NBA_Project(Revised)

October 4, 2020

Intro: I performed the following analysis for my intermediate data programming with Python class at the University of Washington. The assignment called for us to to select a data set, identify research questions and then perform an analysis to answer said questions. This project attempted to assess the value of the NBA combine in terms of evaluating prospects by attempting to determine if any of the combine measurements was a strong predictor of in game performance. In order to compensate for differences in playing styles I primarily used a metric called composite metric called win shares, which attempts to assign "credit" for winning a certain number of games to a particular athlete. I also used PPG and noted other relationships between in game statistics and combine measurements that seemed significant.

For ease of use I've put most of my write-up and my python code into a Jupyter notebook for easier review and recreation of my results. I've also made a few tweaks or revisions to improve the project beyond what I originally turned in. However, the notebook is quite extensive so there is a PDF that provides a high-level summary of the results and methodology, along with a list of assumptions and caveats. Finally, I've added a section at the end where I discuss future related analyses I'd like to perform.

The research questions were as follows:

1. How good are the NBA's methods for evaluating talent overall? Meaning: is there a strong relationship between draft position and in game performance? Are "draft busts" where players significantly underperform relative to their draft position rare or common?
2. Is there a significant relationship between draft position and combine performance? Going to Kevin Durant's statements, does the combine hold any value for the athletes?
3. Are there any relationships, patterns or strong correlations between NBA combine performance and in game statistics like PPG and WS? E.g. do faster and more agile players score more points?

[1]:
```
# first we import all the packages we're going to use for this project
# it's worth noting that there are packages here that this version of the␣
↪project didn't use.

import pandas as pd
import math
import os
import glob2 as glob
import numpy as np
from sklearn import linear_model, metrics
import matplotlib.pyplot as plt
```

```python
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as seabornInstance
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn import metrics
import seaborn as sns
%matplotlib inline
```

[2]:
```python
# setting the option to view the entire data frame without columns being hidden

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

[3]:
```python
# I used the glob package to open up a directory containing all of the csv␣
↪files with the NBA season by season
# player statistics and then merge them all into one data frame, where each␣
↪subsequent csv is appended to the
# end of the data frame.
# "header=1" was used to not include an erroneous header row from converting␣
↪the HTML table to a CSV


nba_df = pd.concat([pd.read_csv(f, header=1) for f in glob.glob(os.path.
↪join('nba_stats', "*.csv"))], sort=False)
nba_df.shape
```

[3]:
```
(4847, 33)
```

[4]:
```python
# according to the NBA you have to play 58 games in a season for
# that season to be statistically significant as far as qualifying to be in the␣
↪statistical rankings.
# So we'll sort out only the seasons where an individual athlete played in at␣
↪least 58 games.


nba_min = nba_df[(nba_df['G'] >= 58)]
nba_min.shape
```

[4]:
```
(2529, 33)
```

For this round of the analysis I just focused on whether or not an individual athlete had managed to complete a single 58 game season rather than having a minimum number of seasons. I.e. evaluating the ability of a given player to produce at least one statistically significant (and hopefully above average) NBA season, rather than evaluating longevity.

```
[5]: # sort out just the statistics we want to use in our analysis, dropping␣
 ↪statistics due to being
# functions of other stats (E.g. FG%) or just to simplify our analysis.


nba_subset = nba_min[['Player', 'Season', 'WS', 'G', 'MP', 'ORB', 'DRB',
                      'TRB','AST', 'STL', 'BLK', 'PTS', 'TOV']]
```

```
[6]: # calculate points per game, since it's not in the original data set

nba_subset.loc[:,'PPG'] = (nba_subset.loc[:,'PTS'] / nba_subset.loc[:,'G'])
```

```
/Users/markhamlee/opt/anaconda3/envs/cse163/lib/python3.7/site-
packages/pandas/core/indexing.py:376: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: http://pandas.pydata.org/pandas-
docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  self.obj[key] = _infer_fill_value(value)
/Users/markhamlee/opt/anaconda3/envs/cse163/lib/python3.7/site-
packages/pandas/core/indexing.py:494: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: http://pandas.pydata.org/pandas-
docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  self.obj[item] = s
```

Now that we have the data for all of the NBA players, let's sort it by win shares and see who the leaders were over this seven year time period.

```
[19]: nba_subset = nba_subset.sort_values('WS', ascending=False)
nba_subset.head(30)
```

```
[19]:           Player   Season    WS   G    MP  ORB  DRB  TRB  AST  STL  \
0         LeBron James  2012-13  19.3  76  2877   97  513  610  551  129
1         Kevin Durant  2013-14  19.2  81  3122   58  540  598  445  103
2         Kevin Durant  2012-13  18.9  81  3119   46  594  640  374  116
3         LeBron James  2009-10  18.5  76  2966   71  483  554  651  125
4        Stephen Curry  2015-16  17.9  79  2700   68  362  430  527  169
5         James Harden  2014-15  16.4  81  2981   75  384  459  565  154
7           Chris Paul  2014-15  16.1  82  2857   52  324  376  838  156
6         Kevin Durant  2009-10  16.1  82  3239  105  518  623  231  112
8         LeBron James  2013-14  15.9  77  2902   81  452  533  488  121
9        Stephen Curry  2014-15  15.7  80  2613   56  285  341  619  163
10        LeBron James  2010-11  15.6  79  3063   80  510  590  554  124
11        James Harden  2017-18  15.4  72  2551   41  348  389  630  126
```

12	James Harden	2018-19	15.2	78	2867	66	452	518	586	158
13	James Harden	2016-17	15.0	81	2947	95	564	659	907	121
14	Pau Gasol	2010-11	14.7	82	3037	268	568	836	273	48
15	Kevin Durant	2015-16	14.5	72	2578	45	544	589	361	69
16	LeBron James	2011-12	14.5	62	2326	94	398	492	387	115
17	Giannis Antetokounmpo	2018-19	14.4	72	2358	159	739	898	424	92
19	Dwight Howard	2010-11	14.4	78	2935	309	789	1098	107	107
18	Rudy Gobert	2018-19	14.4	81	2577	309	732	1041	161	66
20	Rudy Gobert	2016-17	14.3	81	2744	314	721	1035	97	49
21	Kevin Love	2013-14	14.3	77	2797	224	739	963	341	59
22	Anthony Davis	2014-15	14.0	68	2455	173	523	696	149	100
23	LeBron James	2017-18	14.0	82	3026	97	612	709	747	116
24	Karl-Anthony Towns	2017-18	14.0	82	2918	238	774	1012	199	64
25	Russell Westbrook	2015-16	14.0	80	2750	145	481	626	834	163
27	Chris Paul	2010-11	13.9	80	2880	38	289	327	782	188
26	Chris Paul	2012-13	13.9	70	2335	53	209	262	678	169
28	Jimmy Butler	2016-17	13.8	76	2809	129	341	470	417	143
29	Anthony Davis	2017-18	13.7	75	2727	187	645	832	174	115

	BLK	PTS	TOV	PPG
0	67	2036	226	26.789474
1	59	2593	285	32.012346
2	105	2280	280	28.148148
3	77	2258	261	29.710526
4	15	2375	262	30.063291
5	60	2217	321	27.370370
7	15	1564	190	19.073171
6	84	2472	271	30.146341
8	26	2089	270	27.129870
9	16	1900	249	23.750000
10	50	2111	284	26.721519
11	50	2191	315	30.430556
12	58	2818	387	36.128205
13	38	2356	464	29.086420
14	130	1541	142	18.792683
15	85	2029	250	28.180556
16	50	1683	213	27.145161
17	110	1994	268	27.694444
19	186	1784	279	22.871795
18	187	1284	130	15.851852
20	214	1137	147	14.037037
21	35	2010	196	26.103896
22	200	1656	95	24.352941
23	71	2251	347	27.451220
24	115	1743	159	21.256098
25	20	1878	342	23.475000
27	5	1268	177	15.850000

```
26   10  1186  159  16.942857
28   32  1816  159  23.894737
29  193  2110  162  28.133333
```

[20]:
```python
# count uniques in the player column just to see
# the number of athletes represented in the dataset

nba_subset['Player'].nunique()
```

[20]:
```
692
```

The NBA player data set contains data on 546 total players across 1,784 "seasons" from the perspective of seasons represented for the players as whole, this also means that on average there are a little over three seasons per player. Looking at the data on the top 30 players, we can also see that only 12 players are represented in the top 30 seasons by win share. This suggests that in a typical draft very few if not none of the players will join the league's top echelons. Given this, we will evaluate drafting based on the probability that a player selected in the top 15 of draft is at minimum above average and ideally in the top 25% of NBA players statistically.

Now that we have the NBA dataset we'll use for our analysis, let's walk through some high-level stats for the dataset as a whole

[21]:
```python
nba_subset.describe()
```

[21]:
```
                WS            G           MP          ORB          DRB  \
count  2529.000000  2529.000000  2529.000000  2529.000000  2529.000000
mean      4.193199    72.222222  1850.041914    81.546066   248.976671
std       2.938548     7.396039   586.837982    65.618253   135.497959
min      -2.100000    58.000000   267.000000     5.000000    17.000000
25%       2.100000    66.000000  1402.000000    33.000000   153.000000
50%       3.600000    73.000000  1859.000000    59.000000   214.000000
75%       5.700000    79.000000  2293.000000   113.000000   316.000000
max      19.300000    83.000000  3239.000000   440.000000   848.000000

               TRB          AST          STL         BLK          PTS  \
count  2529.000000  2529.000000  2529.000000  2529.00000  2529.000000
mean    330.522341   172.988533    58.776592    37.51720   802.836299
std     189.924047   142.893163    31.092214    35.49262   423.972868
min      22.000000     8.000000     6.000000     0.00000    62.000000
25%     192.000000    74.000000    36.000000    14.00000   484.000000
50%     284.000000   128.000000    53.000000    26.00000   724.000000
75%     419.000000   226.000000    76.000000    49.00000  1039.000000
max    1247.000000   907.000000   191.000000   269.00000  2818.000000

               TOV          PPG
count  2529.000000  2529.000000
mean    104.901542    11.007937
std      58.351803     5.529088
```

```
min      11.000000     0.968750
25%      62.000000     6.870968
50%      93.000000     9.920635
75%     134.000000    14.082192
max     464.000000    36.128205
```

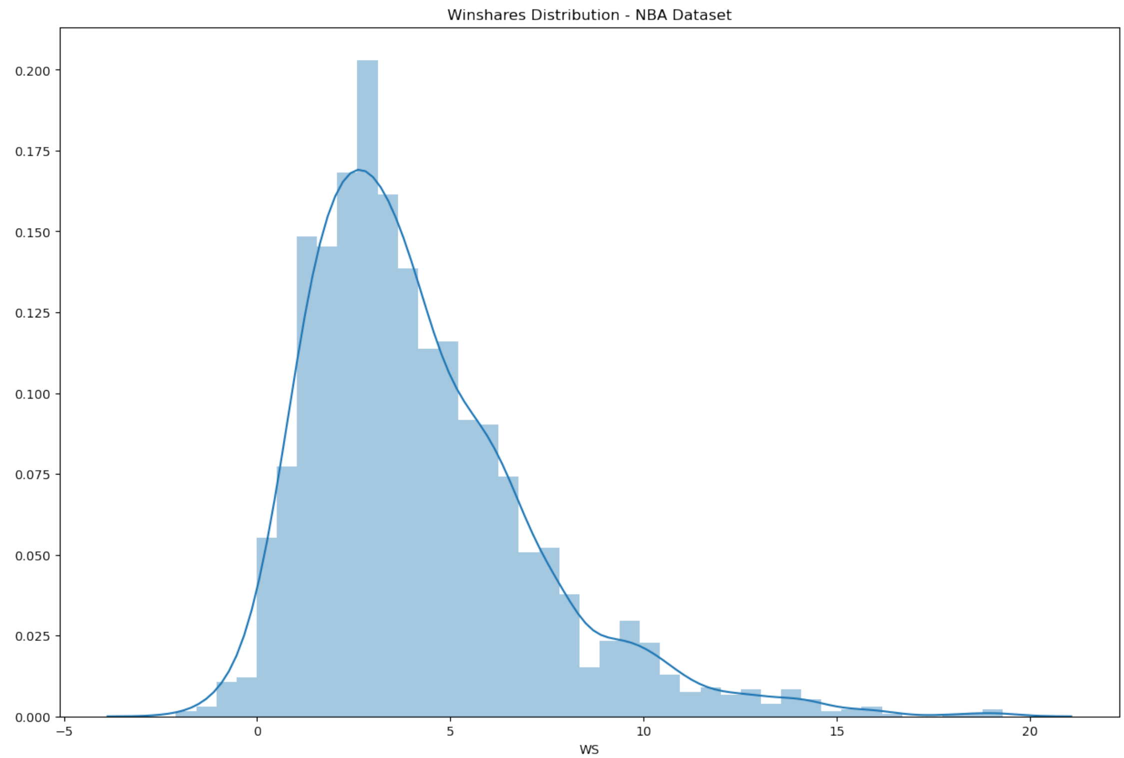
Some key pieces of information about our dataset includes:

- Average Win Share is 4.19 games with a standard deviation of 2.93 games
- Average PPG is 11.008 with a standard deviation of 5.53
- The 75% percentile is interesting as well, 5.7 for Win Share and 14.082 for PPG

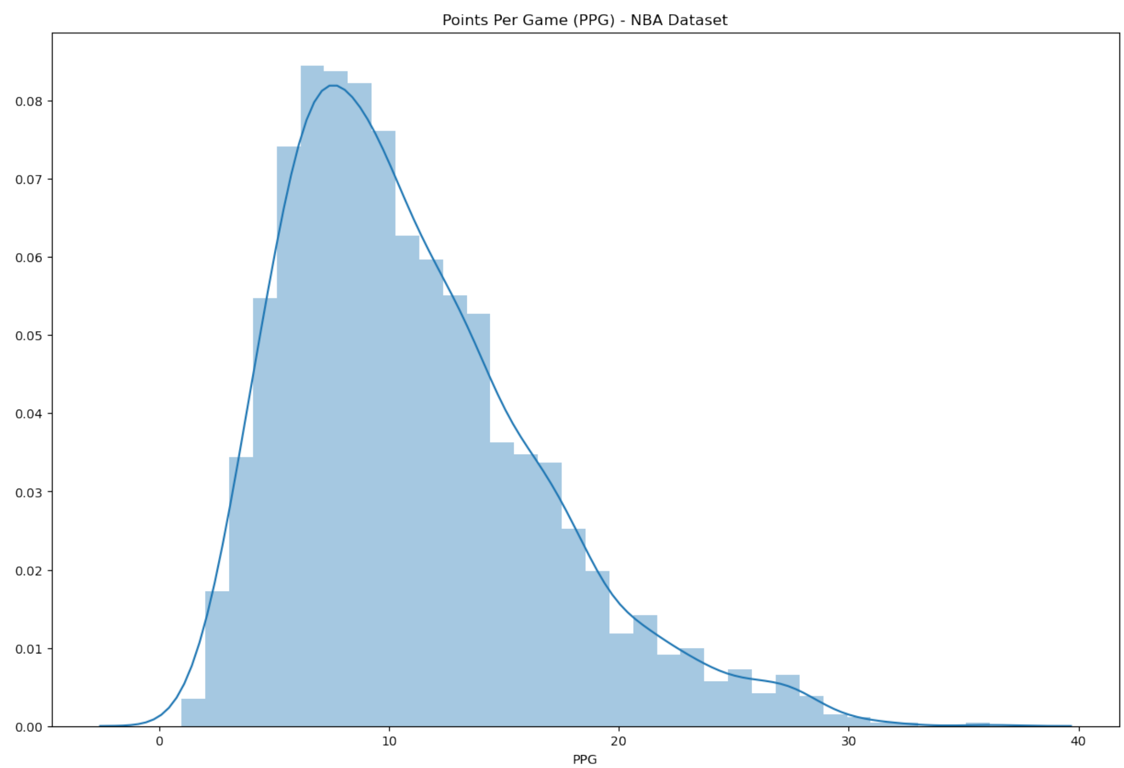
We'll use the average win share and PPG metrics to see how the players in our combine data sets compare to the broader NBA dataset as a group, since there will be players in the NBA player data set that are not in the combine data set. We'll also use the data to evaluate things ranging from draft effectiveness to what % of NBA first round picks have produced elite seasons in terms of seasons in the top 25% of the league statistically.

[22]:
```
# let's look at the distribution of PPG and win share

plt.figure(figsize=(15,10))
plt.tight_layout()
seabornInstance.distplot(nba_subset['WS'])
plt.title('Winshares Distribution - NBA Dataset')

plt.figure(figsize=(15,10))
plt.tight_layout()
seabornInstance.distplot(nba_subset['PPG'])
plt.title('Points Per Game (PPG) - NBA Dataset')
```

[22]:
```
Text(0.5, 1.0, 'Points Per Game (PPG) - NBA Dataset')
```



Winshares Distribution - NBA Dataset
0.200
0.175
0.150
0.125
0.100
0.075
0.050
0.025
0.000
−5
0
5
10
15
20
WS

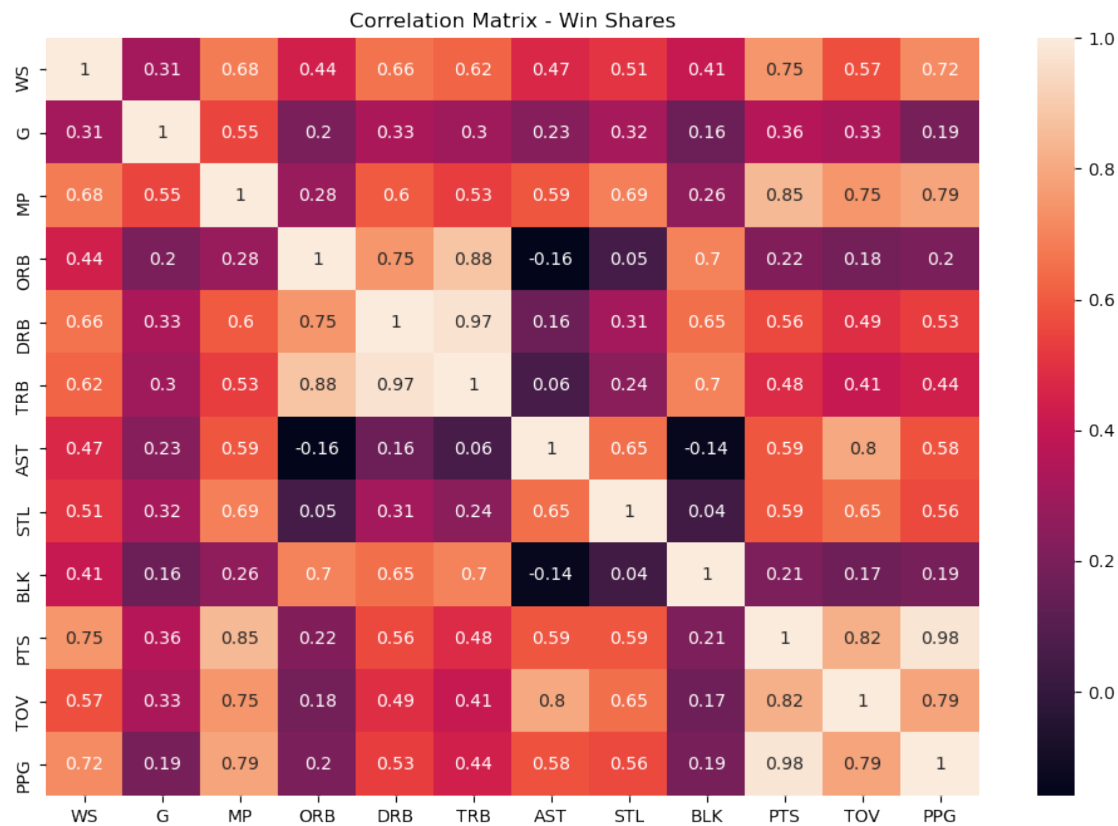


Points Per Game (PPG) - NBA Dataset
0.08
0.07
0.06
0.05
0.04
0.03
0.02
0.01
0.00
0
10
20
30
40
PPG

PPG is closer to being a normal distribution, but overall both distributions are skewed to the right. This more or less means that most players are grouped near the average, and the exceptional ones are rare/outliers, which suggests that our analysis is probably going to be better at predicting "good" rather than great players, as the great ones are so rare.

[23]:
```python
# Let's quickly evaluate our primary metric, how does WS correlate to other statistics in our data set?

# this code sets the size of our plots so they're easier to read
plt.rcParams['figure.figsize'] = [12, 8]
plt.rcParams['figure.dpi'] = 100

#calculates correlations and then uses the heatmap function of seaborn to plot it
corr_ws = nba_subset.corr().round(2)
sns.heatmap(data=corr_ws, annot=True)
plt.title('Correlation Matrix - Win Shares')
```

[23]:
```
Text(0.5, 1.0, 'Correlation Matrix - Win Shares')
```



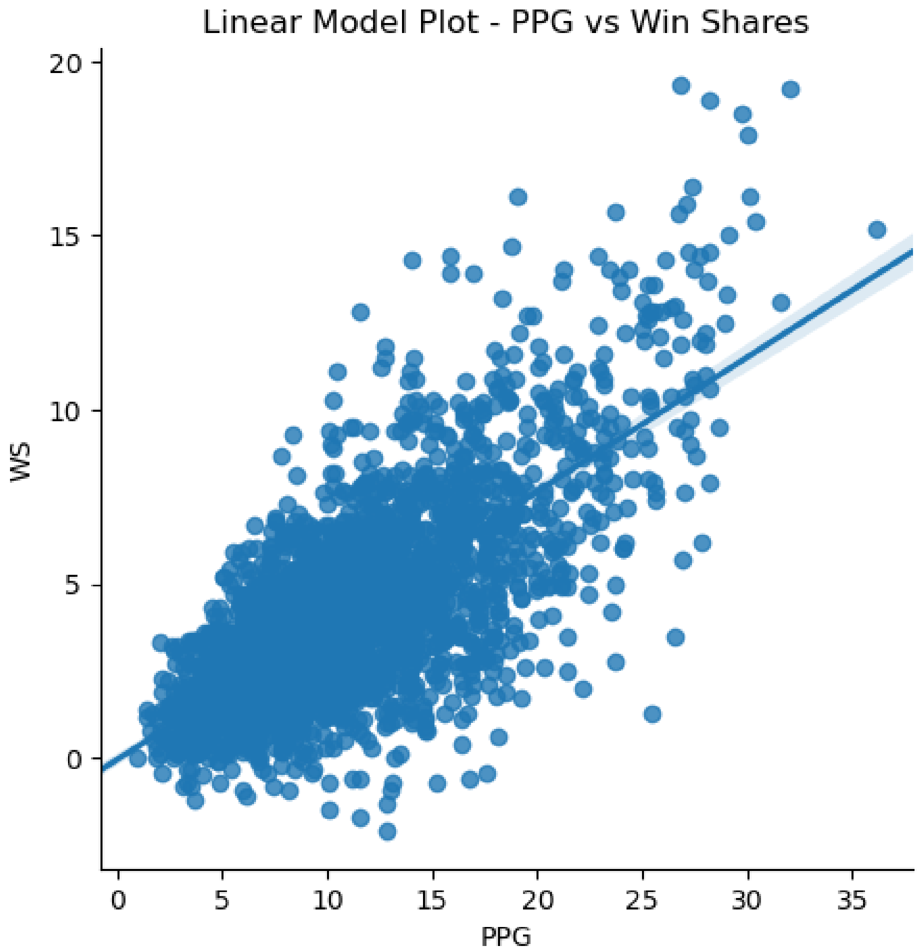
Looking through the data we see a variety of patterns, and they all fairly logical and not particularly unexpected:

- Turnover has a high correlation with PPG, Points, Assists and Minutes Played, which suggests that the players who touch the ball the most due to their scoring, assists and the like are just more often in a position to make a mistake. It's also worth noting that the turnover number is the total number of turnovers, it's not expressed in terms of turnovers per touches.
- Steals has a good correlation with assists and turnovers, which is interesting as it suggests that the players who turn over the ball the most are often the ones causing their opponents to do the same. It's also possible that in the process of stealing the ball they sometimes lose control and get both stats at the same time
- TRB and BLKs have a strong correlation, which is a case of mathematics proving what makes intuitive sense, namely: players that are good at getting rebounds also tend to be good shot blockers. Again, this makes sense: if you have the height, leaping ability and good awareness of where the ball is to get a rebound, you can probably also apply those skills to blocking shots.
- Win shares has its highest correlations with PPG (0.72), Defensive Rebounds (0.66) and Minutes Played (0.68), none of this is surprising, as it means that players who score a lot, take scoring opportunities away from opponents and are durable enough to play a lot of minutes are going to help their teams more. What's interesting are the items that weren't in the strong correlation or at least close category, namely: offensive rebounds (0.41) and assists (0.48), which probably means that the giving your team extra scoring chances or passing the ball to others that score isn't as valuable to your team as just scoring yourself.

For purposes of a visual let's make a linear regression plot of PPG vs. Win Share, just so we have a reference image of what a strong correlation looks like.

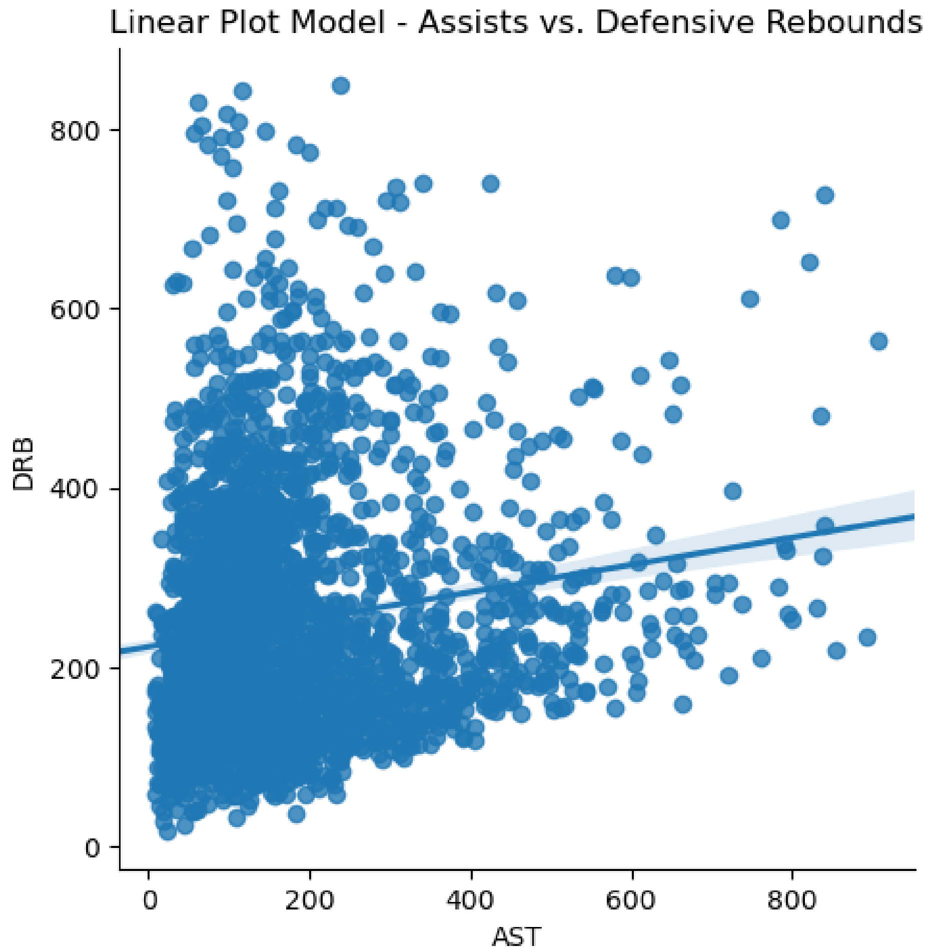
[24]:
```
sns.lmplot(x="PPG", y="WS", data=nba_subset)
plt.title('Linear Model Plot - PPG vs Win Shares')
```

[24]:
```
Text(0.5, 1.0, 'Linear Model Plot - PPG vs Win Shares')
```



[25]:
```
# Also for reference let's plot one of the weak correlations, assists vs.␣
↪defensive rebounds

sns.lmplot(x="AST", y="DRB", data=nba_subset)
plt.title('Linear Plot Model - Assists vs. Defensive Rebounds')
```

[25]:
```
Text(0.5, 1.0, 'Linear Plot Model - Assists vs. Defensive Rebounds')
```



As we can see from the above, we have a very general trend line with a fairly weak slope and the individual data points scattered all over the graph. If the NBA Combine does in fact relate to in game performance, our linear model plots will look more like the first graph than they will the second. Before we merge the player stats data with the combine data, let's aggregate the data by player and then sort in descending order by Win Share.

[26]:
```
nba_players = nba_subset.groupby('Player').mean()
nba_players = nba_players.sort_values('WS', ascending=False)

nba_players.head(30)
```

[26]:
```
                         WS          G           MP        ORB  \
Player
LeBron James      14.966667  74.555556  2795.111111  86.555556
Kevin Durant      14.088889  74.222222  2748.777778  50.444444
James Harden      12.180000  76.500000  2610.600000  58.200000
Chris Paul        12.100000  67.222222  2274.333333  41.666667
```

Stephen Curry	11.475000	77.125000	2687.000000	54.375000
Karl-Anthony Towns	11.350000	80.750000	2780.000000	256.500000
Rudy Gobert	11.100000	76.250000	2352.750000	274.000000
Anthony Davis	10.400000	68.333333	2376.333333	172.333333
Damian Lillard	10.042857	78.428571	2843.857143	49.571429
Russell Westbrook	9.955556	77.000000	2694.444444	126.444444
Nikola Joki_	9.725000	77.000000	2179.500000	204.000000
Jimmy Butler	9.285714	68.714286	2410.000000	106.428571
Dwight Howard	9.271429	76.142857	2548.285714	266.285714
Kawhi Leonard	9.257143	65.428571	2027.714286	83.000000
Chris Bosh	9.225000	75.000000	2576.500000	143.750000
Brandon Roy	9.100000	65.000000	2419.000000	73.000000
LaMarcus Aldridge	8.966667	75.000000	2636.111111	203.666667
Blake Griffin	8.925000	71.125000	2498.625000	159.625000
Giannis Antetokounmpo	8.866667	77.500000	2536.666667	124.666667
Chauncey Billups	8.750000	72.500000	2400.000000	25.000000
Hassan Whiteside	8.700000	74.000000	2104.000000	262.666667
Ben Simmons	8.700000	80.000000	2716.000000	158.500000
Al Horford	8.687500	74.750000	2462.125000	151.250000
Dirk Nowitzki	8.671429	75.000000	2399.428571	47.714286
Kevin Love	8.657143	68.714286	2230.571429	188.714286
Tyson Chandler	8.460000	68.600000	2037.600000	231.200000
Pau Gasol	8.362500	70.375000	2270.625000	178.625000
Kyle Lowry	8.244444	71.111111	2369.777778	65.333333
DeAndre Jordan	8.220000	76.600000	2232.100000	258.800000
Kyrie Irving	8.216667	67.333333	2324.000000	49.500000

	DRB	TRB	AST	STL \
Player				
LeBron James	481.000000	567.555556	561.000000	115.000000
Kevin Durant	503.333333	553.777778	331.000000	83.333333
James Harden	341.800000	400.000000	474.300000	118.900000
Chris Paul	251.444444	293.111111	637.222222	144.444444
Stephen Curry	293.500000	347.875000	517.500000	135.125000
Karl-Anthony Towns	701.250000	957.750000	209.750000	61.250000
Rudy Gobert	605.750000	879.750000	114.500000	56.000000
Anthony Davis	533.333333	705.666667	127.333333	91.833333
Damian Lillard	277.428571	327.000000	497.000000	76.142857
Russell Westbrook	440.000000	566.444444	682.555556	138.222222
Nikola Joki_	532.500000	736.500000	396.500000	84.500000
Jimmy Butler	253.714286	360.142857	255.857143	115.857143
Dwight Howard	702.000000	968.285714	113.857143	72.428571
Kawhi Leonard	332.428571	415.428571	157.428571	115.857143
Chris Bosh	462.000000	605.750000	130.000000	62.000000
Brandon Roy	212.000000	285.000000	305.000000	61.000000
LaMarcus Aldridge	466.444444	670.111111	157.666667	54.666667
Blake Griffin	480.750000	640.375000	315.000000	64.875000

Giannis Antetokounmpo	516.000000	640.666667	320.166667	93.166667
Chauncey Billups	184.000000	209.000000	398.000000	76.500000
Hassan Whiteside	660.666667	923.333333	47.666667	49.000000
Ben Simmons	519.500000	678.000000	635.500000	126.000000
Al Horford	457.125000	608.375000	270.125000	60.750000
Dirk Nowitzki	442.714286	490.428571	165.000000	51.000000
Kevin Love	588.142857	776.857143	176.285714	50.285714
Tyson Chandler	458.200000	689.400000	59.600000	42.400000
Pau Gasol	528.875000	707.500000	227.625000	33.000000
Kyle Lowry	264.777778	330.111111	479.000000	100.888889
DeAndre Jordan	606.500000	865.300000	70.200000	47.900000
Kyrie Irving	201.166667	250.666667	393.333333	93.666667

	BLK	PTS	TOV	PPG
Player				
LeBron James	53.666667	2005.000000	269.444444	26.875245
Kevin Durant	87.555556	2084.333333	235.777778	27.960399
James Harden	38.300000	1862.700000	277.000000	24.220056
Chris Paul	10.111111	1214.333333	157.333333	18.077013
Stephen Curry	17.250000	1823.250000	243.125000	23.666221
Karl-Anthony Towns	120.250000	1796.750000	198.500000	22.283774
Rudy Gobert	181.250000	916.000000	125.500000	11.846473
Anthony Davis	164.333333	1601.166667	126.166667	23.184079
Damian Lillard	24.428571	1844.142857	216.428571	23.635619
Russell Westbrook	25.222222	1844.555556	319.444444	24.033432
Nikola Joki_	55.250000	1251.500000	183.250000	16.298174
Jimmy Butler	34.428571	1231.571429	107.285714	18.191087
Dwight Howard	152.000000	1315.000000	221.571429	17.205421
Kawhi Leonard	43.428571	1157.714286	94.714286	17.481569
Chris Bosh	74.000000	1407.250000	140.000000	18.877648
Brandon Roy	16.000000	1398.000000	129.000000	21.507692
LaMarcus Aldridge	81.666667	1558.000000	121.888889	20.790056
Blake Griffin	37.125000	1556.375000	186.750000	21.851485
Giannis Antetokounmpo	104.333333	1457.500000	204.666667	18.976168
Chauncey Billups	11.500000	1317.500000	177.000000	18.162861
Hassan Whiteside	188.666667	1078.666667	129.333333	14.522007
Ben Simmons	65.500000	1308.000000	276.000000	16.357087
Al Horford	89.750000	1103.000000	118.000000	14.726329
Dirk Nowitzki	47.285714	1488.142857	105.285714	19.861248
Kevin Love	30.142857	1281.571429	134.857143	18.485603
Tyson Chandler	75.800000	675.800000	95.000000	9.850642
Pau Gasol	108.125000	1137.625000	134.875000	16.160514
Kyle Lowry	20.777778	1139.333333	173.444444	16.005407
DeAndre Jordan	129.100000	760.800000	109.400000	9.849896
Kyrie Irving	23.166667	1551.500000	176.500000	23.076274

Looking at the data we see that LeBron James and Kevin Durant are the top two in terms of WS, while Kevin scores more points it appears that LeBron has a higher WS due to contributing more in other ways in terms of assists, rebounds and steals.

[27]:
```python
# validate how many athletes are included

total_athletes = len(nba_players)
total_athletes
```

[27]:
```
692
```

[28]:
```python
# Use pandas import combine data


all_combine_data = pd.read_csv('new_data/nba_draft_combine_all_years.csv')
all_combine_data
```

[28]:
```
     Unnamed: 0                 Player  Year  Draft pick  Height (No Shoes)  \
0             0           Blake Griffin  2009         1.0              80.50
1             1       Terrence Williams  2009        11.0              77.00
2             2        Gerald Henderson  2009        12.0              76.00
3             3        Tyler Hansbrough  2009        13.0              80.25
4             4              Earl Clark  2009        14.0              80.50
..          …                     …   …         …                …
512         512               Peter Jok  2017         NaN              76.25
513         513            Rawle Alkins  2017         NaN              74.50
514         514   Sviatoslav Mykhailiuk  2017         NaN              78.50
515         515            Thomas Welsh  2017         NaN              83.50
516         516             V.J. Beachem  2017         NaN              78.25

     Height (With Shoes)  Wingspan  Standing reach  Vertical (Max)  \
0                  82.00     83.25           105.0            35.5
1                  78.25     81.00           103.5            37.0
2                  77.00     82.25           102.5            35.0
3                  81.50     83.50           106.0            34.0
4                  82.25     86.50           109.5            33.0
..                   …         …              …             …
512                77.75     80.00           102.0            31.0
513                75.75     80.75            99.0            40.5
514                79.50     77.00           100.0            33.0
515                84.50     84.00           109.5             NaN
516                80.00     82.25           104.5            37.0

     Vertical (Max Reach)  Vertical (No Step)  Vertical (No Step Reach)  \
0                   140.5                32.0                      137.0
1                   140.5                30.5                      134.0
2                   137.5                31.5                      134.0
3                   140.0                27.5                      133.5
```

```
4              142.5           28.5                   138.0
..               …              …                   …
512            133.0           26.5                   128.5
513            139.5           31.5                   130.5
514            133.0           27.0                   127.0
515              NaN            NaN                     NaN
516            141.5           30.0                   134.5

     Weight  Body Fat  Hand (Length)  Hand (Width)  Bench  Agility  Sprint
0     248.0       8.2            NaN           NaN   22.0    10.95    3.28
1     213.0       5.1            NaN           NaN    9.0    11.15    3.18
2     215.0       4.4            NaN           NaN    8.0    11.17    3.14
3     234.0       8.5            NaN           NaN   18.0    11.12    3.27
4     228.0       5.2            NaN           NaN    5.0    11.17    3.35
..      …       …            …           …    …     …     …
512   202.0      11.0           8.25          9.50    NaN    11.34    3.41
513   223.0      11.0           8.75         10.00    NaN    11.99    3.30
514   220.0      11.4           8.00          9.25    NaN    12.40    3.53
515   254.0      10.9           9.00         10.50    NaN      NaN     NaN
516   193.0       6.8           8.50          9.00    NaN    11.18    3.26

[517 rows x 19 columns]
```

Merge combine data with the NBA player statistics even though we don't have a full set of combine data for all the athletes, because since we do have draft position for all the players we can at least evaluate WS and PPG vs. draft position.

[29]:
```python
combine_nba_merge = pd.merge(nba_players, all_combine_data, on='Player')
combine_nba_merge.head(30)
```

[29]:
```
              Player         WS          G           MP         ORB  \
0       James Harden  12.180000  76.500000  2610.600000   58.200000
1      Stephen Curry  11.475000  77.125000  2687.000000   54.375000
2        Rudy Gobert  11.100000  76.250000  2352.750000  274.000000
3      Anthony Davis  10.400000  68.333333  2376.333333  172.333333
4     Damian Lillard  10.042857  78.428571  2843.857143   49.571429
5       Jimmy Butler   9.285714  68.714286  2410.000000  106.428571
6      Kawhi Leonard   9.257143  65.428571  2027.714286   83.000000
7      Blake Griffin   8.925000  71.125000  2498.625000  159.625000
8   Hassan Whiteside   8.700000  74.000000  2104.000000  262.666667
9        Paul George   8.162500  74.750000  2541.625000   69.875000
10    Andre Drummond   8.071429  77.428571  2387.285714  372.571429
11     Isaiah Thomas   7.550000  73.500000  2202.333333   41.000000
12    Terrence Jones   7.300000  76.000000  2078.000000  162.000000
13     Pascal Siakam   7.000000  80.500000  2113.500000  101.500000
14       Greg Monroe   6.471429  76.857143  2279.285714  227.571429
15      Steven Adams   6.466667  77.833333  2087.833333  269.333333
```

16	Otto Porter	6.450000	76.500000	2186.250000	97.000000
17	Monte Morris	6.200000	82.000000	1970.000000	35.000000
18	Draymond Green	6.185714	76.142857	2139.857143	89.285714
19	Bradley Beal	6.140000	75.400000	2665.200000	62.800000
20	Montrezl Harrell	6.100000	72.000000	1505.000000	122.666667
21	Kemba Walker	6.087500	75.625000	2575.875000	43.625000
22	Tobias Harris	6.016667	74.833333	2469.000000	74.500000
23	Gordon Hayward	6.000000	73.500000	2253.375000	53.500000
24	John Wall	5.983333	75.166667	2738.166667	43.000000
25	Kenneth Faried	5.980000	72.600000	1900.400000	231.600000
26	Jarrett Allen	5.900000	76.000000	1768.500000	167.500000
27	Klay Thompson	5.850000	76.875000	2542.500000	34.125000
28	Derrick Favors	5.837500	72.750000	1885.125000	185.250000
29	Tristan Thompson	5.800000	77.666667	2230.166667	265.000000

	DRB	TRB	AST	STL	BLK	PTS	\
0	341.800000	400.000000	474.300000	118.900000	38.300000	1862.700000	
1	293.500000	347.875000	517.500000	135.125000	17.250000	1823.250000	
2	605.750000	879.750000	114.500000	56.000000	181.250000	916.000000	
3	533.333333	705.666667	127.333333	91.833333	164.333333	1601.166667	
4	277.428571	327.000000	497.000000	76.142857	24.428571	1844.142857	
5	253.714286	360.142857	255.857143	115.857143	34.428571	1231.571429	
6	332.428571	415.428571	157.428571	115.857143	43.428571	1157.714286	
7	480.750000	640.375000	315.000000	64.875000	37.125000	1556.375000	
8	660.666667	923.333333	47.666667	49.000000	188.666667	1078.666667	
9	413.750000	483.625000	249.250000	133.250000	33.250000	1491.250000	
10	688.000000	1060.571429	89.428571	103.714286	120.714286	1094.428571	
11	149.000000	190.000000	379.500000	71.833333	7.666667	1406.166667	
12	366.000000	528.000000	87.000000	53.000000	99.000000	921.000000	
13	355.000000	456.500000	203.500000	67.500000	47.000000	971.500000	
14	444.857143	672.428571	173.714286	87.714286	47.428571	1081.714286	
15	305.000000	574.333333	78.166667	69.666667	77.333333	751.166667	
16	307.500000	404.500000	115.750000	95.250000	36.250000	881.250000	
17	159.000000	194.000000	297.000000	73.000000	4.000000	851.000000	
18	439.285714	528.571429	369.857143	104.285714	82.285714	689.285714	
19	242.800000	305.600000	305.000000	89.400000	30.200000	1589.200000	
20	230.666667	353.333333	100.000000	42.333333	68.000000	908.000000	
21	246.000000	289.625000	413.500000	99.875000	29.125000	1501.125000	
22	402.500000	477.000000	156.333333	59.500000	35.166667	1270.666667	
23	255.375000	308.875000	250.750000	73.625000	29.750000	1112.750000	
24	293.833333	336.833333	706.166667	134.166667	47.500000	1418.833333	
25	394.200000	625.800000	79.200000	57.600000	61.800000	877.400000	
26	362.500000	530.000000	79.500000	35.500000	104.000000	730.000000	
27	232.125000	266.250000	177.000000	70.125000	41.500000	1499.375000	
28	345.000000	530.250000	80.625000	56.625000	96.625000	855.500000	
29	406.000000	671.000000	64.166667	40.000000	60.666667	729.666667	

	TOV	PPG	Unnamed: 0	Year	Draft pick	Height (No Shoes)	\
0	277.000000	24.220056	22	2009	3.0	76.00	
1	243.125000	23.666221	41	2009	7.0	74.00	
2	125.500000	11.846473	227	2013	27.0	84.50	
3	126.166667	23.184079	151	2012	1.0	81.25	
4	216.428571	23.635619	197	2012	6.0	73.75	
5	107.285714	18.191087	117	2011	30.0	78.00	
6	94.714286	17.481569	103	2011	15.0	78.00	
7	186.750000	21.851485	0	2009	1.0	80.50	
8	129.333333	14.522007	70	2010	33.0	82.50	
9	193.250000	19.478483	51	2010	10.0	79.75	
10	138.428571	13.921590	200	2012	9.0	81.75	
11	171.833333	18.898851	138	2011	60.0	68.75	
12	71.000000	12.118421	160	2012	18.0	80.25	
13	110.500000	12.098302	408	2016	27.0	80.25	
14	151.428571	14.141847	88	2010	7.0	81.75	
15	110.500000	9.658378	214	2013	12.0	82.75	
16	60.250000	11.447905	229	2013	3.0	79.50	
17	52.000000	10.378049	489	2017	51.0	73.25	
18	155.571429	9.049048	178	2012	35.0	77.75	
19	169.000000	20.745446	172	2012	3.0	75.25	
20	81.333333	12.227923	354	2015	32.0	79.00	
21	164.625000	19.577720	140	2011	9.0	71.50	
22	108.833333	16.862456	107	2011	19.0	78.50	
23	145.500000	14.991876	90	2010	9.0	78.75	
24	292.500000	18.763386	50	2010	1.0	74.75	
25	103.200000	11.987294	110	2011	22.0	78.00	
26	92.500000	9.532639	467	2017	22.0	81.00	
27	131.625000	19.393003	99	2011	11.0	77.75	
28	104.375000	11.852725	67	2010	3.0	80.75	
29	86.666667	9.332197	126	2011	4.0	79.50	

	Height (With Shoes)	Wingspan	Standing reach	Vertical (Max)	\
0	77.25	82.75	103.5	37.0	
1	75.25	75.50	97.0	35.5	
2	86.00	92.50	115.0	29.0	
3	82.50	89.50	108.0	NaN	
4	74.75	79.75	95.5	39.5	
5	79.75	79.50	101.5	39.0	
6	79.00	87.00	106.0	32.0	
7	82.00	83.25	105.0	35.5	
8	83.50	91.00	113.0	31.5	
9	80.75	83.25	107.0	NaN	
10	83.75	90.25	109.5	33.5	
11	70.25	73.75	91.5	40.0	
12	81.50	86.25	107.0	34.5	
13	81.50	87.25	107.5	36.5	

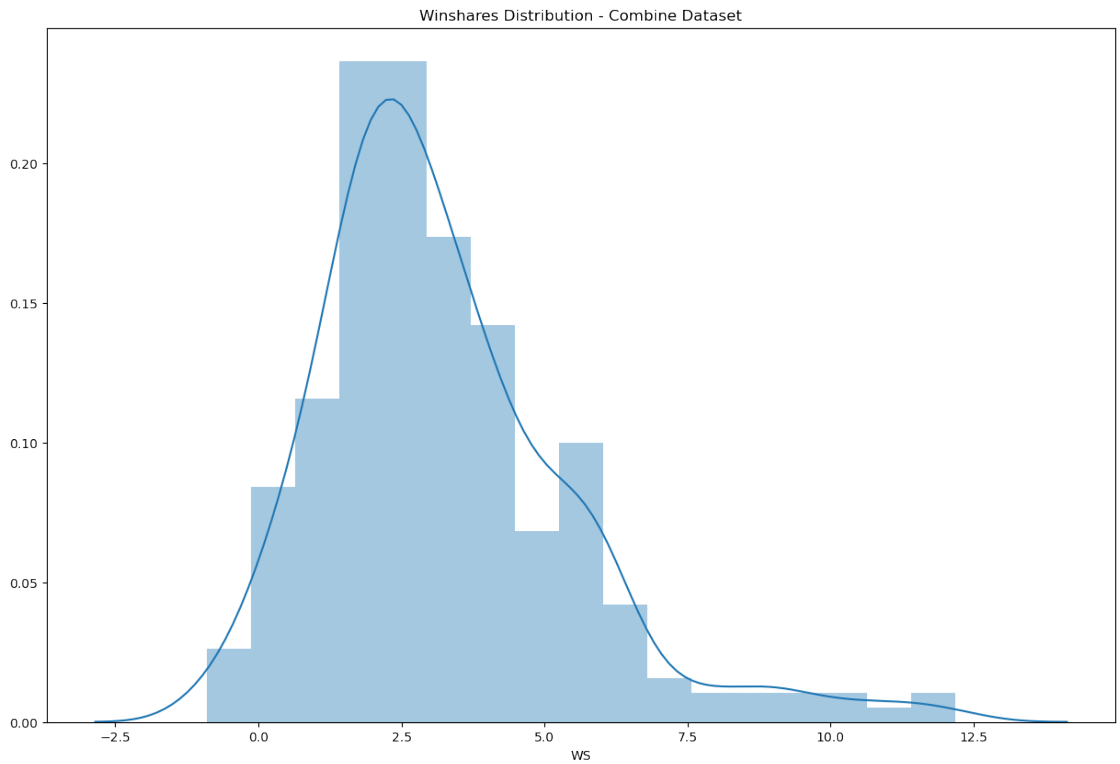
14	83.00	86.25	108.5	29.0
15	84.00	88.50	109.5	33.0
16	80.50	85.50	105.5	36.0
17	74.50	76.00	96.5	33.5
18	79.50	85.25	105.0	33.0
19	76.75	80.00	100.0	39.0
20	79.50	88.25	109.0	NaN
21	73.00	75.50	91.5	39.5
22	79.75	83.00	103.5	37.5
23	80.00	79.75	103.0	34.5
24	76.00	81.25	101.5	39.0
25	79.50	84.00	108.0	35.0
26	82.25	89.25	109.5	35.5
27	79.25	81.00	103.5	31.5
28	82.25	88.00	110.0	35.5
29	80.75	85.25	108.5	35.0

	Vertical (Max Reach)	Vertical (No Step)	Vertical (No Step Reach) \
0	140.5	31.5	135.0
1	132.5	29.5	126.5
2	144.0	25.0	140.0
3	NaN	NaN	NaN
4	135.0	34.5	130.0
5	140.5	32.0	133.5
6	138.0	25.5	131.5
7	140.5	32.0	137.0
8	144.5	27.0	140.0
9	NaN	NaN	NaN
10	143.0	31.5	141.0
11	131.5	31.5	123.0
12	141.5	29.5	136.5
13	144.0	30.5	138.0
14	137.5	25.0	133.5
15	142.5	28.5	138.0
16	141.5	27.0	132.5
17	130.0	28.0	124.5
18	138.0	28.0	133.0
19	139.0	33.0	133.0
20	NaN	NaN	NaN
21	131.0	32.0	123.5
22	141.0	31.5	135.0
23	137.5	30.5	133.5
24	140.5	30.0	131.5
25	143.0	30.5	138.5
26	145.0	31.5	141.0
27	135.0	26.5	130.0
28	145.5	31.5	141.5

```
29               143.5              30.0                    138.5

    Weight  Body Fat  Hand (Length)  Hand (Width)  Bench  Agility  Sprint
0    222.0      10.1            NaN           NaN   17.0    11.10    3.13
1    181.0       5.7            NaN           NaN   10.0    11.07    3.28
2    238.0       4.4           9.75         10.00    7.0    12.85    3.57
3    222.0       7.9           9.00          8.50    NaN      NaN     NaN
4    189.0       5.9           8.75          9.75   13.0    11.15    3.34
5    222.0       5.4           9.00          9.00   14.0    11.92    3.15
6    227.0       5.4           9.75         11.25    3.0    11.45    3.15
7    248.0       8.2            NaN           NaN   22.0    10.95    3.28
8    227.0       5.5           8.25         10.25   12.0    11.83    3.54
9    214.0       5.0           8.50          9.00    4.0      NaN     NaN
10   279.0       7.5           9.50          9.50   10.0    10.83    3.39
11   186.0       6.7           8.25          9.00   13.0    10.49    3.14
12   252.0       7.7           9.00         10.00   12.0    11.57    3.40
13   227.0       5.2           9.25         10.00    NaN    11.25    3.41
14   247.0      11.2           8.75          9.50   15.0    12.10    3.35
15   255.0       6.7           9.50         11.00   16.0    11.85    3.40
16   198.0       6.7           8.75          9.25    9.0    11.25    3.40
17   175.0       6.9           8.25          8.75    NaN    11.00    3.19
18   236.0      11.3           9.00          9.50    9.0    11.01    3.40
19   202.0       6.0           8.50          9.00    8.0    10.95    3.28
20   253.0      11.9           9.00          9.75    NaN      NaN     NaN
21   184.0       5.9           8.00          9.00    7.0    10.87    3.16
22   223.0       8.4           8.75          9.00   12.0    10.96    3.17
23   211.0       6.9           8.50          9.25   10.0    11.73    3.22
24   196.0       5.6           8.25          9.50    NaN    10.84    3.14
25   225.0       6.3           8.50         10.25   16.0    11.35    3.26
26   234.0       7.4           9.50         10.50    NaN    11.82    3.21
27   206.0       8.0           8.75          9.25    5.0    10.99    3.24
28   245.0       6.5           8.75          9.25   14.0    11.74    3.25
29   227.0       6.2           8.75          9.25    9.0    10.92    3.26
```

[31]:
```python
# let's look at the Win Share distribution for the merged data set of athletes␣
↪who
# participated in the combine, and their average win share for their 58 game␣
↪seasons

plt.figure(figsize=(15,10))
plt.tight_layout()
seabornInstance.distplot(combine_nba_merge['WS'])
plt.title('Winshares Distribution - Combine Dataset')
```

[31]:
```
Text(0.5, 1.0, 'Winshares Distribution - Combine Dataset')
```



Winshare for our combine data appears to be a "near normal" distribution, and we see the expected pattern of things being skewed left.

[32]:
```
# subset the data to compare draft picks to NBA performance

combine_subset = combine_nba_merge[['Draft pick', 'WS', 'TRB', 'AST', 'STL',␣
 ↪'PPG', 'Height (No Shoes)',
                                    'BLK']]
combine_subset.shape
```

[32]: (247, 8)

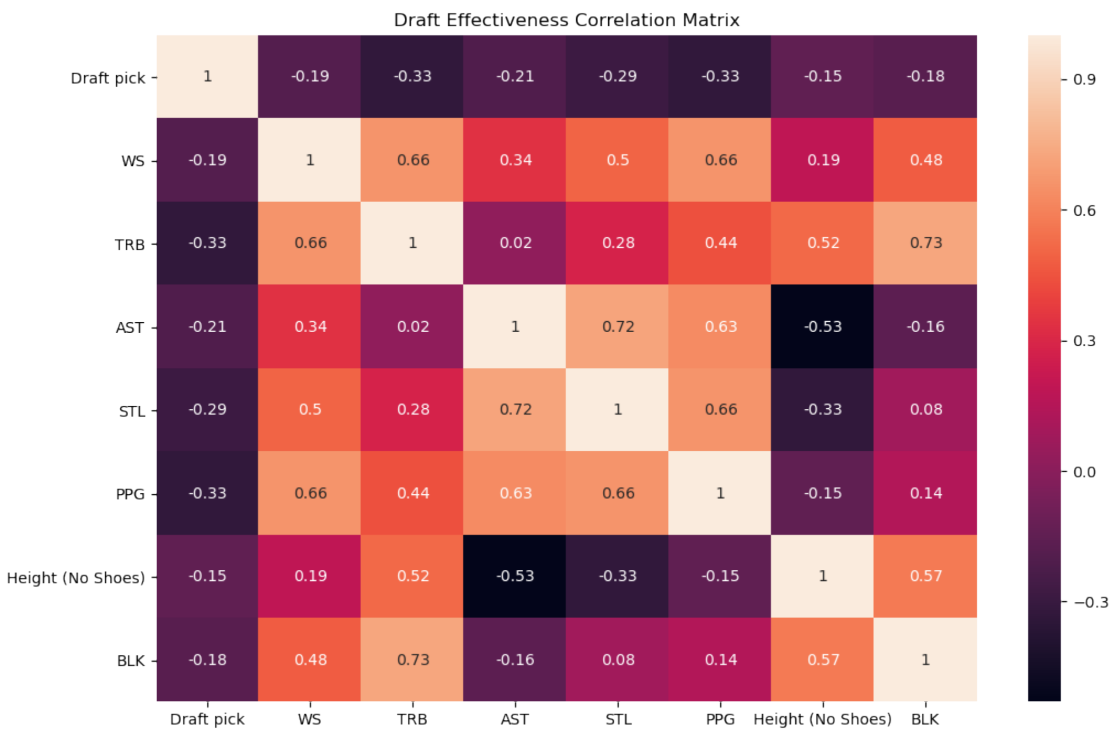
[33]:
```
# summary statistics for our measured dataset

combine_subset.describe()
```

[33]:
```
        Draft pick          WS         TRB         AST         STL  \
count   232.000000  247.000000  247.000000  247.000000  247.000000
mean     22.142241    3.219196  292.352095  145.027386   54.804547
std      14.150445    2.217590  150.791097  111.933981   26.628137
min       1.000000   -0.900000   32.000000   11.000000    9.000000
25%      10.000000    1.775000  190.250000   67.375000   34.416667
50%      20.000000    2.750000  261.000000  106.500000   51.000000
```

```
75%      33.000000    4.268750   359.500000  190.000000    68.850000
max      60.000000   12.180000  1060.571429  706.166667   135.125000

              PPG  Height (No Shoes)         BLK
count  247.000000         247.000000  247.000000
mean     9.586418          77.743927   33.213741
std      4.509504           3.246005   29.874199
min      1.881356          68.750000    2.000000
25%      6.679919          75.500000   13.750000
50%      8.644622          77.750000   24.800000
75%     11.973208          80.000000   41.861111
max     24.220056          85.250000  188.666667
```

[34]:
```python
# plot a correlation matrix

draft_corr = combine_subset.corr().round(2)
sns.heatmap(data=draft_corr, annot=True)
plt.title('Draft Effectiveness Correlation Matrix')
```

[34]: Text(0.5, 1.0, 'Draft Effectiveness Correlation Matrix')

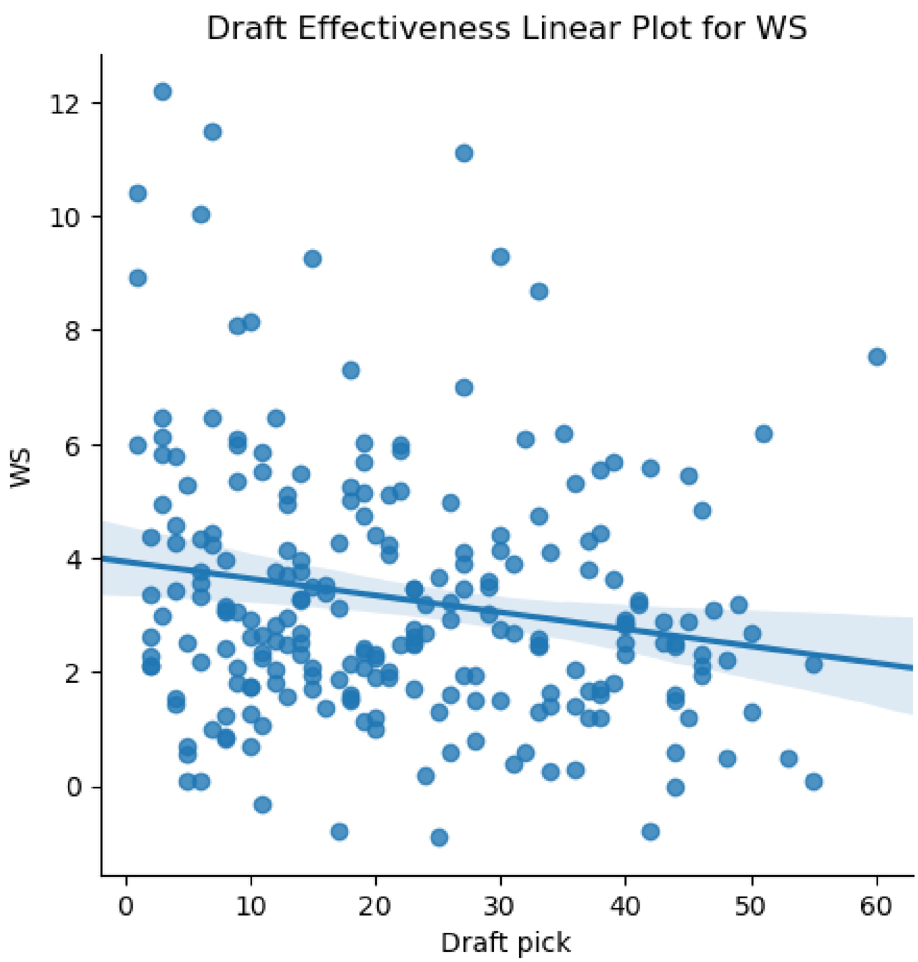


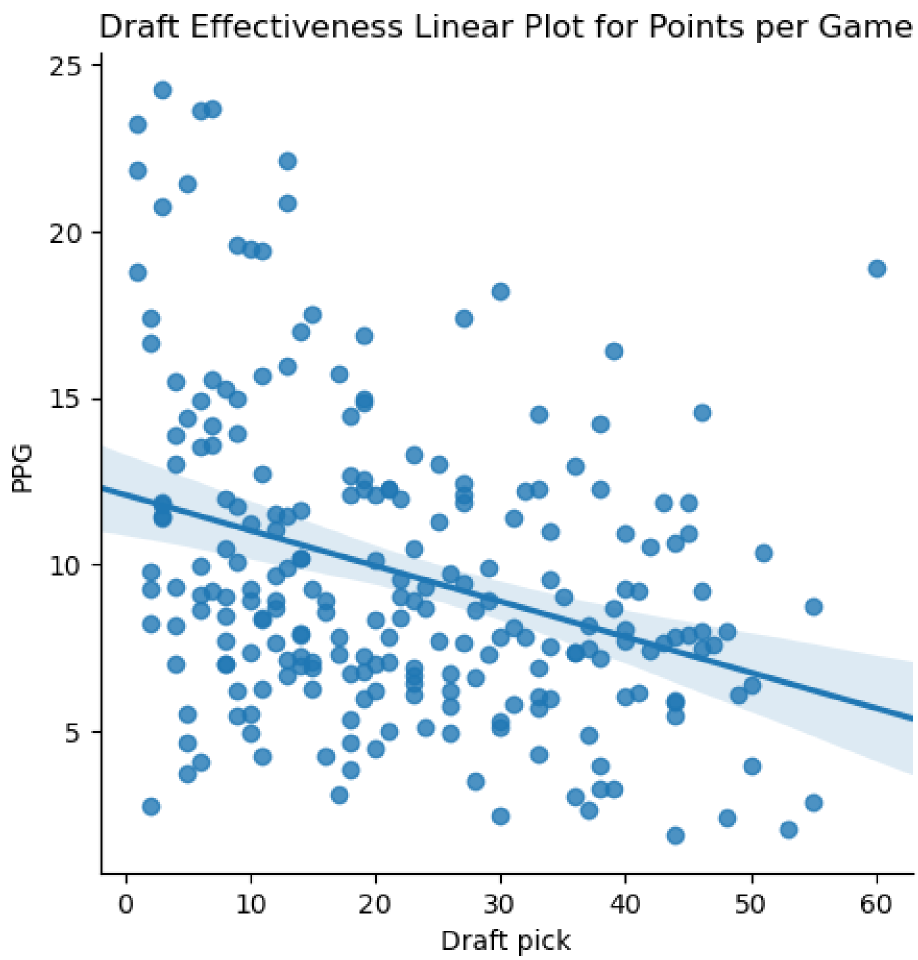
	Draft pick	WS	TRB	AST	STL	PPG	Height (No Shoes)	BLK
Draft pick	1	-0.19	-0.33	-0.21	-0.29	-0.33	-0.15	-0.18
WS	-0.19	1	0.66	0.34	0.5	0.66	0.19	0.48
TRB	-0.33	0.66	1	0.02	0.28	0.44	0.52	0.73
AST	-0.21	0.34	0.02	1	0.72	0.63	-0.53	-0.16
STL	-0.29	0.5	0.28	0.72	1	0.66	-0.33	0.08
PPG	-0.33	0.66	0.44	0.63	0.66	1	-0.15	0.14
Height (No Shoes)	-0.15	0.19	0.52	-0.53	-0.33	-0.15	1	0.57
BLK	-0.18	0.48	0.73	-0.16	0.08	0.14	0.57	1

Looking at the correlation matrix, we see that the correlation between draft position and stats like PPG, STL is weak at best, there is a relationship, but it isn't significant. Let's look at linear model plot of this data

[35]:
```python
# linear plot to visualize the correlation between draft position and win share

sns.lmplot(x='Draft pick', y='WS', data=combine_subset)
plt.title('Draft Effectiveness Linear Plot for WS')


# linear plot to visualize the correlation between draft position and PPG
sns.lmplot(x='Draft pick', y='PPG', data=combine_subset)
plt.title('Draft Effectiveness Linear Plot for Points per Game')
```

[35]:
```
Text(0.5, 1.0, 'Draft Effectiveness Linear Plot for Points per Game')
```





The relationship we see is that while the top players in the league do tend to have been first round or low draft picks not all low draft picks turn out to be great players. Looking at the graph the trend seems to be that high performing first round draft picks are outliers, while being above average looks to be 50/50 odds. From our earlier analysis we know that average PPG is 11.2 and average win share is 4.43, so let's use that information to evaluate just how many of those players perform above average.

[36]:
```
# first we isolate the athletes taken in the top 15% of the draft

high_draft = combine_subset[(combine_subset['Draft pick'] <= 15)]
high_draft.shape
```

[36]:
```
(93, 8)
```

We had 247 total athletes and 93 or 37.65% were taken in the top 25% (first 15 picks) of the draft, which gives us another finding: regardless of performance trends, higher draft picks do at least seem to be more likely to play a 58 game season. This isn't particularly surprising given the investment teams make into those players.

[38]:
```python
# add columns for above average, below average, top 25%, not top 25% to make␣
↪counting in each category
# as well as calculating sample proportions for the purposes of statistical␣
↪signifigance tests easier.

# add the above_average, below average columns


avg_ws = nba_subset['WS'].mean()
avg_ppg = nba_subset['PPG'].mean()


high_draft.loc[:, 'ws_above_average'] = high_draft.loc[:,'WS'].apply(lambda x:␣
↪0 if x <= avg_ws else 1)
high_draft.loc[:, 'ws_below_average'] = high_draft.loc[:,'WS'].apply(lambda x:␣
↪0 if x >= avg_ws else 1)
high_draft.loc[:, 'ppg_above_average'] = high_draft.loc[:,'PPG'].apply(lambda x:
↪ 0 if x <= avg_ppg else 1)
high_draft.loc[:, 'ppg_below_average'] = high_draft.loc[:,'PPG'].apply(lambda x:
↪ 0 if x >= avg_ppg else 1)


high_draft
```

[38]:
```
     Draft pick         WS         TRB         AST         STL        PPG  \
0           3.0  12.180000  400.000000  474.300000  118.900000  24.220056
1           7.0  11.475000  347.875000  517.500000  135.125000  23.666221
3           1.0  10.400000  705.666667  127.333333   91.833333  23.184079
4           6.0  10.042857  327.000000  497.000000   76.142857  23.635619
6          15.0   9.257143  415.428571  157.428571  115.857143  17.481569
..          …         …          …          …          …         …
225        10.0   0.700000   72.000000  100.000000   30.500000   7.366595
229         5.0   0.550000  131.500000  154.500000   31.000000   5.517554
239         6.0   0.100000  191.000000  356.000000   82.000000  13.506173
240         5.0   0.100000  166.000000  188.000000   78.000000   3.756410
242        11.0  -0.300000  348.000000  223.000000   43.000000   8.410256

     Height (No Shoes)         BLK  ws_above_average  ws_below_average  \
0                76.00   38.300000                 1                 0
1                74.00   17.250000                 1                 0
3                81.25  164.333333                 1                 0
4                73.75   24.428571                 1                 0
6                78.00   43.428571                 1                 0
..                 …          …               …                …
225              72.75    3.000000                 0                 1
229              76.50   13.000000                 0                 1
239              71.25    2.000000                 0                 1
```

```
240              75.00   36.000000                0                 1
242              77.00   10.000000                0                 1

     ppg_above_average  ppg_below_average
0                    1                  0
1                    1                  0
3                    1                  0
4                    1                  0
6                    1                  0
..                 …                  …
225                  0                  1
229                  0                  1
239                  1                  0
240                  0                  1
242                  0                  1

[93 rows x 12 columns]
```

[39]:
```python
# count how many athletes above average for WS

ws_above_count = high_draft['ws_above_average'].sum()
ws_below_count = high_draft['ws_below_average'].sum()

ws_above_count
```

[39]: 31

[40]:
```python
# count how many athletes above average for PPG and calculate N

ppg_above_count = high_draft['ppg_above_average'].sum()
ppg_below_count = high_draft['ppg_below_average'].sum()
n = len(high_draft)
ppg_above_count
```

[40]: 43

[41]:
```python
import statsmodels.api as sm

# null hypothesis: 50/50 chance of selecting an above average player or
# an average player in the top 15 picks of the draft
# alternate hypothesis: the % of top 15 draft picks that are high scorers will
# be greater than those that are low scorers


pnull = 0.50
phat = ppg_above_count/n
```

ppg_null

```
sm.stats.proportions_ztest(phat * n, n, pnull, alternative='larger')
```

[41]:
```
(-0.7279311237543241, 0.7666721251426709)
```

Looking the above we see that with a p-value of 0.767 we can't reject the null hypothesis, meaning: in all likelihood the NBA's evaluation procedures are a little better than a coin flip from the perspetive of a high draft pick being an above average scorer.

[42]:
```
high_draft.loc[:, 'ws_above_75th%'] = high_draft.loc[:,'WS'].apply(lambda x: 0␣
 ↪if x <= 5.7 else 1)


high_draft.loc[:, 'ppg_above_75th%'] = high_draft.loc[:,'PPG'].apply(lambda x:␣
 ↪0 if x <= 14.082 else 1)
high_draft.loc[:, 'ppg_below_75th%'] = high_draft.loc[:,'PPG'].apply(lambda x:␣
 ↪0 if x >= 14.082 else 1)


high_draft
```

[42]:
```
     Draft pick         WS         TRB         AST         STL        PPG  \
0           3.0  12.180000  400.000000  474.300000  118.900000  24.220056
1           7.0  11.475000  347.875000  517.500000  135.125000  23.666221
3           1.0  10.400000  705.666667  127.333333   91.833333  23.184079
4           6.0  10.042857  327.000000  497.000000   76.142857  23.635619
6          15.0   9.257143  415.428571  157.428571  115.857143  17.481569
..          …          …           …           …           …          …
225        10.0   0.700000   72.000000  100.000000   30.500000   7.366595
229         5.0   0.550000  131.500000  154.500000   31.000000   5.517554
239         6.0   0.100000  191.000000  356.000000   82.000000  13.506173
240         5.0   0.100000  166.000000  188.000000   78.000000   3.756410
242        11.0  -0.300000  348.000000  223.000000   43.000000   8.410256

     Height (No Shoes)         BLK  ws_above_average  ws_below_average  \
0                76.00   38.300000                 1                 0
1                74.00   17.250000                 1                 0
3                81.25  164.333333                 1                 0
4                73.75   24.428571                 1                 0
6                78.00   43.428571                 1                 0
..                 …          …                …                …
225              72.75    3.000000                 0                 1
229              76.50   13.000000                 0                 1
239              71.25    2.000000                 0                 1
240              75.00   36.000000                 0                 1
```

```
242             77.00   10.000000                0                 1

     ppg_above_average  ppg_below_average  ws_above_75th%  ws_below_75th%  \
0                    1                  0               1               0
1                    1                  0               1               0
3                    1                  0               1               0
4                    1                  0               1               0
6                    1                  0               1               0
..                 …                  …             …             …
225                  0                  1               0               1
229                  0                  1               0               1
239                  1                  0               0               1
240                  0                  1               0               1
242                  0                  1               0               1

     ppg_above_75th%  ppg_below_75th%
0                  1                0
1                  1                0
3                  1                0
4                  1                0
6                  1                0
..               …              …
225                0                1
229                0                1
239                0                1
240                0                1
242                0                1

[93 rows x 16 columns]
```

[43]:
```python
# count number of athletes taken in the top 25% of the draft that produce win␣
↪shares in the 75th percentile:

ws_above_count = high_draft['ws_above_75th%'].sum()
ws_above_count_per = ws_above_count/n

print('# of athletes who produced win shares in the 75th percentile:',␣
↪ws_above_count)
print('% of athletes who produced win shares in the 75th percentile:',␣
↪ws_above_count_per)
```

```
# of athletes who produced win shares in the 75th percentile: 18
% of athletes who produced win shares in the 75th percentile: 0.1935483870967742
```

[44]:
```python
ppg_above_count = high_draft['ppg_above_75th%'].sum()
ppg_above_count_per = ppg_above_count/n
```

```
print('# of athletes who produced win shares in the 75th percentile:',␣
 ↪ppg_above_count)
print('% of athletes who produced win shares in the 75th percentile:',␣
 ↪ppg_above_count_per)
```

```
# of athletes who produced win shares in the 75th percentile: 26
% of athletes who produced win shares in the 75th percentile:
0.27956989247311825
```

Let's summarize the current findings:

- Only 31/93 athletes taken in the top 15 picks of the draft were at or above average in terms of win share or roughly 33%
- 43/93 athletes were above average in terms of PPG, but according to our statistical significance tests, the difference between above and below average isn't statistically significant. Meaning: when it comes to PPG it's roughly 50/50 whether or not a top 15 draft pick scores more than average
- 93/247 athletes were picked in the top 25% of the draft or 40% of our athletes, the number would've been 25% if every draft position had an equal chance of playing at least one 58 game season. This means that at the very least, higher draft picks get more playing time, probably due to some combination of the team's investment, expectations and talent.
- 19.35% of the athletes (18 total) selected in the top 15 of the draft produced win shares in the 75th percentile
- 27.96% of the athletes (26 total) selected in the top 15 of the draft produced PPG in the 75th percentile

[45]:
```
# analyze the data with just the data we have full combine data for

full_combine = combine_nba_merge.dropna()
full_combine.shape
```

[45]:
```
(102, 31)
```

[46]:
```
full_combine.describe()
```

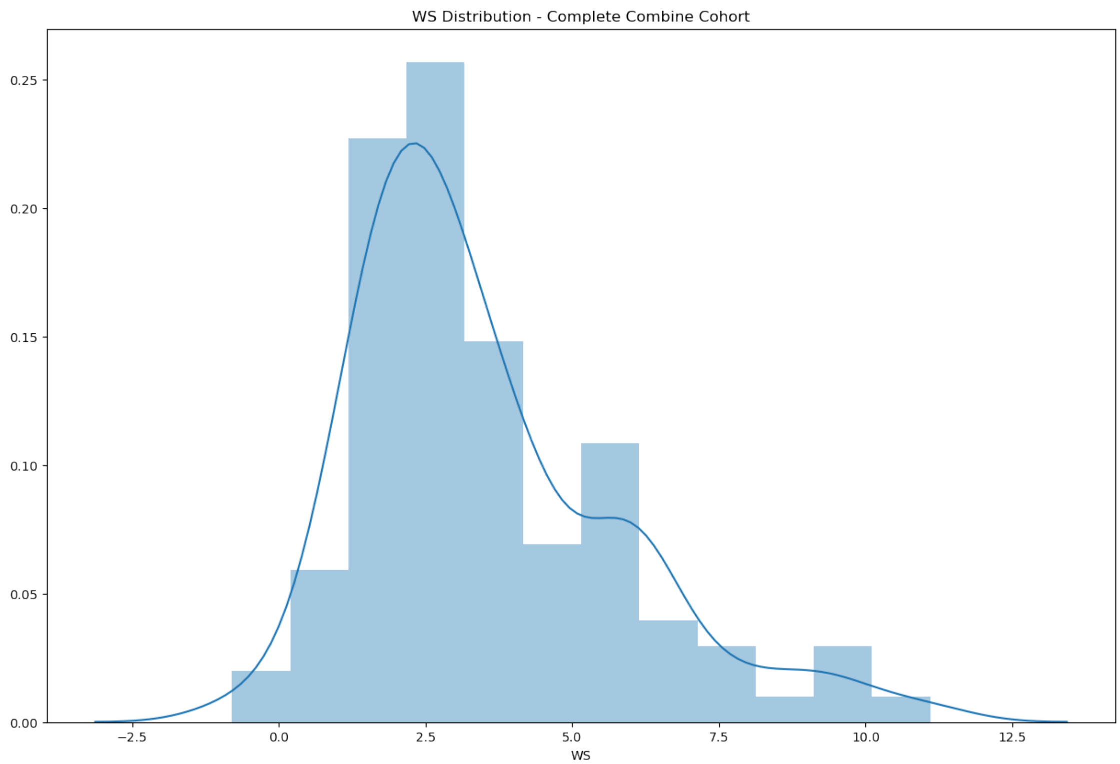
[46]:
```
                WS           G           MP         ORB         DRB  \
count   102.000000  102.000000   102.000000  102.000000  102.000000
mean      3.507382   71.688994  1706.222771   80.681182  231.321771
std       2.268007    5.246972   501.985141   67.842633  113.729105
min      -0.800000   58.000000   441.000000    8.000000   24.000000
25%       1.900000   68.083333  1340.700000   40.466667  152.016667
50%       2.841667   72.633333  1719.071429   56.107143  210.187500
75%       4.395000   75.906250  2085.375000   96.083333  280.250000
max      11.100000   80.000000  2843.857143  372.571429  688.000000

              TRB         AST         STL         BLK          PTS  \
count  102.000000  102.000000  102.000000  102.000000   102.000000
mean   312.002953  134.717678   55.003221   35.156435   707.429369
```

```
std     173.164928   99.099288   25.554404   33.046846   340.869286
min      32.000000   22.000000   11.250000    2.000000   111.000000
25%     199.783333   64.462500   37.083333   15.000000   495.843750
50%     268.375000  101.866667   52.600000   25.750000   650.773810
75%     377.642857  176.178571   69.916667   42.041667   875.112500
max    1060.571429  497.000000  124.000000  188.666667  1844.142857

              TOV         PPG  Unnamed: 0         Year  Draft pick  \
count  102.000000  102.000000  102.000000   102.000000  102.000000
mean    87.623125    9.712010  181.784314  2012.107843   20.794118
std     45.253674    4.440690   89.057955     1.566398   13.232377
min     20.000000    1.881356   52.000000  2010.000000    2.000000
25%     60.500000    6.743794  110.500000  2011.000000   10.000000
50%     74.683333    8.980595  171.000000  2012.000000   19.000000
75%    108.383929   11.875959  228.500000  2013.000000   29.750000
max    250.000000   23.635619  364.000000  2015.000000   60.000000

       Height (No Shoes)  Height (With Shoes)    Wingspan  Standing reach  \
count          102.00000           102.000000  102.000000      102.000000
mean            77.89951            79.213235   82.699510      103.387255
std              3.34084             3.334376    3.923668        4.819467
min             68.75000            70.250000   70.750000       89.500000
25%             76.00000            77.250000   80.000000      100.500000
50%             78.00000            79.375000   82.875000      104.000000
75%             80.18750            81.437500   85.500000      106.875000
max             84.50000            86.000000   92.500000      115.000000

       Vertical (Max)  Vertical (Max Reach)  Vertical (No Step)  \
count      102.000000            102.000000          102.000000
mean        35.676471            139.063725           30.029412
std          3.572202              3.793997            3.089285
min         28.000000            129.500000           23.000000
25%         33.500000            136.000000           28.000000
50%         36.000000            140.000000           30.000000
75%         37.500000            142.000000           32.000000
max         44.000000            146.000000           38.000000

       Vertical (No Step Reach)      Weight    Body Fat  Hand (Length)  \
count                102.000000  102.000000  102.000000     102.000000
mean                 133.416667  217.705882    6.806863       8.781863
std                    4.399079   23.329999    1.850997       0.499593
min                  123.000000  171.000000    3.200000       7.500000
25%                  130.625000  199.250000    5.400000       8.500000
50%                  133.500000  221.500000    6.600000       8.750000
75%                  136.500000  234.000000    7.775000       9.000000
max                  142.000000  279.000000   11.400000      10.000000
```

```
       Hand (Width)       Bench     Agility      Sprint
count    102.000000  102.000000  102.000000  102.000000
mean       9.460784   10.500000   11.238333    3.282843
std        0.730984    4.754883    0.521675    0.129258
min        7.000000    1.000000   10.070000    3.020000
25%        9.000000    7.000000   10.890000    3.190000
50%        9.500000   10.000000   11.175000    3.270000
75%       10.000000   14.000000   11.477500    3.350000
max       11.250000   22.000000   12.850000    3.810000
```

[47]:
```
# let's take a look at the Win Share distribution

plt.figure(figsize=(15,10))
plt.tight_layout()
seabornInstance.distplot(full_combine['WS'])
plt.title('WS Distribution - Complete Combine Cohort')
```

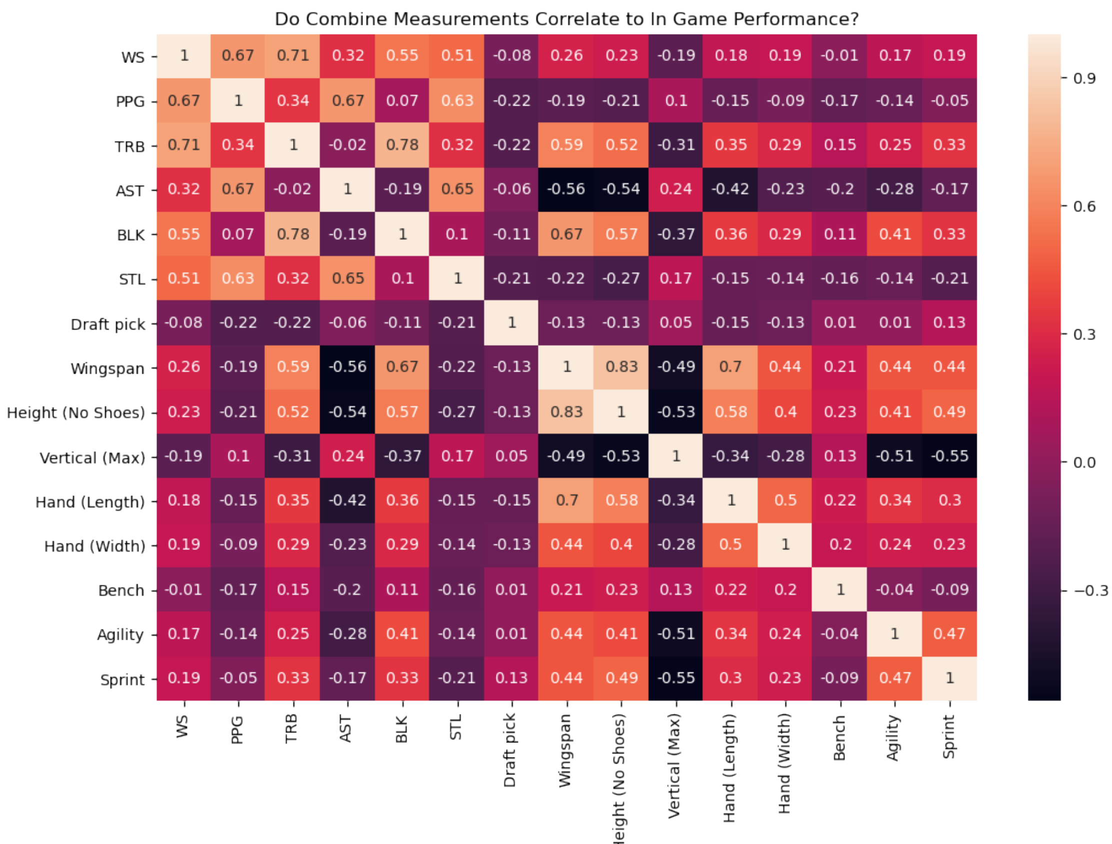
[47]:
```
Text(0.5, 1.0, 'WS Distribution - Complete Combine Cohort')
```



[48]:
```
# compute correlation matrix
```

```python
combine_correlation = full_combine[['WS', 'PPG', 'TRB', 'AST', 'BLK', 'STL', 'Draft pick', 'Wingspan', 'Height (No Shoes)', 'Vertical (Max)', 'Hand (Length)',
                                    'Hand (Width)', 'Bench', 'Agility', 'Sprint']]

combine_corr = combine_correlation.corr().round(2)
sns.heatmap(data=combine_corr, annot=True)
plt.title('Do Combine Measurements Correlate to In Game Performance?')
```

[48]: Text(0.5, 1.0, 'Do Combine Measurements Correlate to In Game Performance?')



According to the correlation matrix there doesn't seem to be a strong correlation between any of the selected combine measurements and win share or PPG, as the "strongest" coefficients were:

- 0.26 for wingspan and winshare
- 0.19 for sprint and winshare
- 0.23 for height and winshare
- -0.22 and -0.26 for PPG vs. draft pick and height respectively

We also had some near strong coefficients for in game stats outside of our primary ones: * 0.67 for wingspan and blocks * 0.59 for wingspan and total rebounds

We also have some other relationships that are "stronger" than most, but those are probably more a factor of the bodytypes that play certain positions.

- -0.56 and -0.55 for wingspan and height respectively vs. assists, which is probably more a reflection of the fact that players that play positions like point guard that create more assists tend to be shorter than larger athletes that play center or power forward.

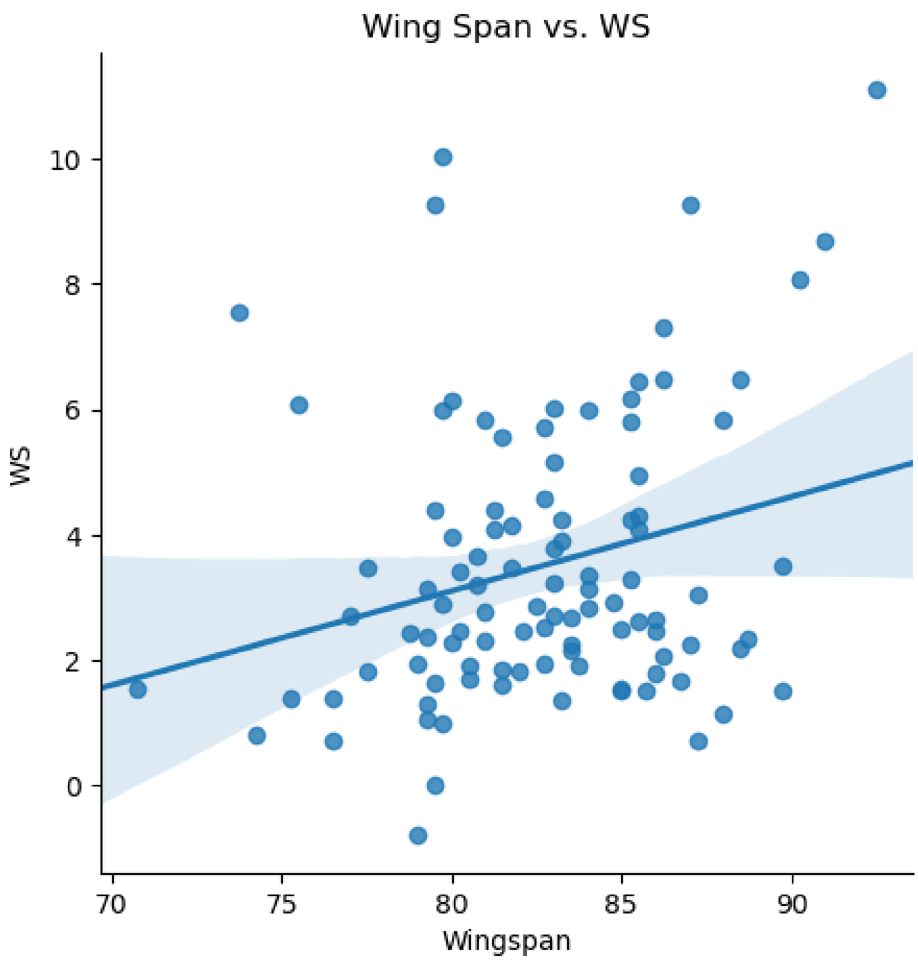
Going back to the Kevin Durant example, bench press had the weakest relationships of any combine measurement with win share and PPG (-0.01 and -0.17 respectively), supporting Kevin's claim that on court skills matter more than things like strength and speed.

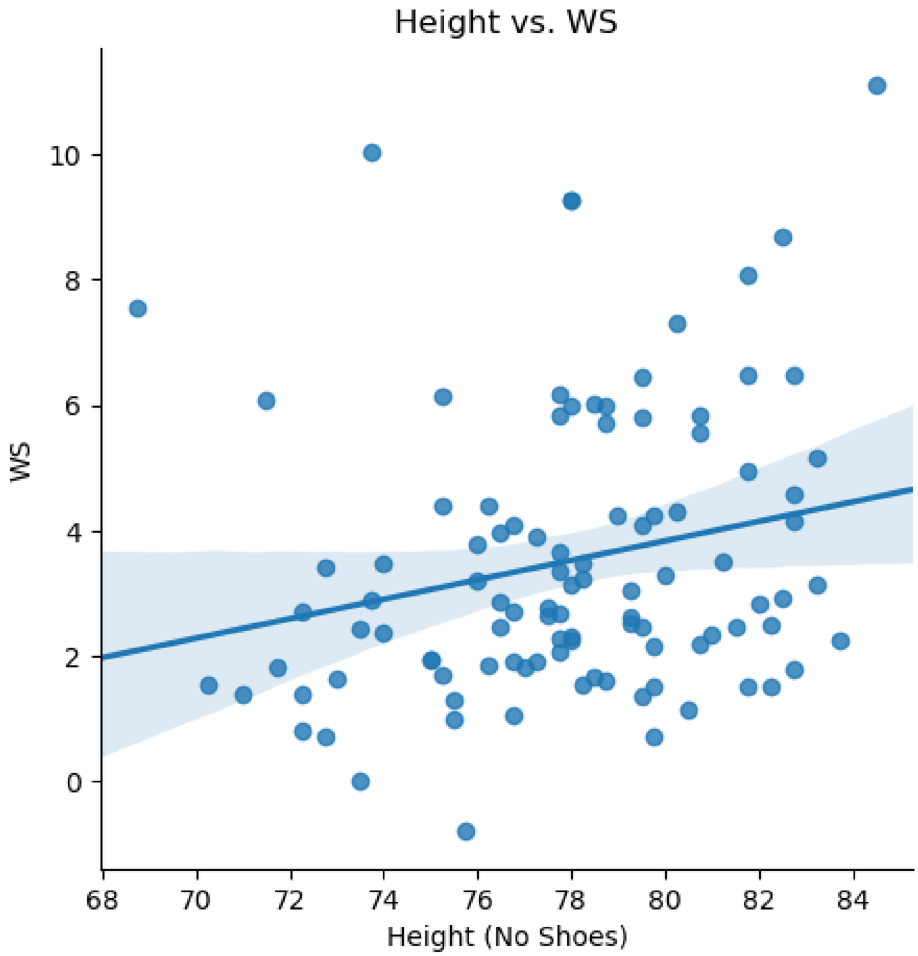
The strongest relationships were between the individual combine measurements, however, they're more a function of human anatomy, however the interesting thing is that height and hand length don't have a strong correlation. * Height and wingspan(0.83) * Hand length and wingspan (0.7)

If we were going to do a regression or similar analysis we might have to take certain correlated independent variables out to avoid collinearity, but we don't have a strong enough relationship between WS or PPG to pursue that avenue. We also don't see a strong correlation between any of the combine measurements and draft position.

[49]:
```python
# what does a linear plot look like for the strongest correlators for win share?
↪

# linear plot to visualize the correlation between WS and Wingspan
sns.lmplot(x='Wingspan', y='WS', data=combine_correlation)
plt.title('Wing Span vs. WS')


# linear plot to visualize the correlation between WS and Agility
sns.lmplot(x='Height (No Shoes)', y='WS', data=combine_correlation)
plt.title('Height vs. WS')
```

[49]:
```
Text(0.5, 1.0, 'Height vs. WS')
```


Wing Span vs. WS
WS
Wingspan
0
2
4
6
8
10
70
75
80
85
90



Both linear plots show a fairly weak relationship between height and win shares and wingspan and win shares. Similar to the draft there is a trend where the players with the best win shares will tend to be taller or have long arms, but only a minority of all the players with those characteristics will put up above average numbers. So while all things being equal the larger athlete will perform better, the NBA doesn't appear to have found a way to get to a point where one could definitively say that "all things are equal" between two athletes.

Summarizing all of our findings:

- There doesn't seem to be any significant relationship between the combine measurements and win share or PPG, which supports the idea that the combine isn't of much value to teams.
- One area where the combine could show value is that if a team is trying to decide between players who appear to be of similar skill, choosing the taller player or the athlete with longer arms should result in better defensive numbers in terms of blocks and rebounds.
- There wasn't a significant relationship between combine performance and draft position, thus supporting the idea that the combine doesn't help players.

- 19.35% of the athletes (18 total) selected in the top 15 of the draft produced win shares in

the 75th percentile
- 27.96% of the athletes (26 total) selected in the top 15 of the draft produced PPG in the 75th percentile
- The game's best players were nearly always top draft picks, but a top draft pick is not necessarily going to be a top player, in fact the odds are against a top draft pick being a top player.
- Only 31/93 (33%) athletes taken in the top 15 picks of the draft were at or above average in terms of win share
- 43/93 athletes were above average in terms of PPG, but according to our statistical significance tests, the was no real difference between the % of players above or below the PPG average. Meaning: when it comes to PPG it's roughly 50/50 whether or not a top 15 draft pick is going to be an above average scorer.

- 93/247 athletes were picked in the top 25% of the draft or 37.65% of the athletes in the data set a the number that would've been 25% if every draft position had an equal chance of playing at least one 58 game season, this means that at the very least, higher draft picks get more playing time, probably due to some combination of the team's investment, expectations and talent.

Overall it doesn't appear that the NBA has the "formula" to identify above average players, because while the best players do tend to be high draft picks, the odds of a top draft being above average is 50/50 at best and barely 28% will perform in the 75th percentile for PPG and barely 20% will reach that level for win shares. It also doesn't appear that the combine has much value in terms of giving teams additional information to evaluate players with, as the data collected doesn't appear to be particularly relevant to in game performance outside of blocks. While our data set was small and barely spanned a decade, it does appear that on a preliminary basis NBA star Kevin Durant was correct when he asserted that the NBA Combine was a "waste of time" as it wasn't relevant to basketball. It's worth noting that Kevin was ranked near the bottom for combine performances as far as the athletic measurements (speed, agility, strength and jumping ability), but was selected number two overall and is already a future hall of famer with multiple years of being one of the NBA's best ahead of him.

Future Improvements:

1. Get a broader dataset that has data spanning at least 20 seasons.
2. Use the above to re-analyze draft position vs. in game performance
3. Acquire 20 years of combine data to see if the trends change with more data
4. Create a visualization showing how many players from a given draft were above average, to see if the NBA's evaluation methods have improved over time
5. See if I can repeat the above for the WNBA to see if evaluating patterns are similar to the NBA

```
[ ]:
```