In [6]:

```
import sqlite3
import pandas as pd
from sklearn.tree import DecisionTreeRegressor
from sklearn.linear_model import LinearRegression
from sklearn.model_selection import train_test_split
from sklearn.metrics import mean_squared_error
from math import sqrt
import seaborn as sns
```

In [8]:

```
cnx = sqlite3.connect('/Users/dvora/Documents/Dhaval_GitData/soccer/database.sqli
te')
df = pd.read_sql_query("SELECT * FROM Player_Attributes", cnx)
```

In [9]:

```
cursor = cnx.cursor()
cursor.execute("SELECT name FROM sqlite_master WHERE type='table';")
print(cursor.fetchall())
```

```
[('sqlite_sequence',), ('Player_Attributes',), ('Player',), ('Match',)
, ('League',), ('Country',), ('Team',), ('Team_Attributes',)]
```

In [10]:

```
df.head(2)
```

Out[10]:

	id	player_fifa_api_id	player_api_id	date	overall_rating	potential	preferred_foo
0	1	218353	505942	2016-02-18 00:00:00	67.0	71.0	right
1	2	218353	505942	2015-11-19 00:00:00	67.0	71.0	right

2 rows × 42 columns

In [11]:

```
df.shape
```

Out[11]:

```
(183978, 42)
```

In [12]:

```
df.describe()
```

Out[12]:

	id	player_fifa_api_id	player_api_id	overall_rating	potential
count	183978.00000	183978.000000	183978.000000	183142.000000	183142.000000
mean	91989.50000	165671.524291	135900.617324	68.600015	73.460353
std	53110.01825	53851.094769	136927.840510	7.041139	6.592271
min	1.00000	2.000000	2625.000000	33.000000	39.000000
25%	45995.25000	155798.000000	34763.000000	64.000000	69.000000
50%	91989.50000	183488.000000	77741.000000	69.000000	74.000000
75%	137983.75000	199848.000000	191080.000000	73.000000	78.000000
max	183978.00000	234141.000000	750584.000000	94.000000	97.000000

8 rows × 38 columns

In [13]:

```
df.keys()
```

Out[13]:

```
Index(['id', 'player_fifa_api_id', 'player_api_id', 'date', 'overall_r
ating',
       'potential', 'preferred_foot', 'attacking_work_rate',
       'defensive_work_rate', 'crossing', 'finishing', 'heading_accura
cy',
       'short_passing', 'volleys', 'dribbling', 'curve', 'free_kick_ac
curacy',
       'long_passing', 'ball_control', 'acceleration', 'sprint_speed',
       'agility', 'reactions', 'balance', 'shot_power', 'jumping', 'st
amina',
       'strength', 'long_shots', 'aggression', 'interceptions', 'posit
ioning',
       'vision', 'penalties', 'marking', 'standing_tackle', 'sliding_t
ackle',
       'gk_diving', 'gk_handling', 'gk_kicking', 'gk_positioning',
       'gk_reflexes'],
      dtype='object')
```

In [14]:

```
df.columns
```

Out[14]:

```
Index(['id', 'player_fifa_api_id', 'player_api_id', 'date', 'overall_r
ating',
       'potential', 'preferred_foot', 'attacking_work_rate',
       'defensive_work_rate', 'crossing', 'finishing', 'heading_accura
cy',
       'short_passing', 'volleys', 'dribbling', 'curve', 'free_kick_ac
curacy',
       'long_passing', 'ball_control', 'acceleration', 'sprint_speed',
       'agility', 'reactions', 'balance', 'shot_power', 'jumping', 'st
amina',
       'strength', 'long_shots', 'aggression', 'interceptions', 'posit
ioning',
       'vision', 'penalties', 'marking', 'standing_tackle', 'sliding_t
ackle',
       'gk_diving', 'gk_handling', 'gk_kicking', 'gk_positioning',
       'gk_reflexes'],
      dtype='object')
```

In [15]:

```
df.isnull().sum(axis=0)
```

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
dtype: int64
```

In [16]:

```
df = df.dropna()
```

In [17]:

```
df.isnull().sum(axis=0)
```

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    0
defensive_work_rate    0
crossing               0
finishing              0
heading_accuracy       0
short_passing          0
volleys                0
dribbling              0
curve                  0
free_kick_accuracy     0
long_passing           0
ball_control           0
acceleration           0
sprint_speed           0
agility                0
reactions              0
balance                0
shot_power             0
jumping                0
stamina                0
strength               0
long_shots             0
aggression             0
interceptions          0
positioning            0
vision                 0
penalties              0
marking                0
standing_tackle        0
sliding_tackle         0
gk_diving              0
gk_handling            0
gk_kicking             0
gk_positioning         0
gk_reflexes            0
dtype: int64
```

In [18]:

```
df['overall_rating'] = df['overall_rating'].astype('int')
df.corr(method='pearson',min_periods=1).transpose().sort_values('overall_rating',
ascending=False)
```

Out[18]:

	id	player_fifa_api_id	player_api_id	overall_rating	poten
overall_rating	-0.003738	-0.278703	-0.328315	1.000000	0.7654
reactions	-0.005740	-0.233465	-0.312538	0.771856	0.5809
potential	0.000837	-0.021252	0.010588	0.765435	1.0000
short_passing	-0.006701	-0.065311	-0.090237	0.458243	0.3825
ball_control	-0.013976	-0.024942	-0.053940	0.443991	0.4018
long_passing	-0.008137	-0.111272	-0.139584	0.434525	0.3431
vision	-0.007928	-0.163099	-0.188087	0.431493	0.3792
shot_power	-0.010371	-0.080175	-0.126514	0.428053	0.3254
penalties	-0.011751	-0.175255	-0.162481	0.392715	0.3152
long_shots	-0.010382	-0.068652	-0.119638	0.392668	0.3130
positioning	-0.015643	-0.078862	-0.105157	0.368978	0.3268
volleys	-0.006916	-0.088726	-0.131262	0.361739	0.3016
curve	-0.019523	-0.052501	-0.099430	0.357566	0.2960
crossing	-0.020231	-0.065631	-0.113365	0.357320	0.2772
dribbling	-0.014784	0.047551	0.015616	0.354191	0.3399
free_kick_accuracy	-0.008396	-0.108735	-0.152683	0.349800	0.2628
finishing	-0.008171	-0.029836	-0.062312	0.330079	0.2878
stamina	-0.010506	0.015277	-0.109958	0.325606	0.2594
aggression	-0.018034	-0.170147	-0.212509	0.322782	0.1621
strength	-0.008954	-0.178351	-0.234866	0.315684	0.1223
heading_accuracy	-0.011781	-0.103500	-0.130282	0.313324	0.2060
jumping	-0.004279	-0.073277	-0.141646	0.258978	0.1745
sprint_speed	-0.011897	0.178343	0.094236	0.253048	0.3406
interceptions	-0.008480	-0.169307	-0.185482	0.249094	0.1632
acceleration	-0.008212	0.178267	0.101536	0.243998	0.3388
agility	-0.000947	0.116309	0.026467	0.239963	0.2937
standing_tackle	-0.012515	-0.071128	-0.086706	0.163986	0.0820
balance	-0.009909	0.008350	0.021300	0.160211	0.2022
marking	-0.010329	-0.075568	-0.089772	0.132185	0.0540
sliding_tackle	-0.011101	-0.055218	-0.073595	0.128054	0.0632

gk_kicking	0.008758	-0.248222	-0.229704	0.028799	0.0922
gk_diving	0.014251	-0.092945	-0.071825	0.027675	-0.012
gk_positioning	0.014015	-0.140925	-0.125525	0.008029	0.0044
gk_reflexes	0.014671	-0.131531	-0.121947	0.007804	0.0049
gk_handling	0.010911	-0.138844	-0.125345	0.006717	0.0058
id	1.000000	0.003744	0.002048	-0.003738	0.0008
player_fifa_api_id	0.003744	1.000000	0.556557	-0.278703	-0.021
player_api_id	0.002048	0.556557	1.000000	-0.328315	0.0105

38 rows × 38 columns

In [19]:

```
df['overall_rating'] = df['overall_rating'].astype('int')
display(df.sort_values('overall_rating', ascending=False).head(10)[['overall_rati
ng', 'reactions', 'potential', 'short_passing', 'long_passing','ball_control', 'v
ision', 'shot_power']])
```

	overall_rating	reactions	potential	short_passing	long_passing	ball_control
102493	94	96.0	97.0	89.0	76.0	96.0
102484	94	92.0	95.0	88.0	79.0	96.0
102494	94	96.0	97.0	89.0	76.0	96.0
102499	94	95.0	96.0	89.0	75.0	97.0
102498	94	95.0	96.0	89.0	75.0	97.0
102497	94	95.0	96.0	89.0	75.0	96.0
102496	94	96.0	97.0	89.0	76.0	96.0
102482	94	92.0	94.0	88.0	79.0	96.0
102483	94	92.0	94.0	88.0	79.0	96.0
102495	94	96.0	97.0	89.0	76.0	96.0

In [20]:

```
feature_columns = ['reactions', 'potential']
X = df[feature_columns]
y = df.overall_rating
lm = LinearRegression()
lm.fit(X,y)
```

Out[20]:

```
LinearRegression(copy_X=True, fit_intercept=True, n_jobs=1, normalize=
False)
```

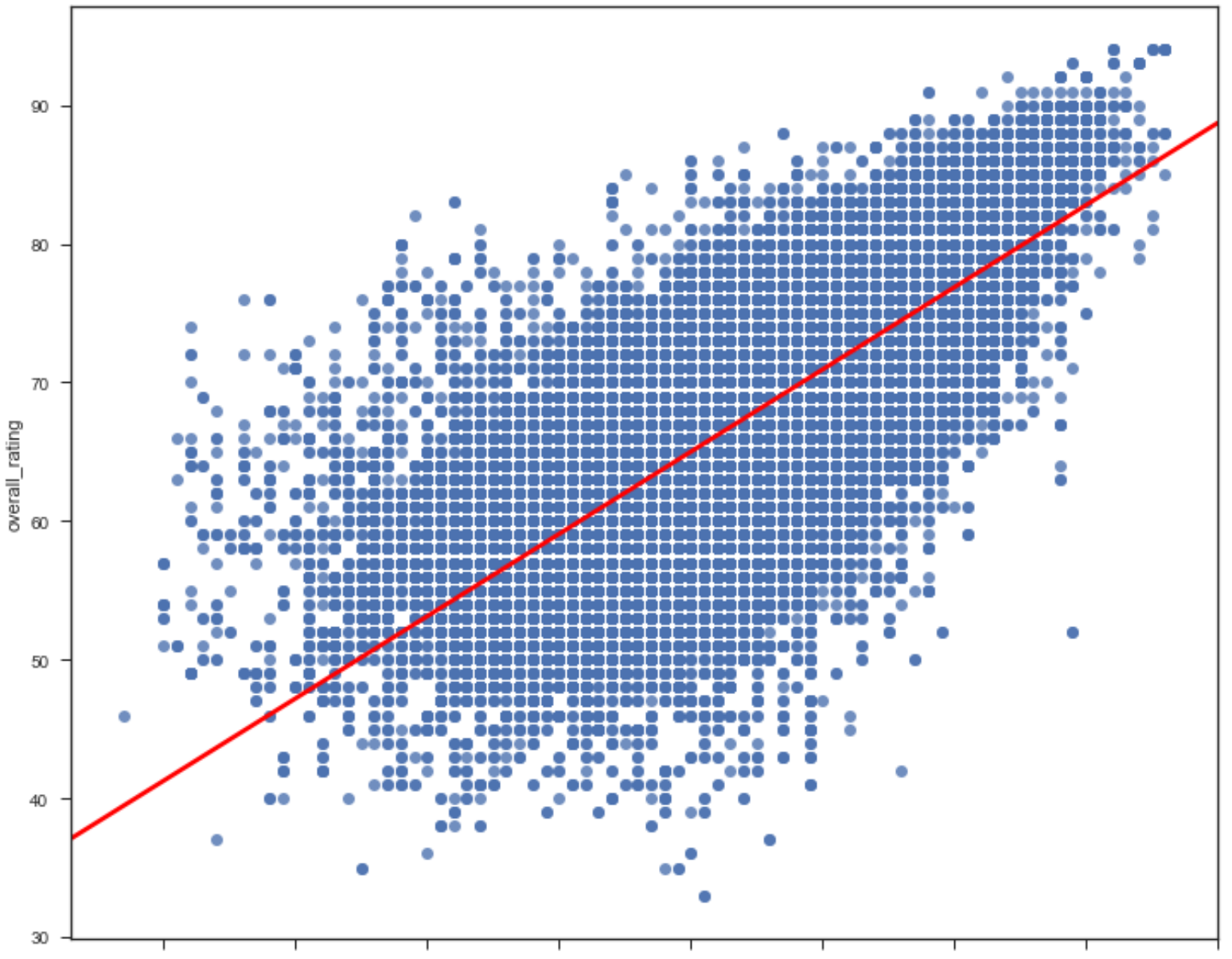
In [21]:

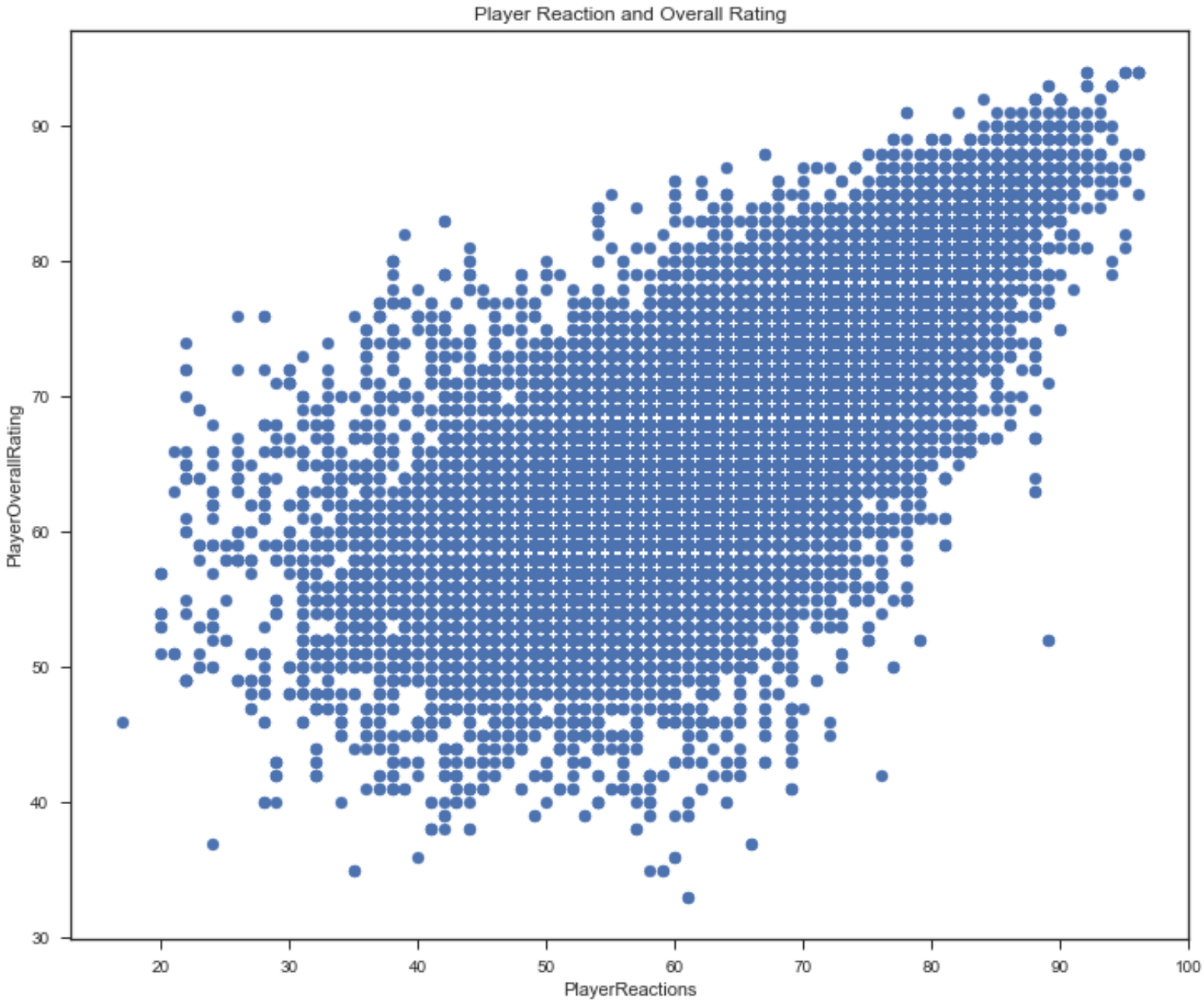
```
print(lm.intercept_)
```

```
5.98720885989
```

In [31]:

```
from matplotlib.pyplot import figure
import  matplotlib.pyplot as plt
figure(num=None, figsize=(12,10), facecolor='w', edgecolor='B')
plt.scatter(df.reactions, df.overall_rating)
plt.xlabel('PlayerReactions')
plt.ylabel('PlayerOverallRating')
plt.title("Player Reaction and Overall Rating")
plt.show()
```


overall_rating
90
80
70
60
50
40
30
20
30
40
50
60
70
80
90
100
reactions



In [30]:

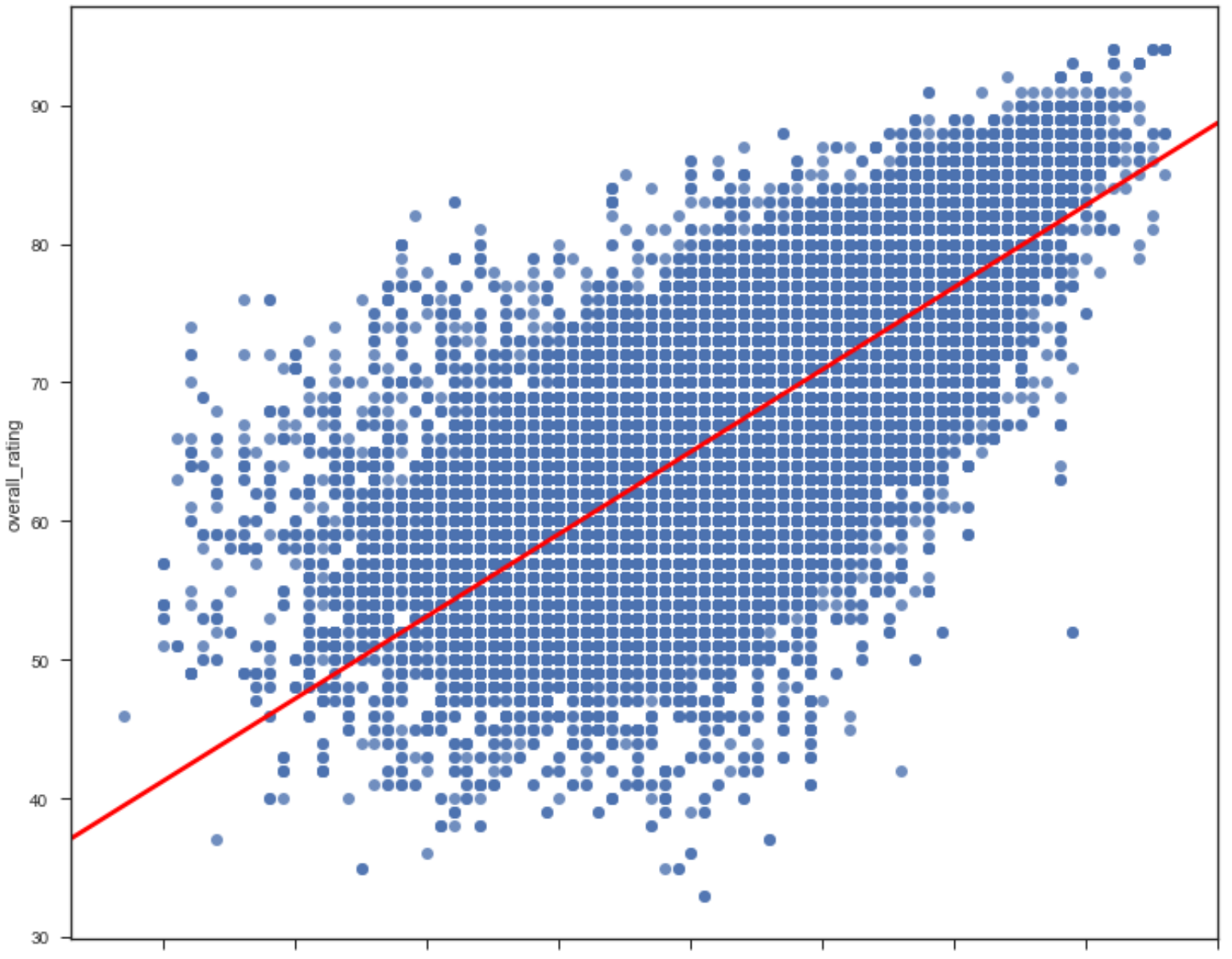
```
sns.set_style('ticks')
fig, ax = plt.subplots()
fig.set_size_inches(12, 10)
sns.regplot('reactions', 'overall_rating', df, line_kws = {"color":"r"}, ci=None)
```

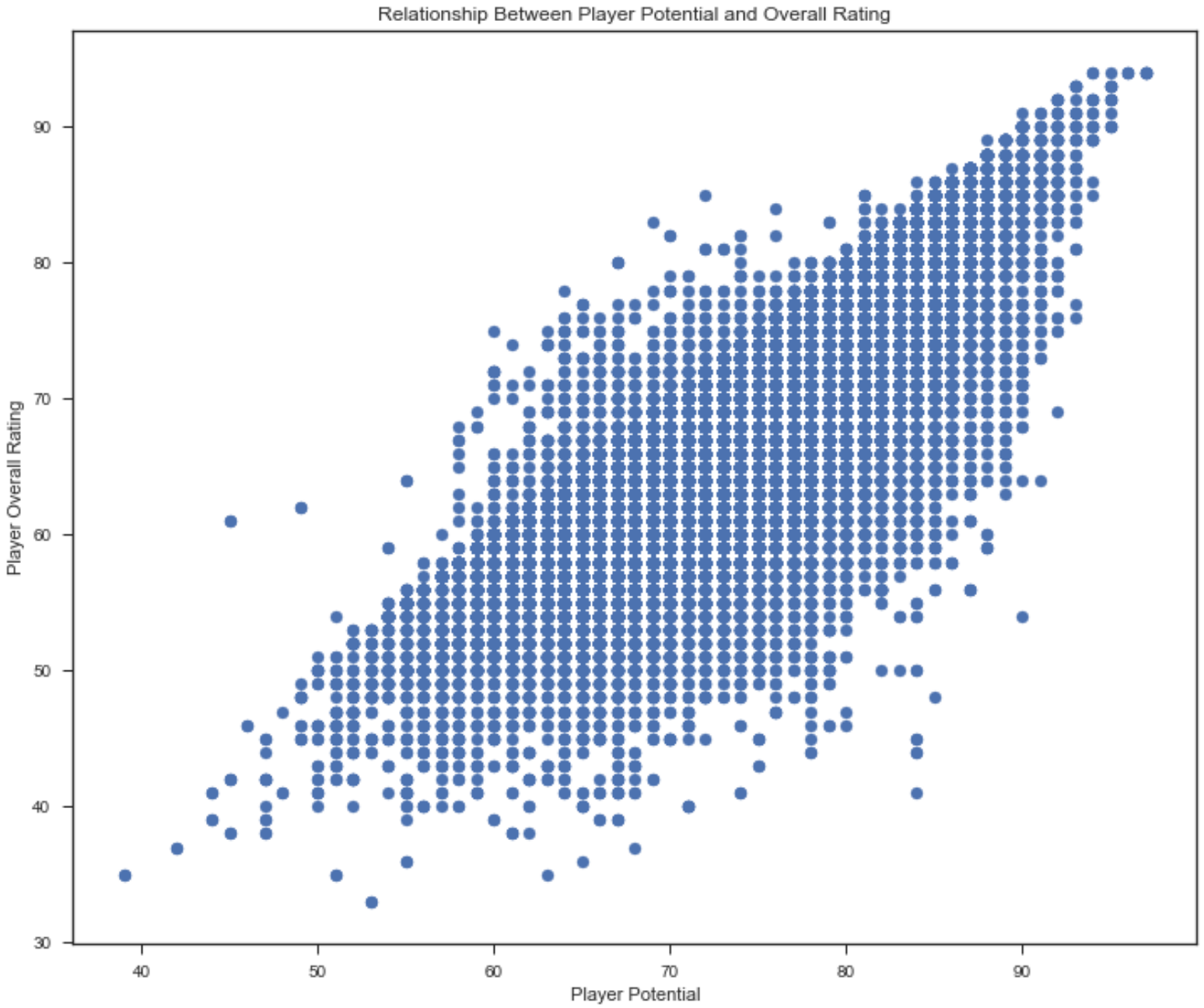
Out[30]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x10e42ff60>
```

In [25]:

```
figure(num=None, figsize=(12,10), facecolor='w', edgecolor='B')
plt.scatter(df.potential, df.overall_rating)
plt.xlabel('Player Potential')
plt.ylabel('Player Overall Rating')
plt.title("Relationship Between Player Potential and Overall Rating")
plt.show()
```


overall_rating
90
80
70
60
50
40
30
20
30
40
50
60
70
80
90
100
reactions



In [26]:

```
import statsmodels.formula.api as smf
lm = smf.ols(formula='overall_rating ~ reactions + potential', data=df).fit()
lm.conf_int()
lm.summary()
```

Out[26]:

OLS Regression Results

Dep. Variable:	overall_rating	**R-squared:**	0.747
Model:	OLS	**Adj. R-squared:**	0.747
Method:	Least Squares	**F-statistic:**	2.669e+05
Date:	Wed, 22 Aug 2018	**Prob (F-statistic):**	0.00
Time:	23:40:27	**Log-Likelihood:**	-4.8349e+05
No. Observations:	180354	**AIC:**	9.670e+05
Df Residuals:	180351	**BIC:**	9.670e+05
Df Model:	2		
Covariance Type:	nonrobust		

	coef	**std err**	**t**	**P>|t|**	**[0.025**	**0.975]**
Intercept	5.9872	0.094	64.015	0.000	5.804	6.171
reactions	0.3795	0.001	339.663	0.000	0.377	0.382
potential	0.5109	0.002	329.122	0.000	0.508	0.514

Omnibus:	21652.406	**Durbin-Watson:**	0.395
Prob(Omnibus):	0.000	**Jarque-Bera (JB):**	56282.876
Skew:	-0.684	**Prob(JB):**	0.00
Kurtosis:	5.370	**Cond. No.**	1.12e+03

In [27]:

```
feature_columns = ['overall_rating', 'reactions', 'potential', 'short_passing', '
long_passing','ball_control', 'vision', 'shot_power']
X = df[feature_columns]
y = df.overall_rating
lm = LinearRegression()
lm.fit(X,y)

print(lm.intercept_)
print(lm.coef_)
```

```
1.84741111298e-12
[  1.00000000e+00  -5.55117304e-16   2.83273210e-16  -2.48737474e-16
   4.90027673e-16  -5.96037288e-16   1.16965327e-16   1.58473429e-16]
```

In [29]:

```
lm = smf.ols(formula='overall_rating ~ reactions + potential + short_passing + lo
ng_passing + ball_control + vision + shot_power', data=df).fit()

lm.conf_int()
lm.summary()
```

Out[29]:

OLS Regression Results

Dep. Variable:	overall_rating	**R-squared:**	0.758
Model:	OLS	**Adj. R-squared:**	0.758
Method:	Least Squares	**F-statistic:**	8.089e+04
Date:	Wed, 22 Aug 2018	**Prob (F-statistic):**	0.00
Time:	23:41:29	**Log-Likelihood:**	-4.7948e+05
No. Observations:	180354	**AIC:**	9.590e+05
Df Residuals:	180346	**BIC:**	9.591e+05
Df Model:	7		
Covariance Type:	nonrobust		

	coef	**std err**	**t**	**P>|t|**	**[0.025**	**0.975]**
Intercept	5.6254	0.092	61.085	0.000	5.445	5.806
reactions	0.3548	0.001	303.481	0.000	0.352	0.357
potential	0.5033	0.002	324.132	0.000	0.500	0.506
short_passing	0.0031	0.002	1.930	0.054	-4.79e-05	0.006
long_passing	0.0510	0.001	51.949	0.000	0.049	0.053
ball_control	-0.0384	0.001	-28.066	0.000	-0.041	-0.036
vision	-0.0161	0.001	-17.660	0.000	-0.018	-0.014
shot_power	0.0456	0.001	56.094	0.000	0.044	0.047

Omnibus:	18238.928	**Durbin-Watson:**	0.390
Prob(Omnibus):	0.000	**Jarque-Bera (JB):**	46323.238
Skew:	-0.591	**Prob(JB):**	0.00
Kurtosis:	5.183	**Cond. No.**	1.92e+03

In [ ]: