segmiller_fp02

April 7, 2022

[1]:
```
#Importing all of the packages necessary for data analysis
import os
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

[2]:
```
#Importing the csv file containing the data and checking to make sure it
#read in correctly
df = pd.read_csv("/home/jupyter-jjsegmil@ncsu.edu/dsc002/madden21_ratings.csv",
                 low_memory=False)
df.head()
```

[2]:
```
       Team           Full Name  Overall Rating Position  Age  Speed  \
0      Rams        Aaron Donald              99       RE   29     82
1    Saints      Michael Thomas              99       WR   27     89
2  Panthers  Christian McCaffrey              99       HB   24     92
3  Patriots      Stephon Gilmore              99       CB   29     92
4    Chiefs      Patrick Mahomes              99       QB   24     81

   Acceleration  Awareness  Agility  Strength  …  Jersey Number  \
0            90         99       86        99  …             99
1            92         99       92        77  …             13
2            93         97       97        72  …             22
3            94         99       94        70  …             24
4            87         97       88        69  …             15

   Total Salary   Signing Bonus         Archetype         Running Style  \
0     101892000        40000000    DE_PowerRusher  Default Stride Loose
1      62750000        35130000    WR_RouteRunner               Default
2      45840000        32190000  HB_ReceivingBack  Short Stride Default
3      33550000        31450000        CB_MantoMan               Default
4       6840000        34420000     QB_Improviser               Default

   Years Pro  Height  Weight  Birthdate        College
0          6      73     280  5/23/1991     Pittsburgh
1          4      75     212   3/3/1993     Ohio State
```

```
2           3      71      205   6/7/1996        Stanford
3           8      73      202  9/19/1990  South Carolina
4           3      75      230  9/17/1995      Texas Tech

[5 rows x 69 columns]
```

[4]:

```
#Creating a new data frame so I can iterate over the first one read in and
#calculate a new field "Adj Overall". I am using conditional logic to apply
#a different formula to each position within the data frame. The "Adj Overall"
#column is calculated using the formula from easports.com but removing
#Awareness and rescaling the other attributes. Adding this option to ignore
#a warning that alerts me I am not writing to the original dataset since␣
 ↪that8888888888
#is not my intention

pd.options.mode.chained_assignment = None
df1 = df
df1["Adj Overall"] = 0

for ind in df.index:
    if df['Position'][ind] == "QB":
        df1["Adj Overall"][ind] = (df.at[ind, "Throw Power"]*.295) + (df.
 ↪at[ind, "Throw Accuracy Short"]*.204) + (df.at[ind, "Throw Accuracy Mid"]*.
 ↪204) + (df.at[ind, "Throw Accuracy Deep"]*.1363) + (df.at[ind, "Play␣
 ↪Action"]*.0454) + (df.at[ind, "Speed"]*.0568) + (df.at[ind, "Agility"]*.
 ↪0227) + (df.at[ind, "Throw On The Run"]*.0227) + (df.at[ind,␣
 ↪"Acceleration"]*.0114)
    elif df["Position"][ind] == "HB":
        df1["Adj Overall"][ind] = (df.at[ind, "Strength"]*.0326) + (df.at[ind,␣
 ↪"Agility"]*.087) + (df.at[ind, "Speed"]*.1413) + (df.at[ind,␣
 ↪"Acceleration"]*.087) + (df.at[ind, "Catching"]*.0435) + (df.at[ind,␣
 ↪"Carrying"]*.1413) + (df.at[ind, "Trucking"]*.087) + (df.at[ind, "Change Of␣
 ↪Direction"]*.087) + (df.at[ind, "Ball Carrier Vision"]*.1413) + (df.at[ind,␣
 ↪"Stiff Arm"]*.0217) + (df.at[ind, "Spin Move"]*.0435) + (df.at[ind, "Juke␣
 ↪Move"]*.0435) + (df.at[ind, "Catch In Traffic"]*.0109) + (df.at[ind, "Short␣
 ↪Route Running"]*.0109) + (df.at[ind, "Medium Route Running"]*.0109) + (df.
 ↪at[ind, "Deep Route Running"]*.0109)
    elif df["Position"][ind] == "FB":
        df1["Adj Overall"][ind] = (df.at[ind, "Strength"]*.0353) + (df.at[ind, "Agility"]*.0353) + (df.at[ind, "Speed"]*.0588) + (df.at[ind, "Acceleration"]*.0588) + (df.at[ind, "Catching"]*.047) + (df.at[ind, "Carrying"]*.0824) + (df.at[ind, "Pass Block Finesse"]*.0275) + (df.at[ind, "Pass Block Power"]*.0275) + (df.at[ind, "Pass Blocking"]*.0275) + (df.at[ind, "Run Block Finesse"]*.0627) + (df.at[ind, "Run Block Power"]*.0627) + (df.at[ind, "Run Block Power"]*.0627) + (df.at[ind, "Trucking"]*.0588) + (df.at[ind, "Change Of Direction"]*.0235) + (df.at[ind, "Ball Carrier Vision"]*.0352) + (df.at[ind, "Stiff Arm"]*.0352) + (df.at[ind, "Impact Blocking"]*.2588)
    elif df["Position"][ind] == "WR":
        df1["Adj Overall"][ind] = (df.at[ind, "Strength"]*.0227) + (df.at[ind, "Agility"]*.0568) + (df.at[ind, "Speed"]*.1363) + (df.at[ind, "Acceleration"]*.0909) + (df.at[ind, "Catching"]*.1363) + (df.at[ind, "Carrying"]*.0454) + (df.at[ind, "Jumping"]*.0454) + (df.at[ind, "Trucking"]*.0114) + (df.at[ind, "Change Of Direction"]*.0454) + (df.at[ind, "Ball Carrier Vision"]*.0114) + (df.at[ind, "Stiff Arm"]*.0114) + (df.at[ind, "Spin Move"]*.0114) + (df.at[ind, "Juke Move"]*.0114) + (df.at[ind, "Spectacular Catch"]*.0454) + (df.at[ind, "Catch In Traffic"]*.0909) + (df.at[ind, "Short Route Running"]*.0454) + (df.at[ind, "Medium Route Running"]*.0454) + (df.at[ind, "Deep Route Running"]*.0454) + (df.at[ind, "Release"]*.0909)
    elif df["Position"][ind] == "TE":
        df1["Adj Overall"][ind] = (df.at[ind, "Strength"]*.0682) + (df.at[ind, "Agility"]*.0454) + (df.at[ind, "Speed"]*.0795) + (df.at[ind, "Acceleration"]*.0454) + (df.at[ind, "Catching"]*.1023) + (df.at[ind, "Carrying"]*.0227) + (df.at[ind, "Pass Block Power"]*.0152) + (df.at[ind, "Pass Block Finesse"]*.0152) + (df.at[ind, "Pass Blocking"]*.0152) + (df.at[ind, "Run Block Power"]*.053) + (df.at[ind, "Run Block Finesse"]*.053) + (df.at[ind, "Run Blocking"]*.053) + (df.at[ind, "Jumping"]*.0227) + (df.at[ind, "Trucking"]*.0114) + (df.at[ind, "Change Of Direction"]*.0114) + (df.at[ind, "Ball Carrier Vision"]*.0227) + (df.at[ind, "Stiff Arm"]*.0114) + (df.at[ind, "Spin Move"]*.0114) + (df.at[ind, "Juke Move"]*.0114) + (df.at[ind, "Impact Blocking"]*.0682) + (df.at[ind, "Spectacular Catch"]*.0454) + (df.at[ind, "Catch In Traffic"]*.1023) + (df.at[ind, "Short Route Running"]*.0303) + (df.at[ind, "Medium Route Running"]*.0303) + (df.at[ind, "Deep Route Running"]*.0303) + (df.at[ind, "Release"]*.0227)
    elif df["Position"][ind] == "LT":
        df1["Adj Overall"][ind] = (df.at[ind, "Strength"]*.1548) + (df.at[ind, "Agility"]*.0119) + (df.at[ind, "Speed"]*.0357) + (df.at[ind, "Acceleration"]*.0357) + (df.at[ind, "Pass Block Power"]*.1429) + (df.at[ind, "Pass Block Finesse"]*.1429) + (df.at[ind, "Pass Blocking"]*.1429) + (df.at[ind, "Run Block Finesse"]*.0794) + (df.at[ind, "Run Block Power"]*.0794) + (df.at[ind, "Run Blocking"]*.0794)  + (df.at[ind, "Impact Blocking"]*.0952)
    elif df["Position"][ind] == "LG":
        df1["Adj Overall"][ind] = (df.at[ind, "Strength"]*.1463) + (df.at[ind,␣
↪"Agility"]*.0244) + (df.at[ind, "Speed"]*.0366) + (df.at[ind,␣
↪"Acceleration"]*.0732) + (df.at[ind, "Pass Block Power"]*.1057) + (df.
↪at[ind, "Pass Block Finesse"]*.1057) + (df.at[ind, "Pass Blocking"]*.1057) +␣
↪(df.at[ind, "Run Block Finesse"]*.0871) + (df.at[ind, "Run Block Power"]*.
↪0871) + (df.at[ind, "Run Blocking"]*.0871)  + (df.at[ind, "Impact␣
↪Blocking"]*.122)
    elif df["Position"][ind] == "C":
        df1["Adj Overall"][ind] = (df.at[ind, "Strength"]*.1463) + (df.at[ind,␣
↪"Agility"]*.0122) + (df.at[ind, "Speed"]*.061) + (df.at[ind,␣
↪"Acceleration"]*.061) + (df.at[ind, "Pass Block Power"]*.1016) + (df.at[ind,␣
↪"Pass Block Finesse"]*.1016) + (df.at[ind, "Pass Blocking"]*.1016) + (df.
↪at[ind, "Run Block Finesse"]*.1016) + (df.at[ind, "Run Block Power"]*.1016)␣
↪+ (df.at[ind, "Run Blocking"]*.1016)  + (df.at[ind, "Impact Blocking"]*.11)
    elif df["Position"][ind] == "RG":
        df1["Adj Overall"][ind] = (df.at[ind, "Strength"]*.1463) + (df.at[ind,␣
↪"Agility"]*.0244) + (df.at[ind, "Speed"]*.0366) + (df.at[ind,␣
↪"Acceleration"]*.0732) + (df.at[ind, "Pass Block Power"]*.1057) + (df.
↪at[ind, "Pass Block Finesse"]*.1057) + (df.at[ind, "Pass Blocking"]*.1057) +␣
↪(df.at[ind, "Run Block Finesse"]*.1016) + (df.at[ind, "Run Block Power"]*.
↪1016) + (df.at[ind, "Run Blocking"]*.1016)  + (df.at[ind, "Impact␣
↪Blocking"]*.0976)
    elif df["Position"][ind] == "RT":
        df1["Adj Overall"][ind] = (df.at[ind, "Strength"]*.1481) + (df.at[ind,␣
↪"Agility"]*.0123) + (df.at[ind, "Speed"]*.037) + (df.at[ind,␣
↪"Acceleration"]*.037) + (df.at[ind, "Pass Block Power"]*.0988) + (df.at[ind,␣
↪"Pass Block Finesse"]*.0988) + (df.at[ind, "Pass Blocking"]*.0988) + (df.
↪at[ind, "Run Block Finesse"]*.1193) + (df.at[ind, "Run Block Power"]*.1193)␣
↪+ (df.at[ind, "Run Blocking"]*.1193)  + (df.at[ind, "Impact Blocking"]*.1111)
    elif df["Position"][ind] == "K":
        df1["Adj Overall"][ind] = (df.at[ind, "Kick Power"]*.45) + (df.at[ind,␣
↪"Kick Accuracy"]*.55)
    elif df["Position"][ind] == "P":
        df1["Adj Overall"][ind] = (df.at[ind, "Kick Power"]*.55) + (df.at[ind,␣
↪"Kick Accuracy"]*.45)
    elif df["Position"][ind] == "LE":
        df1["Adj Overall"][ind] = (df.at[ind, "Strength"]*.0562) + (df.at[ind,␣
↪"Agility"]*.0562) + (df.at[ind, "Speed"]*.1124) + (df.at[ind,␣
↪"Acceleration"]*.1124) + (df.at[ind, "Tackle"]*.1124) + (df.at[ind, "Power␣
↪Moves"]*.1348) + (df.at[ind, "Finesse Moves"]*.1348) + (df.at[ind, "Block␣
↪Shedding"]*.1124) + (df.at[ind, "Pursuit"]*.0337) + (df.at[ind, "Play␣
↪Recognition"]*.1236) + (df.at[ind, "Hit Power"]*.0112)
    elif df["Position"][ind] == "RE":
        df1["Adj Overall"][ind] = (df.at[ind, "Strength"]*.0562) + (df.at[ind,␣
↪"Agility"]*.0562) + (df.at[ind, "Speed"]*.1124) + (df.at[ind,␣
↪"Acceleration"]*.1124) + (df.at[ind, "Tackle"]*.1124) + (df.at[ind, "Power␣
↪Moves"]*.1348) + (df.at[ind, "Finesse Moves"]*.1348) + (df.at[ind, "Block␣
↪Shedding"]*.1124) + (df.at[ind, "Pursuit"]*.0337) + (df.at[ind, "Play␣
↪Recognition"]*.1236) + (df.at[ind, "Hit Power"]*.0112)
    elif df["Position"][ind] == "DT":
        df1["Adj Overall"][ind] = (df.at[ind, "Strength"]*.191) + (df.at[ind,␣
↪"Agility"]*.0337) + (df.at[ind, "Speed"]*.0674) + (df.at[ind,␣
↪"Acceleration"]*.0674) + (df.at[ind, "Tackle"]*.09) + (df.at[ind, "Power␣
↪Moves"]*.1573) + (df.at[ind, "Finesse Moves"]*.1124) + (df.at[ind, "Block␣
↪Shedding"]*.1348) + (df.at[ind, "Pursuit"]*.0225) + (df.at[ind, "Play␣
↪Recognition"]*.1236)
    elif df["Position"][ind] == "LOLB":
        df1["Adj Overall"][ind] = (df.at[ind, "Strength"]*.0689) + (df.at[ind,␣
↪"Agility"]*.023) + (df.at[ind, "Speed"]*.1034) + (df.at[ind,␣
↪"Acceleration"]*.0689) + (df.at[ind, "Tackle"]*.1149) + (df.at[ind, "Power␣
↪Moves"]*.1149) + (df.at[ind, "Finesse Moves"]*.0689) + (df.at[ind, "Block␣
↪Shedding"]*.1034) + (df.at[ind, "Pursuit"]*.0689) + (df.at[ind, "Play␣
↪Recognition"]*.1494) + (df.at[ind, "Man Coverage"]*.0344) + (df.at[ind,␣
↪"Zone Coverage"]*.0575) + (df.at[ind, "Hit Power"]*.023)
    elif df["Position"][ind] == "ROLB":
        df1["Adj Overall"][ind] = (df.at[ind, "Strength"]*.0689) + (df.at[ind,␣
↪"Agility"]*.023) + (df.at[ind, "Speed"]*.1149) + (df.at[ind,␣
↪"Acceleration"]*.0689) + (df.at[ind, "Tackle"]*.1149) + (df.at[ind, "Power␣
↪Moves"]*.0689) + (df.at[ind, "Finesse Moves"]*.1149) + (df.at[ind, "Block␣
↪Shedding"]*.0805) + (df.at[ind, "Pursuit"]*.0689) + (df.at[ind, "Play␣
↪Recognition"]*.1494) + (df.at[ind, "Man Coverage"]*.0344) + (df.at[ind,␣
↪"Zone Coverage"]*.0575) + (df.at[ind, "Hit Power"]*.0345)
    elif df["Position"][ind] == "MLB":
        df1["Adj Overall"][ind] = (df.at[ind, "Strength"]*.0465) + (df.at[ind,␣
↪"Agility"]*.0465) + (df.at[ind, "Speed"]*.093) + (df.at[ind,␣
↪"Acceleration"]*.0465) + (df.at[ind, "Tackle"]*.1744) + (df.at[ind, "Power␣
↪Moves"]*.0232) + (df.at[ind, "Finesse Moves"]*.0232) + (df.at[ind, "Block␣
↪Shedding"]*.1512) + (df.at[ind, "Pursuit"]*.1047) + (df.at[ind, "Play␣
↪Recognition"]*.1628) + (df.at[ind, "Man Coverage"]*.0349) + (df.at[ind,␣
↪"Zone Coverage"]*.0581) + (df.at[ind, "Hit Power"]*.0465)
    elif df["Position"][ind] == "CB":
        df1["Adj Overall"][ind] = (df.at[ind, "Strength"]*.0111) + (df.at[ind,␣
↪"Agility"]*.0444) + (df.at[ind, "Speed"]*.1555) + (df.at[ind,␣
↪"Acceleration"]*.1555) + (df.at[ind, "Catching"]*.0111) + (df.at[ind,␣
↪"Tackle"]*.0333) + (df.at[ind, "Jumping"]*.0444) + (df.at[ind, "Play␣
↪Recognition"]*.1555) + (df.at[ind, "Man Coverage"]*.2) + (df.at[ind, "Zone␣
↪Coverage"]*.1555) + (df.at[ind, "Press"]*.0333)
    elif df["Position"][ind] == "SS":
        df1["Adj Overall"][ind] = (df.at[ind, "Strength"]*.0706) + (df.at[ind,␣
↪"Agility"]*.0471) + (df.at[ind, "Speed"]*.1059) + (df.at[ind,␣
↪"Acceleration"]*.0471) + (df.at[ind, "Tackle"]*.1294) + (df.at[ind,␣
↪"Jumping"]*.0353) + (df.at[ind, "Block Shedding"]*.0235) + (df.at[ind,␣
↪"Pursuit"]*.1059) + (df.at[ind, "Play Recognition"]*.1765) + (df.at[ind,␣
↪"Man Coverage"]*.0588) + (df.at[ind, "Zone Coverage"]*.1529) + (df.at[ind,␣
↪"Hit Power"]*.0444)
    elif df["Position"][ind] == "FS":
        df1["Adj Overall"][ind] = (df.at[ind, "Strength"]*.0349) + (df.at[ind,␣
↪"Agility"]*.0581) + (df.at[ind, "Speed"]*.1279) + (df.at[ind,␣
↪"Acceleration"]*.0581) + (df.at[ind, "Catching"]*.0116) + (df.at[ind,␣
↪"Tackle"]*.1163) + (df.at[ind, "Jumping"]*.0581) + (df.at[ind, "Play␣
↪Recognition"]*.1628) + (df.at[ind, "Man Coverage"]*.0814) + (df.at[ind,␣
↪"Zone Coverage"]*.1977) + (df.at[ind, "Hit Power"]*.0349)
    else:
        pass

df1.head(5)
```

[4]:

```
       Team            Full Name  Overall Rating Position  Age  Speed  \
0      Rams         Aaron Donald              99       RE   29     82
1    Saints       Michael Thomas              99       WR   27     89
2  Panthers  Christian McCaffrey              99       HB   24     92
3  Patriots      Stephon Gilmore              99       CB   29     92
4    Chiefs      Patrick Mahomes              99       QB   24     81

   Acceleration  Awareness  Agility  Strength  …  Total Salary  \
0            90         99       86        99  …     101892000
1            92         99       92        77  …      62750000
2            93         97       97        72  …      45840000
3            94         99       94        70  …      33550000
4            87         97       88        69  …       6840000

    Signing Bonus          Archetype         Running Style  Years Pro  Height  \
0         40000000    DE_PowerRusher  Default Stride Loose          6      73
1         35130000    WR_RouteRunner               Default          4      75
2         32190000  HB_ReceivingBack  Short Stride Default          3      71
3         31450000       CB_MantoMan               Default          8      73
4         34420000     QB_Improviser               Default          3      75

   Weight  Birthdate         College  Adj Overall
0     280  5/23/1991      Pittsburgh      93.9188
1     212   3/3/1993      Ohio State      92.7109
2     205   6/7/1996        Stanford      89.0358
3     202  9/19/1990  South Carolina      94.4856
4     230  9/17/1995      Texas Tech      94.6094
```

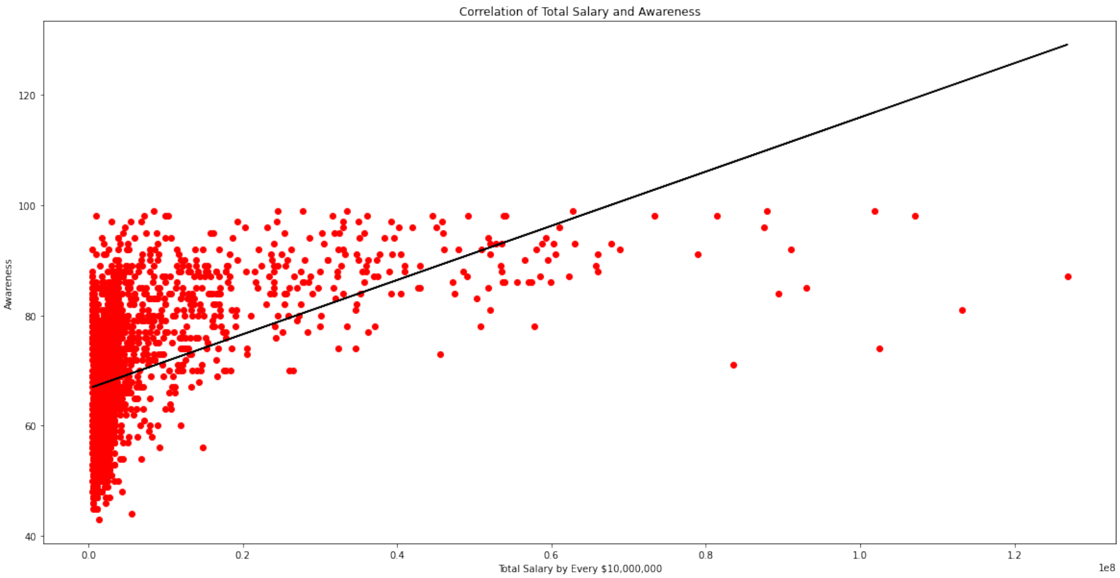
```
[5 rows x 70 columns]
```

[5]:

```
#Creating a scatterplot with a trend line showing the correlation between Total␣
 ↪Salary and Awareness
m, b = np.polyfit(df1["Total Salary"], df1["Awareness"], 1)
fig, ax = plt.subplots(figsize = (20,10))
ax.scatter(df1["Total Salary"], df1["Awareness"], color = "red")
plt.title("Correlation of Total Salary and Awareness")
plt.xlabel("Total Salary by Every $10,000,000")
plt.ylabel("Awareness")
plt.plot(df1["Total Salary"], m*df1["Total Salary"]+b, color = "black")
```

[5]:

```
[<matplotlib.lines.Line2D at 0x7f1f64e39c30>]
```



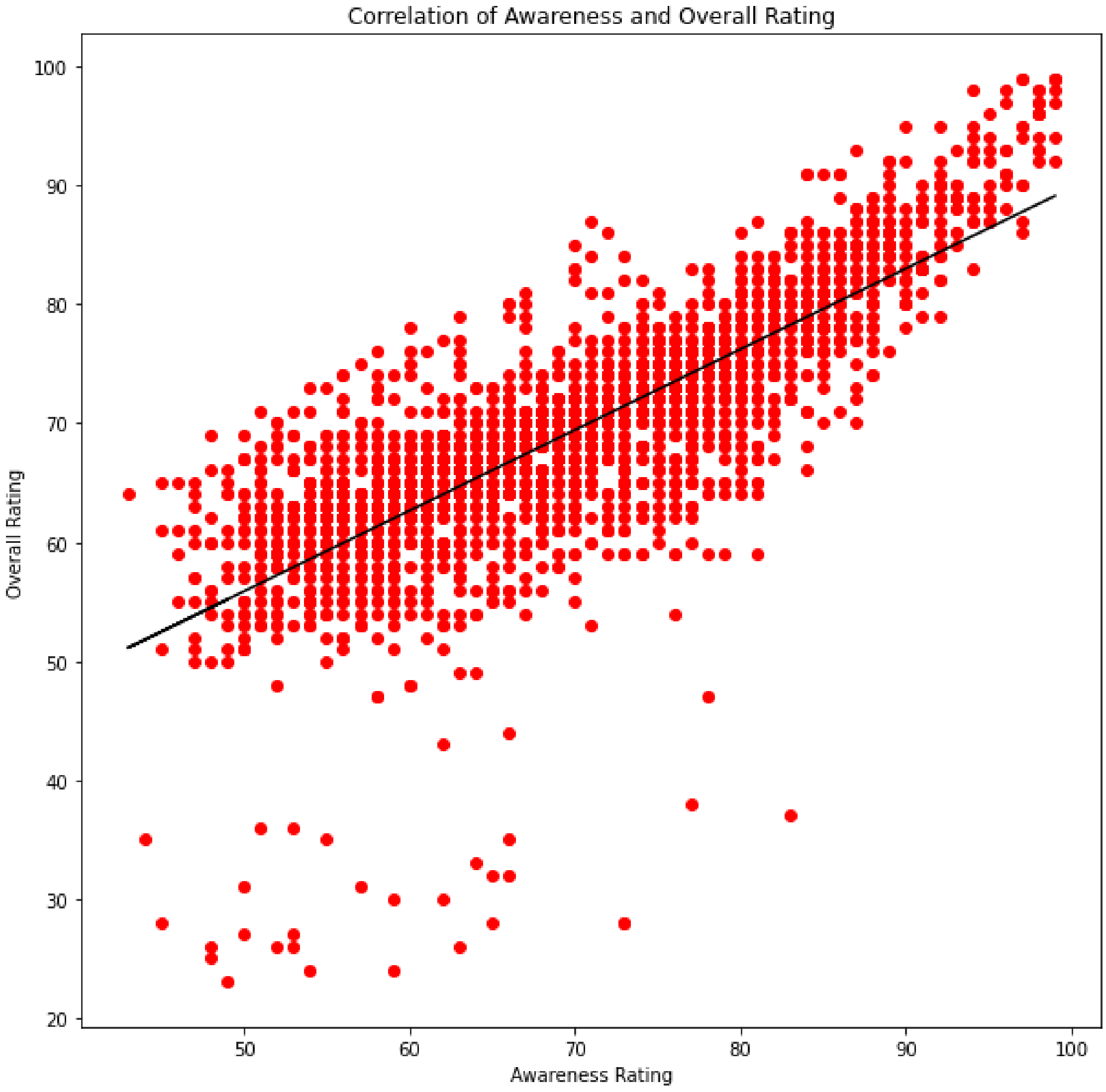
0.1 Plot Interpretation: Total salary is on the x axis here and is being used to predict awareness. Coming into this project I speculated that salary could be an indicator of a higher awareness since players who are highly paid tend to be seen as better even if they may not be. This chart shows that there a strong, positive correlation between the two variables. From this we can state that total salary is correlated with awareness. A sidenote, Madden overalls do not exceed 99, however I let the ylim go past this to show just how high the highly paid players awareness is projected to be.

[7]:

```
#Creating a scatterplot with a trend line showing the correlation between␣
 ↪Awareness and Overall Rating
m, b = np.polyfit(df1["Awareness"], df["Overall Rating"], 1)
```

```
fig, ax = plt.subplots(figsize = (10,10))
ax.scatter(df1["Awareness"], df1["Overall Rating"], color = "red")
plt.title("Correlation of Awareness and Overall Rating")
plt.xlabel("Awareness Rating")
plt.ylabel("Overall Rating")
plt.plot(df1["Awareness"], m*df1["Awareness"]+b, color = "black")
```

```
[7]: [<matplotlib.lines.Line2D at 0x7f1f5c8e1090>]
```



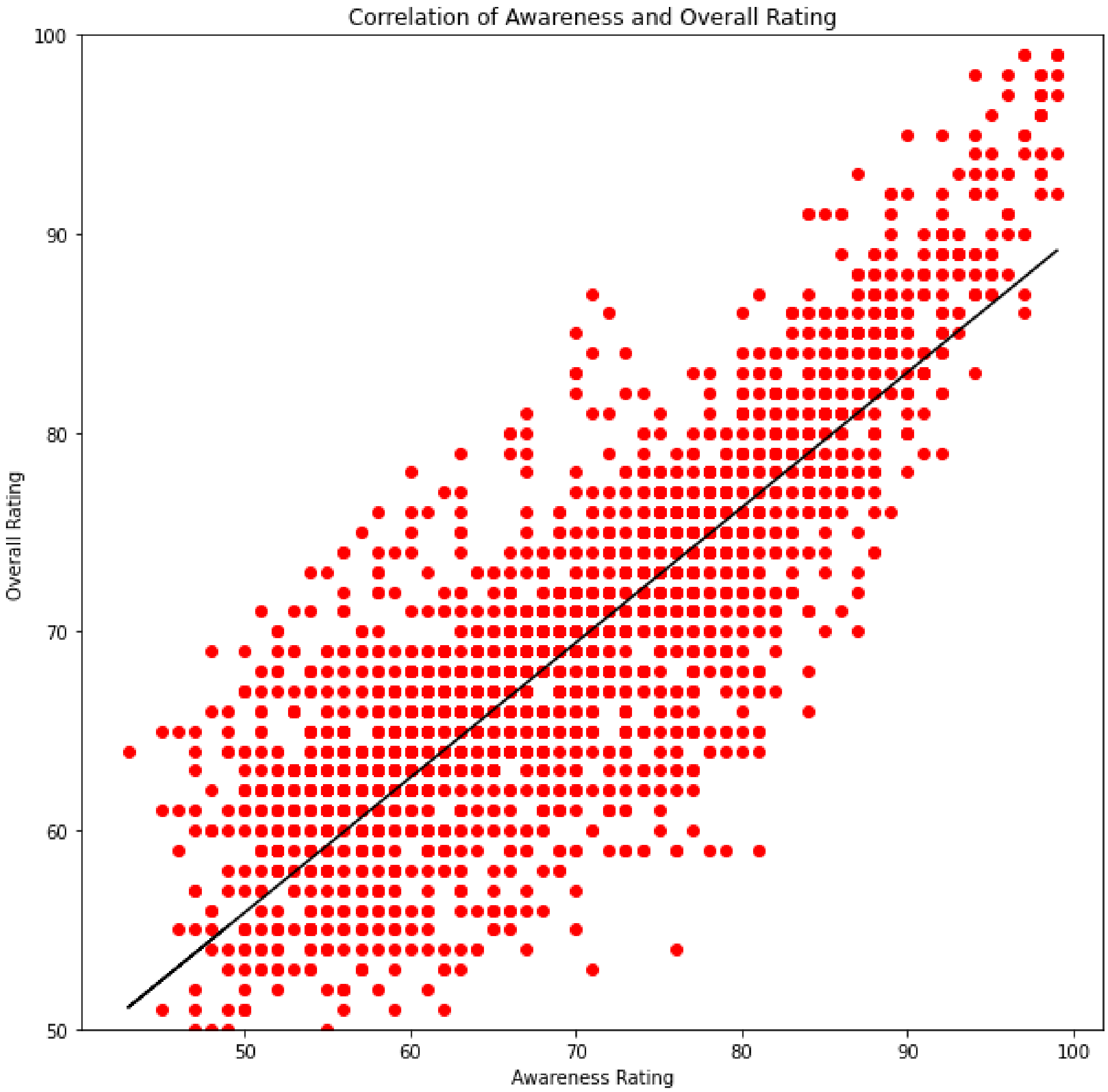
0.2 Plot Interpretation: This graph shows a very clear strong, positive relationshup between awareness and overall. One could interpret this graph as saying awareness predicts overall rating, however correlation does not imply causation. Based off of my starting hypothesis however this is an encouraging start.

[9]:

```
#Creating a scatterplot with a trend line showing the correlation between Awareness and Overall Rating but with long snappers removed
m, b = np.polyfit(df1["Awareness"], df["Overall Rating"], 1)
fig, ax = plt.subplots(figsize = (10,10))
ax.scatter(df1["Awareness"], df1["Overall Rating"], color = "red")
plt.title("Correlation of Awareness and Overall Rating")
plt.xlabel("Awareness Rating")
plt.ylabel("Overall Rating")
plt.ylim(50,100)
plt.plot(df1["Awareness"], m*df1["Awareness"]+b, color = "black")
```

[9]:

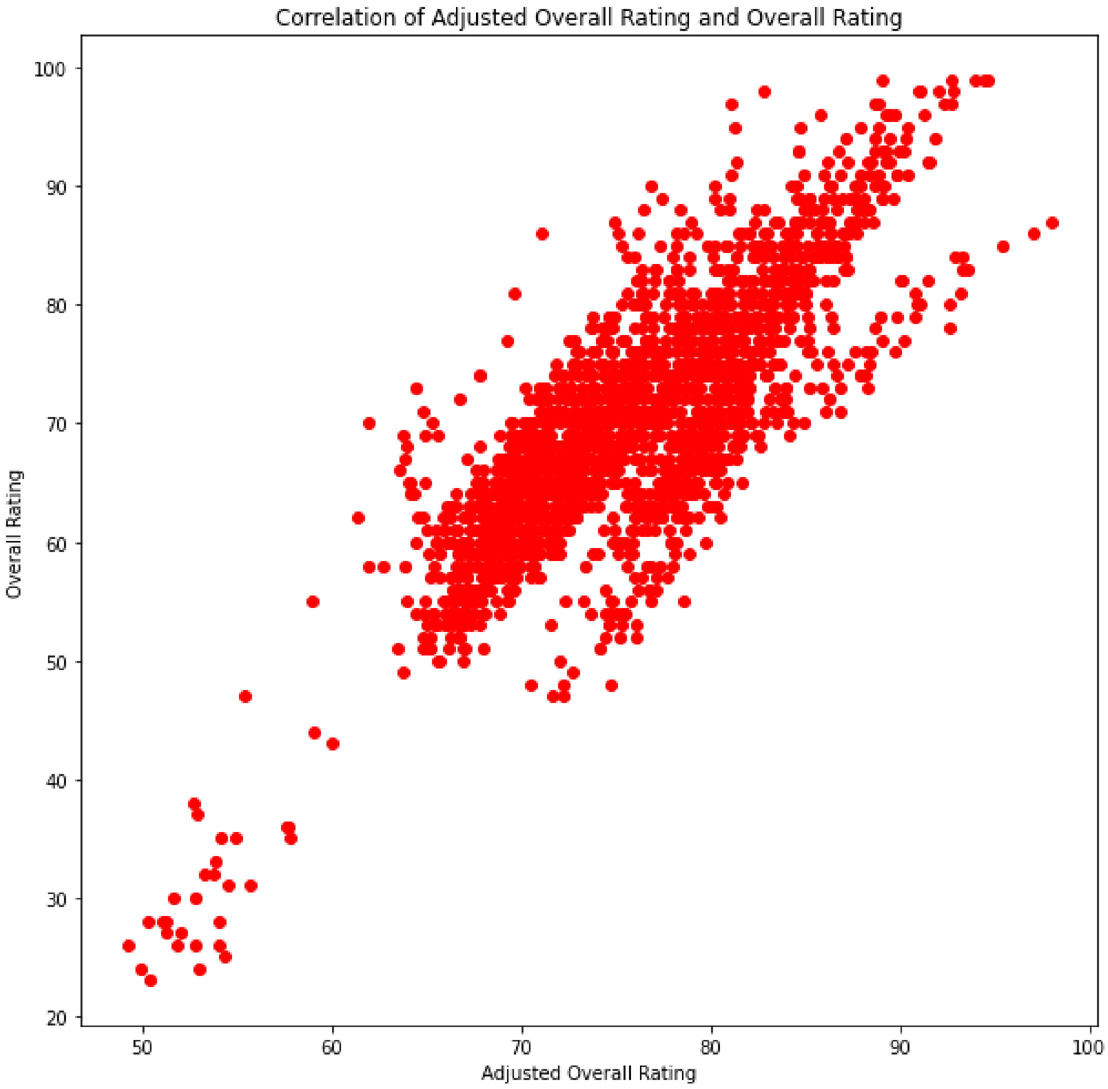
```
[<matplotlib.lines.Line2D at 0x7f1f5c955ae0>]
```



0.3 Plot Interpretation: This is the same graph as before ut with the ylims set to exclude anyone with an overall below 50. I did not like the way the graph looked with the few outliers at the bottom indicating the long snappers. Long snappers have a low overall in Madden because they are listed as tight ends but their skills do not relate to that position, thus the exceptionally low rating.

```
[11]: #Creating a scatterplot with showing the correlation between Adjusted Overall␣
      ↪Rating and Overall Rating
      fig, ax = plt.subplots(figsize = (10,10))
      ax.scatter(df1["Adj Overall"], df1["Overall Rating"], color = "red")
      plt.title("Correlation of Adjusted Overall Rating and Overall Rating")
      plt.xlabel("Adjusted Overall Rating")
      plt.ylabel("Overall Rating")
```

```
[11]: Text(0, 0.5, 'Overall Rating')
```



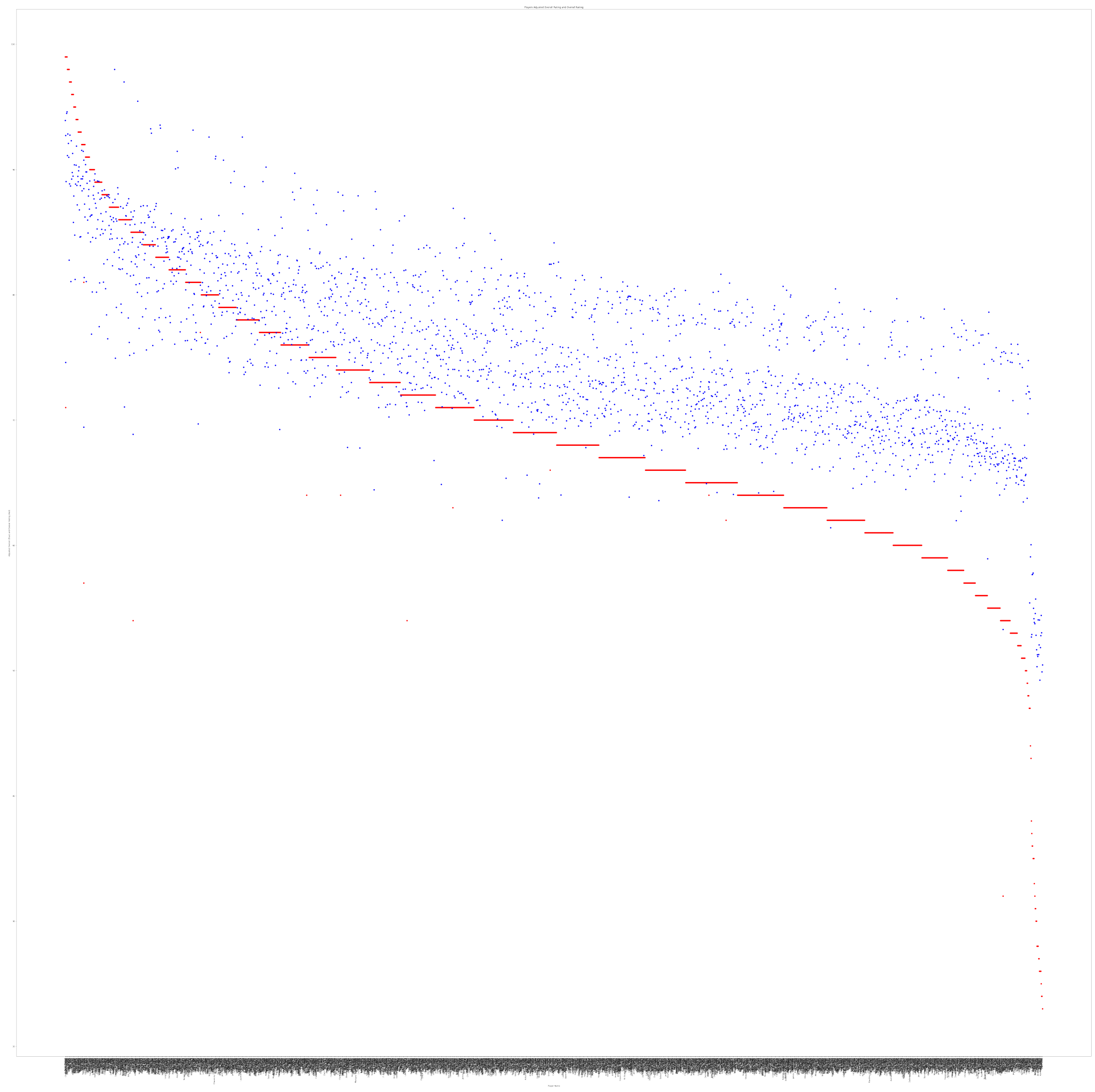
0.4 Plot Interpretation: This plot gives us a great chance to look at how many players are above a certain threshold of overall for both actual overall and adjusted overall. One thing that stands out to me is that the outliers, the long snappers, have a much high adjusted overall rating than actual overall. Something else I noticed is that their tends to be a less players with an adjusted overall above 80 but also a lot less with an adjusted overall below 70 in regards to actual overall. It could be that awareness is used to separate the average players from the great ones but also the much worse ones and spread overall around.

[13]:

```
#Creating a scatterplot of Adjusted Overall and Overall Rating by player
playerName = df1["Full Name"].astype("category")
fig, ax = plt.subplots(figsize = (100,100))
ax.scatter(playerName, df1["Overall Rating"], color = "red")
ax.scatter(playerName, df1["Adj Overall"], color = "blue")
plt.title("Players Adjusted Overall Rating and Overall Rating")
plt.xlabel("Player Name")
plt.xticks(rotation = 90)
plt.ylabel("Adjusted Overall (Blue) and Overall Rating (Red)")
```

[13]:

```
Text(0, 0.5, 'Adjusted Overall (Blue) and Overall Rating (Red)')
```

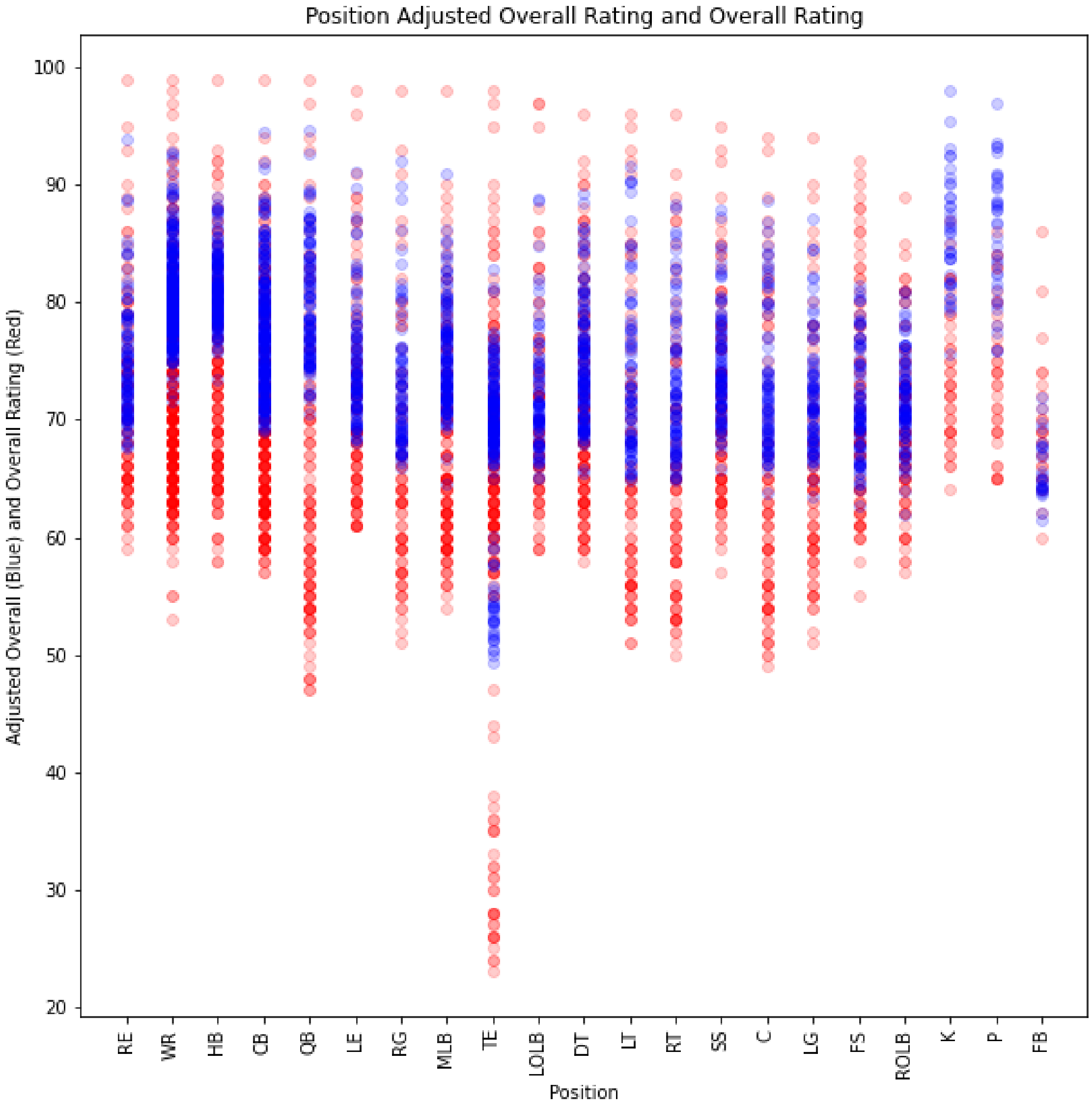


0.5 Plot Interpretation: This graph was intended to show all players overalls vs their adjusted overalls to show the difference at an individual level, however this did not given the big data aspect. As a whole you can how much smaller the spread is for the adjusted overalls, however it cannot not be identified which players are effected the most.

[15]:
```
#Creating a scatterplot of Adjusted Overall and Overall Rating
position = df1["Position"].astype("category")
fig, ax = plt.subplots(figsize = (10,10))
ax.scatter(position, df1["Overall Rating"], color = "red", alpha = .2)
ax.scatter(position, df1["Adj Overall"], color = "blue", alpha = .2)
```

```
plt.title("Position Adjusted Overall Rating and Overall Rating")
plt.xlabel("Position")
plt.xticks(rotation = 90)
plt.ylabel("Adjusted Overall (Blue) and Overall Rating (Red)")
```

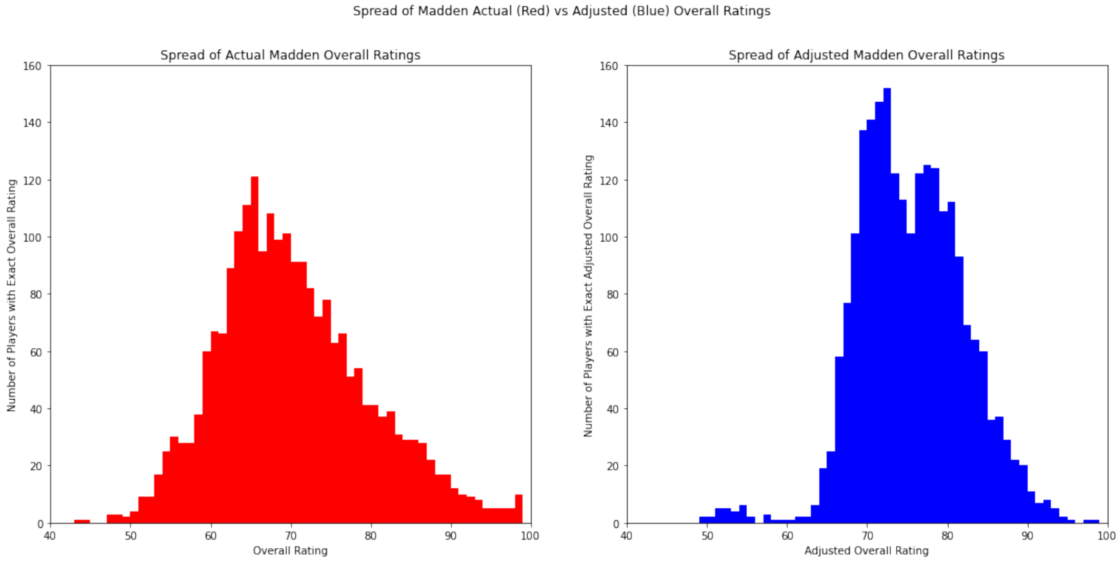
```
[15]: Text(0, 0.5, 'Adjusted Overall (Blue) and Overall Rating (Red)')
```



0.6 Plot Interpretation: I thought a better way to view the above graph that failed was to separate by position. This reiterated what I had already noticed, which was that there was a much greater spread in actual overall rating. This change was not seen in kickers and punters however, as their adjusted overall was higher across the board. One thing I noticed was that actual overalls tended to have many more low ratings than they did high ratings. You can see the blue indicating adjusted overall does not account for many low ratings as there is a lot of red below the blue groups.

```
[45]: #Creating a histogram of Overall Rating
      fig, ax = plt.subplots(nrows = 1, ncols = 2, figsize = (18,8))
      fig.suptitle("Spread of Madden Actual (Red) vs Adjusted (Blue) Overall Ratings")
      ax[0].hist(df1["Overall Rating"], bins = np.arange(20,100), color = "red")
      ax[1].hist(df1["Adj Overall"], bins = np.arange(20,100), color = "blue")
      ax[0].set_title("Spread of Actual Madden Overall Ratings")
      ax[0].set_xlabel("Overall Rating")
      ax[0].set_ylabel("Number of Players with Exact Overall Rating")
      ax[0].set_xlim(40,100)
      ax[0].set_ylim(0,160)
      ax[1].set_title("Spread of Adjusted Madden Overall Ratings")
      ax[1].set_xlabel("Adjusted Overall Rating")
      ax[1].set_ylabel("Number of Players with Exact Adjusted Overall Rating")
      ax[1].set_xlim(40,100)
      ax[1].set_ylim(0,160)
```

```
[45]: (0.0, 160.0)
```



0.7 Plot Interpretation: To reiterate and few the same point in a different way, you can clearly see the difference between how many players occupy the middle of Madden ranges from about 70 to 85. This graph also shows how the actual ratings have a lot more lower rated players as well as higher rated players.

[ ]: