
plt.subplot(2,1,1)
plt.scatter(df_match.rating_diff,df_match.goal_diff, alpha = 0.5)
plt.xlabel('Rating Difference')
plt.ylabel('Goal Difference')
plt.title("Relationship BTW Team Players' Rating Difference and Goal Difference")
plt.plot(df_match.rating_diff, b + m * df_match.rating_diff, 'r-')
```

Out[292]:
```
[<matplotlib.lines.Line2D at 0x1a3a3b6198>]
```



In [276]:
```
np.corrcoef(df_match.rating_diff,df_match.goal_diff)
```

Out[276]:
```
array([[1.        , 0.45046925],
       [0.45046925, 1.        ]])
```

There is a linear relationship between goal difference and overall rating difference with 0.45 correlation coefficient. Although from the scatter plot, we can see the oval shape spreading along the line, we can still say that the sum of overall rating of match players is related to the match goal difference. The trend on the plot is still following the red line.

Conclusions

Here is my summarize of the three questions I am interested in investigating:

1. What are the most important features to distinguish a star player from a normal player?
   I looked at the correlation coefficients and looked at the average values of all features. My analysis shows that a star player must have high potential, fast reaction speed and strong positioning.

Limitations: I have limited understanding of soccer player statistics, so I deleted around 15 columns from the original dataset. If I've done more research of all stats, I might be able to have a more accurate conclusion of important features for star players.

2. Do star players make a huge difference to match results? (Cristiano Ronaldo specifically) By comparing all matches Ronaldo played in the dataset and those matches of his team but without him on the feild, I found that Ronaldo only slightly increased the winning rate of his team. (Increase from 82% to 85% for home game and 61% to 66% for away game) However, since the sample size of matches without Ronaldo is small, the result might not be accurate. We need more matches without Ronaldo result to support this conclusion.

Limitations: Ronaldo played almost all matches. For example, he only missed 1 home game in the season 2008/2009. I don't have enough data points to strongly support my conclusion that matches have similar results with or without him.

3. Are match results related to the sum of overall rating of players? The difference of sum of players' overall rating has linear relationship with goals difference between the two teams. The correlation coefficient is 0.45, which shows the linear relationship but not super strong.

Limitations: I use the average of overall rating of players for the total rating of a team, rather than a player's overall rating of that season. I can also do overall rating query of player of the season, but that would be too complicated and I just want a quick analysis here.