

# Data Analysis Application

Male vs Female Pay Disparity within Professional Sports

Sydney Rudny  
12-15-2020

## Data Source:

Data was pulled from spotrac.com through a premium account membership. Revenue data from pulled from multiple different sources shown below:

WNBA - wsn.com

NBA – espn.com

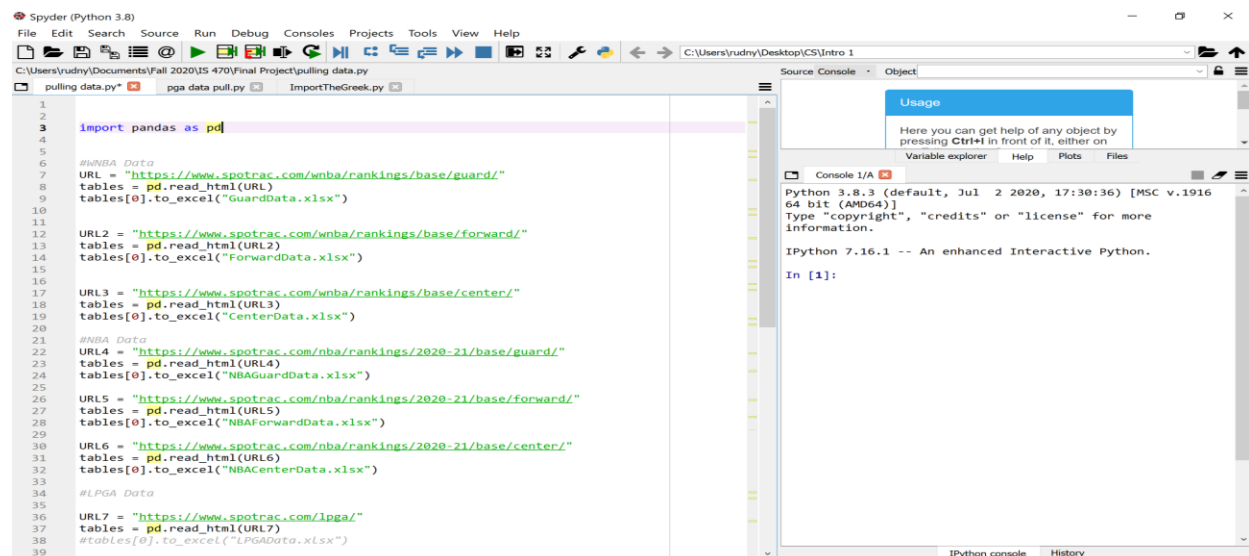
WTA - causeiq.com

ATP - sportspromedia.com

PGA-thengfq.com

## Data ETL

Multiple steps were taken to get the data in correct format to run analysis through RStudio. To grab data off the website Spotrac, a few lines of code in Python were able to pull the data from the web and put it into an Excel worksheet so the data could be cleaned up. Below is a code snippet used to pull the data from the web using the pandas package. NBA and WNBA data was split up by position because functions in both Python and R had issues pulling in the entire table of players when they were all together.



```
1
2
3 import pandas as pd
4
5
6
7 #WNBA Data
8 URL = "https://www.spotrac.com/wnba/rankings/base/guard/"
9 tables = pd.read_html(URL)
10 tables[0].to_excel("GuardData.xlsx")
11
12 URL2 = "https://www.spotrac.com/wnba/rankings/base/forward/"
13 tables = pd.read_html(URL2)
14 tables[0].to_excel("ForwardData.xlsx")
15
16 URL3 = "https://www.spotrac.com/wnba/rankings/base/center/"
17 tables = pd.read_html(URL3)
18 tables[0].to_excel("CenterData.xlsx")
19
20
21 #NBA Data
22 URL4 = "https://www.spotrac.com/nba/rankings/2020-21/base/guard/"
23 tables = pd.read_html(URL4)
24 tables[0].to_excel("NBAGuardData.xlsx")
25
26 URL5 = "https://www.spotrac.com/nba/rankings/2020-21/base/forward/"
27 tables = pd.read_html(URL5)
28 tables[0].to_excel("NBAForwardData.xlsx")
29
30 URL6 = "https://www.spotrac.com/nba/rankings/2020-21/base/center/"
31 tables = pd.read_html(URL6)
32 tables[0].to_excel("NBACenterData.xlsx")
33
34 #LPGA Data
35
36 URL7 = "https://www.spotrac.com/lpga/"
37 tables = pd.read_html(URL7)
38 tables[0].to_excel("LPGAData.xlsx")
39
```

The next step in cleansing the data required each sport to have common column names in order to combine them all together. After reviewing the data these were the best-chosen column options: Player, Position, Base (salary), Team, Sex, and League.

Note: Team/Position were only valuable to NBA and WNBA but kept for analysis.

### *Basketball:*

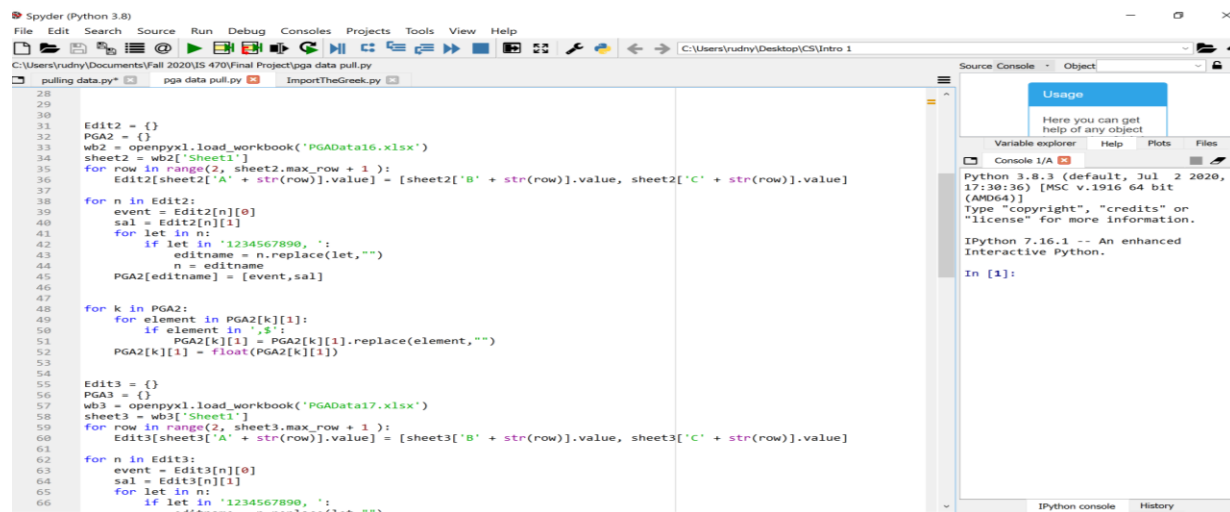
Both the NBA and WNBA had no issues when it came to converting the data into the chosen columns.

### Tennis:

For tennis, the WTA and ATP had a different set up for their data. They gave the players names, the number of championships they had won, and their total earnings. In order to get an accurate comparison, I chose the WTA and ATP Base to be their total earnings divided by the number of championships they won. Although this excludes player who have not won any championships and doesn't reflect events athletes were paid for but didn't win, it was the best way to equalize their earnings. For the team column, ATP and WTA have blanks.

### Golf:

For professional golf, the LPGA had the player name, the number of events they've played, and total earnings. For PGA however it was different. They had player name, number of events for one year, and earnings for one year. In order to get PGA and LPGA on equivalent standards (or as close to it as possible), the last 5 years of PGA data would be combined into total events and total earnings for the players. From there, Base could be found by taking the total earnings and dividing by total events. There were some issues with combining PGA data by name as each player name column had their first and last name and their age. Since their age would increase by one each year they wouldn't be recognized as the same name to combine on. Through use of Python, the age was deleted from the name title, then added to a dictionary as a key with its elements being their earning and number of events for that year. It read through each data set and if the key already existed it would add onto the players earnings and events for a total resulting over the last 5 years. The code is displayed below for putting the data from 2016 into the dictionary.



```
28
29
30
31 Edit2 = {}
32 PGA2 = {}
33 wb2 = openpyxl.load_workbook('PGAData16.xlsx')
34 sheet2 = wb2['Sheet1']
35 for row in range(2, sheet2.max_row + 1):
36     Edit2[sheet2['A' + str(row)].value] = [sheet2['B' + str(row)].value, sheet2['C' + str(row)].value]
37
38 for n in Edit2:
39     event = Edit2[n][0]
40     sal = Edit2[n][1]
41     for let in n:
42         if let in '1234567890 ':
43             editname = n.replace(let, "")
44             n = editname
45     PGA2[editname] = [event, sal]
46
47
48 for k in PGA2:
49     for element in PGA2[k][1]:
50         if element in ',.$':
51             PGA2[k][1] = PGA2[k][1].replace(element, "")
52     PGA2[k][1] = float(PGA2[k][1])
53
54
55 Edit3 = {}
56 PGA3 = {}
57 wb3 = openpyxl.load_workbook('PGAData17.xlsx')
58 sheet3 = wb3['Sheet1']
59 for row in range(2, sheet3.max_row + 1):
60     Edit3[sheet3['A' + str(row)].value] = [sheet3['B' + str(row)].value, sheet3['C' + str(row)].value]
61
62 for n in Edit3:
63     event = Edit3[n][0]
64     sal = Edit3[n][1]
65     for let in n:
66         if let in '1234567890 ':
67             editname = n.replace(let, "")
68             n = editname
69     PGA3[editname] = [event, sal]
```

For final results of data analysis see [Final Project Findings Report](#).