Analyzing Fantasy Football Data

Alex Clark

Abstract

I used a dataset that I compiled from one of my fantasy football leagues. This dataset has 10 years worth of matchup, draft and standings data, and three years worth of lineup data. I aimed to answer how the tight end position could be scored differently and if there is a correlation between total points for each position and where a team ends up in the final standings. I found that the tight end scoring could be improved on, by increasing the points each tight end receive for every catch they make. I also found a slight correlation for teams who have higher scoring quarterback and where they end up in the standings, the rest of the positions were inconclusive.

Motivation

In fantasy football, tight ends typically don’t score many points. They tend to lag behind the scoring of running backs and wide receivers. I want to see what changes could be made to bring the tight end scoring closer to the scoring of running backs and wide receivers.

I also want to get more insight into fantasy football and which position I should be targeting more heavily in my upcoming draft. I want to see if there is a correlation to where teams have ended up in the standings and the total points they had for each position (QB, RB, WR, TE, K, DEF).

Dataset(s)

I used the dataset I curated of my fantasy football league. This dataset has over 10 years of standings, matchup and draft data. This dataset also has three years of lineup data.

Standings data: Final standings from 2008-2017 (wins, loss, total points, etc.)

Matchup data: Matchups of each week from 2008-2017 (teams, points, wins, losses)

Draft data: Record of the draft from 2008-2017 (round, pick, player, team)

Lineup data: Lineup each team started for each week from 2015-2017 (week, team, position, points, game statistics)

Dataset: https://www.kaggle.com/clarkbar36/ron-mexico-fantasy-football-league

Data Preparation and Cleaning

There is cleaning I do each week in the football season when I scrape this data from the website. There was little cleaning I needed to do for this analysis, mostly just subsetting the data to isolate the data I needed to analyze.

Research Question(s)

TE Point Change: How can changing the points a tight end receives per catch effect how the position stacks up to running backs and wide receivers each week?

Position Correlation: Is there any correlation to where a fantasy team ends up in the standings and the total points they scored at each position?

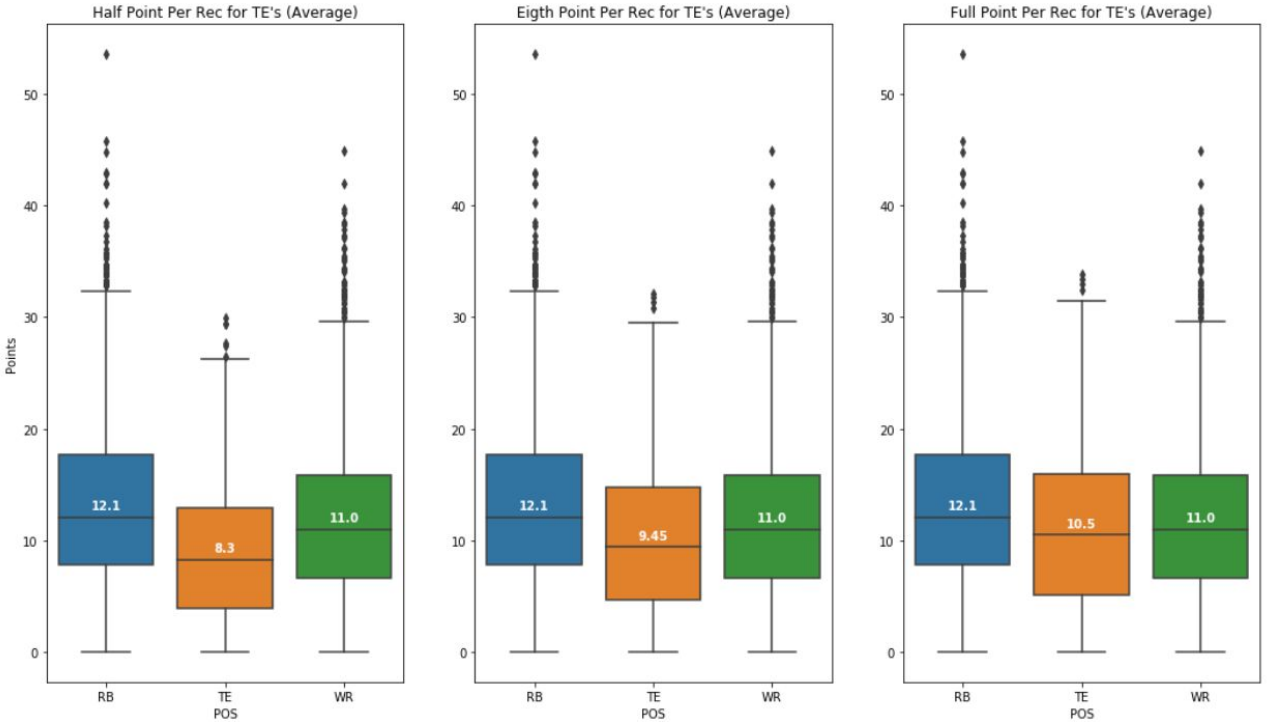
Methods

For the tight end analysis I summarized the data by position and points, then calculated the differences tight ends would receive for different point denominations for a catch. I then put the findings into a box plot to easily see how the changes could affect the point scoring.

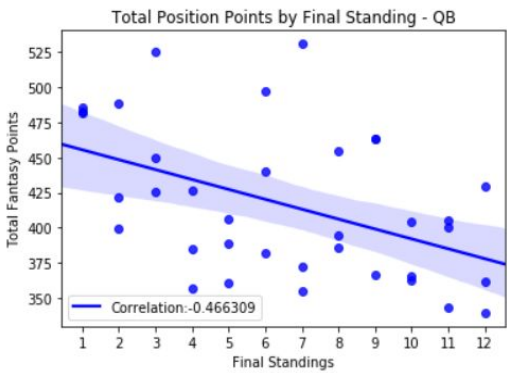
For the position correlation analysis, I summarized the lineups data by season, owner, position and then points, and i combined that with the standings data. Then I did a scatter plot with a best-fit line to see the correlation.

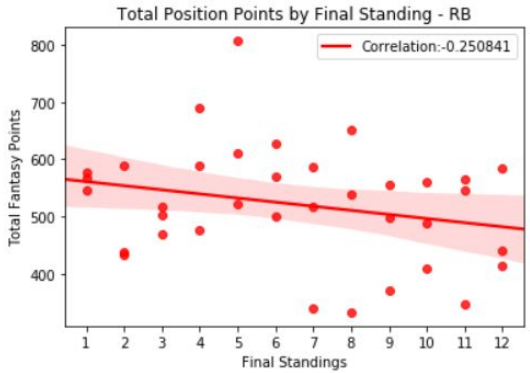
Findings - TE Point Change

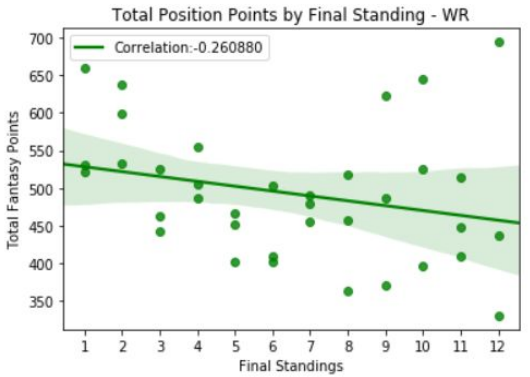
- By increasing the point per reception value for tight ends, the median tight end value per week is more in line with the other two positions.
- This should theoretically allow more tight ends to be more valuable throughout the year, and avoiding the usual top heavy position.

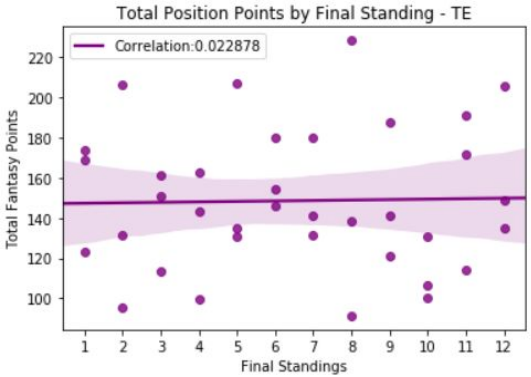


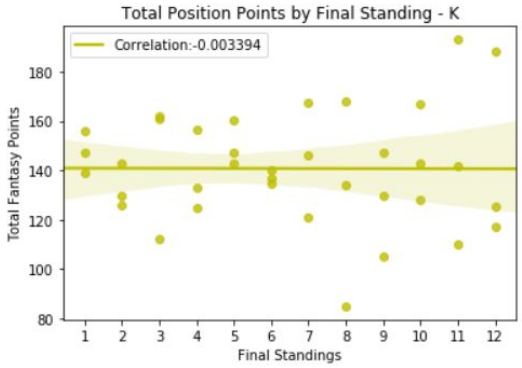
Findings - Position Correlation

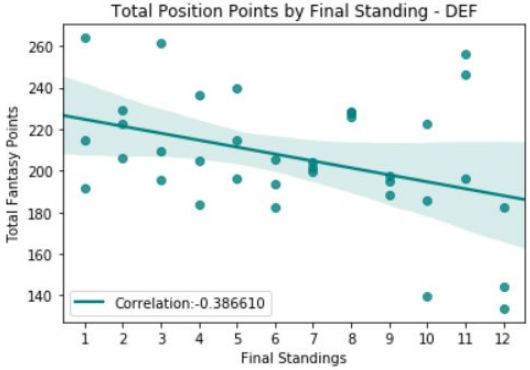












Findings - Position Correlation

- There was not a concrete negative correlation between total points by position and final standings.
- The closest negative correlation are quarterback (-.466309) and defense (-.386610)
- Looking at these graphs it would lead me to believe having the best of one position doesn’t guarantee a high finish in the standings.
- Quarterback has the best chance to give return a better spot in the final standings if a team has the highest scoring quarterback.
- It would make sense to target the better quarterbacks sooner in drafts in order to try and gain that advantage.

Limitations

- This is only three years worth of data.
- This is only one league of the hundreds of thousands of leagues in existence.

Conclusions - TE Point Change

- Tight ends could become more relevant of a position for fantasy football if they get more points per reception than running backs and wide receivers.
- For this study, each position gets .5 points per reception, based on my analysis running backs and wide receivers should remain at .5 points per reception and tight ends should be bumped up to 1 point per reception.
    - This would bring the median value of tight end scoring per week within 1 point of the other positions.

Conclusions - Position Correlation

- There is not a strong correlation between any position total points and where those teams ended up in the standings.
- The closest correlation is with quarterbacks and defense.
- With this information it would make more sense to prioritize quarterbacks and defenses sooner in drafts in order to try and capitalize on this slight correlation.

Acknowledgements

I collected this data myself off of Yahoo!. This data is from a fantasy football league I participate in each year. Thank you Alissa Clark for reviewing this analysis.

References

I did all the work on my own, I did reference stackoverflow.com for tips in python.

Research Question:

Is there any correlation to where a fantasy team ends up in the standings and the total points they scored at each position?

In [1]:
```
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
```

In [2]:
```
all_lineups = pd.read_csv('./Lineups.csv',sep = ',')
all_lineups = all_lineups.drop(['Team'],1)
lineups = all_lineups.rename(columns = {'POS':'Position'})
lineups = lineups[lineups.Starter == 1]
lineups.head()
```

Out[2]:

	Season	Week	Owner	Starter	Position	Name	Fan Pts	% Start	Pass Comp	Pass Yds
0	2016-2017	1	Whitmore	1	QB	Ben Roethlisberger	32.3	92%	27.0	300.0
1	2016-2017	1	Whitmore	1	RB	Lamar Miller	16.5	99%	0.0	0.0
2	2016-2017	1	Whitmore	1	RB	Rashad Jennings	10.1	34%	0.0	0.0
3	2016-2017	1	Whitmore	1	WR	Brandon Marshall	4.7	98%	0.0	0.0
4	2016-2017	1	Whitmore	1	WR	Jeremy Maclin	14.8	77%	0.0	0.0

5 rows × 39 columns

In [3]:
```
full_standings = pd.read_csv('./Standings.csv',sep = ',')
standings = full_standings[full_standings['Season'].isin(["2015-2016","
2016-2017","2017-2018"])]
standings = standings[['Owner','Final_Standings','Season']]
standings.head()
```

Out[3]:

	Owner	Final_Standings	Season
0	Stein	1	2017-2018
1	Walton	2	2017-2018
2	Ritzel	3	2017-2018
3	Clark	4	2017-2018
4	Whitmore	5	2017-2018

In [4]:
```
lineups.shape
```

Out[4]: (5181, 39)

In [5]:
```
lineups1 = lineups[['Season', 'Owner', 'Position', 'Fan Pts']]
lineups1.head()
```

Out[5]:

	Season	Owner	Position	Fan Pts
0	2016-2017	Whitmore	QB	32.3
1	2016-2017	Whitmore	RB	16.5
2	2016-2017	Whitmore	RB	10.1
3	2016-2017	Whitmore	WR	4.7
4	2016-2017	Whitmore	WR	14.8

In [6]:
```
summed_lineups = lineups1.groupby(['Season', 'Owner', 'Position']).sum(
)
summed_lineups=pd.DataFrame(summed_lineups.reset_index())
```

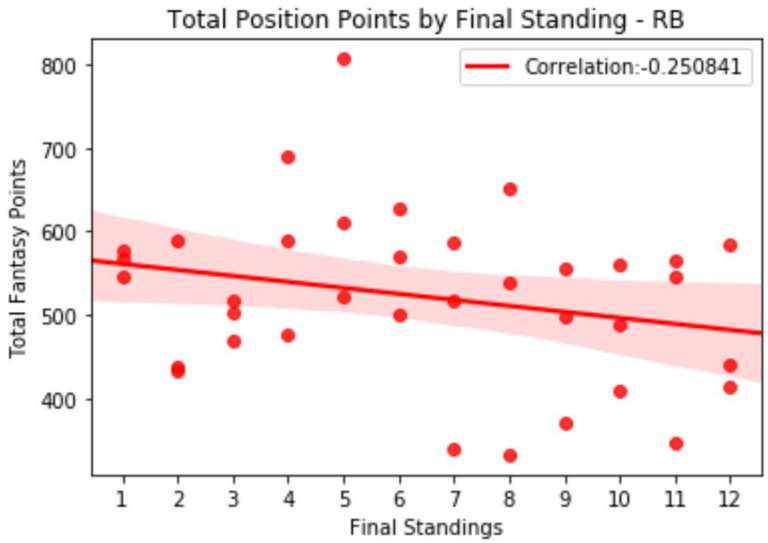
In [7]:
```
combined = pd.merge(standings, summed_lineups)
```

In [8]:
```
combined.head(6)
```

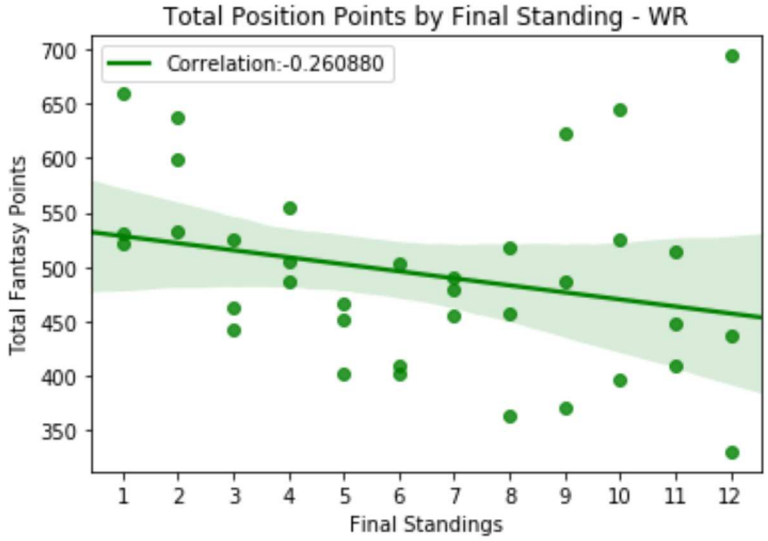
Out[8]:

	Owner	Final_Standings	Season	Position	Fan Pts
0	Stein	1	2017-2018	DEF	263.85
1	Stein	1	2017-2018	K	138.94
2	Stein	1	2017-2018	QB	482.91
3	Stein	1	2017-2018	RB	576.34
4	Stein	1	2017-2018	TE	122.92
5	Stein	1	2017-2018	WR	530.05

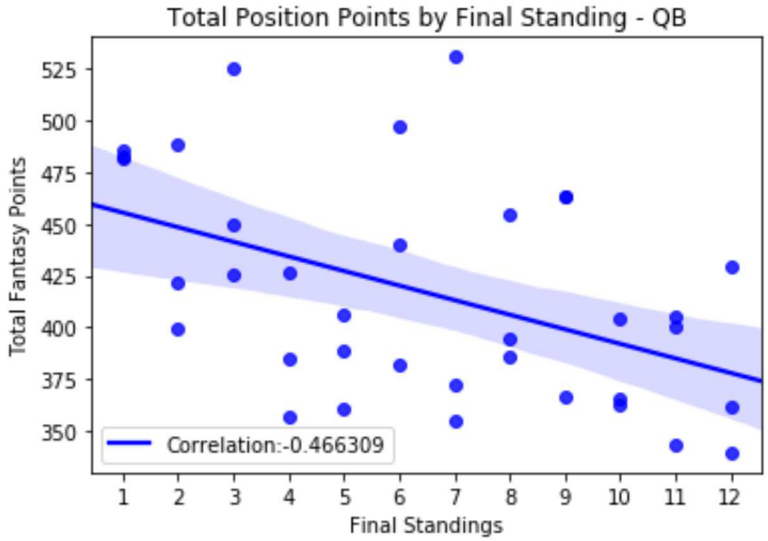
In [9]:
```
RB = combined[combined.Position=="RB"]
RB_matrix = RB.corr()
ax = sns.regplot(x = "Final_Standings", y="Fan Pts", data=RB, color='r'
)
ax.set(xlabel="Final Standings",xticks=np.arange(1,13,1),ylabel="Total
Fantasy Points",
       title="Total Position Points by Final Standing - RB")
plt.legend(["Correlation:{:f}".format(round(RB_matrix.iloc[0,1],6))])
plt.show()
```



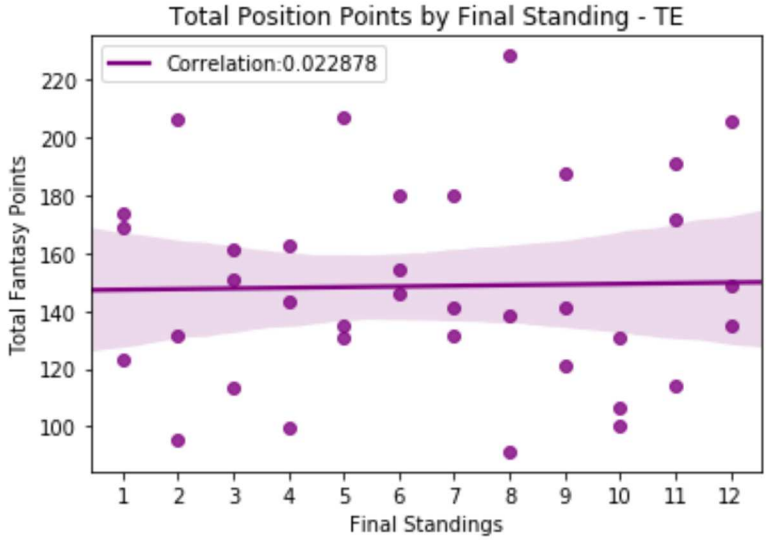
In [10]:
```
WR = combined[combined.Position=="WR"]
WR_matrix = WR.corr()
ax = sns.regplot(x = "Final_Standings", y="Fan Pts", data=WR, color='g'
)
ax.set(xlabel="Final Standings",xticks=np.arange(1,13,1),ylabel="Total
Fantasy Points",
       title="Total Position Points by Final Standing - WR")
plt.legend(["Correlation:{:f}".format(round(WR_matrix.iloc[0,1],6))])
plt.show()
```



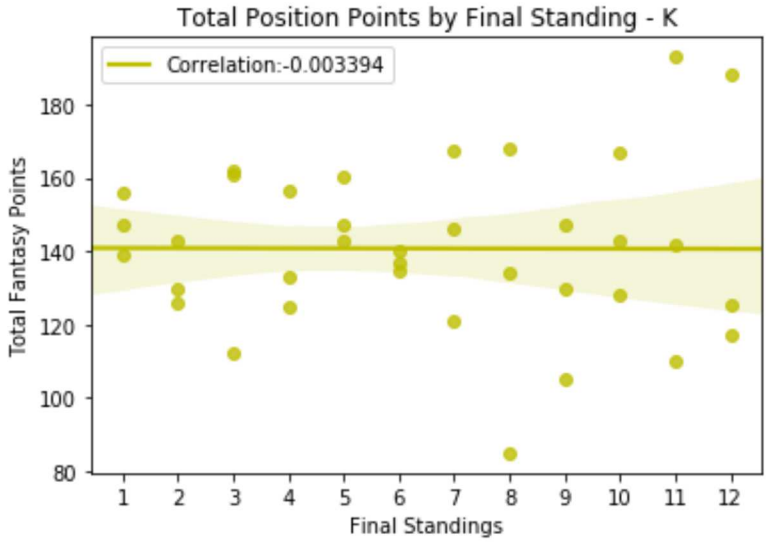
In [11]:
```
QB = combined[combined.Position=="QB"]
QB_matrix = QB.corr()
ax = sns.regplot(x = "Final_Standings", y="Fan Pts", data=QB, color='b'
)
ax.set(xlabel="Final Standings",xticks=np.arange(1,13,1),ylabel="Total
Fantasy Points",
       title="Total Position Points by Final Standing - QB")
plt.legend(["Correlation:{:f}".format(round(QB_matrix.iloc[0,1],6))], l
oc="lower left")
plt.show()
```



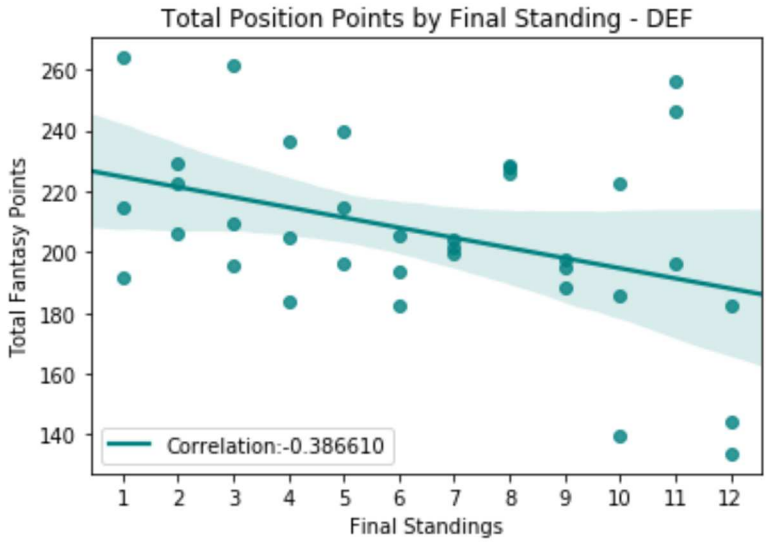
In [12]:
```
TE = combined[combined.Position=="TE"]
TE_matrix = TE.corr()
ax = sns.regplot(x = "Final_Standings", y="Fan Pts", data=TE, color='pu
rple')
ax.set(xlabel="Final Standings",xticks=np.arange(1,13,1),ylabel="Total
Fantasy Points",
       title="Total Position Points by Final Standing - TE")
plt.legend(["Correlation:{:f}".format(round(TE_matrix.iloc[0,1],6))])
plt.show()
```



In [13]:
```
K = combined[combined.Position=="K"]
K_matrix = K.corr()
ax = sns.regplot(x = "Final_Standings", y="Fan Pts", data=K, color='y')
ax.set(xlabel="Final Standings",xticks=np.arange(1,13,1),ylabel="Total
Fantasy Points",
       title="Total Position Points by Final Standing - K")
plt.legend(["Correlation:{:f}".format(round(K_matrix.iloc[0,1],6))])
plt.show()
```



In [14]:
```
DEF = combined[combined.Position=="DEF"]
DEF_matrix = DEF.corr()
ax = sns.regplot(x = "Final_Standings", y="Fan Pts", data=DEF, color='t
eal')
ax.set(xlabel="Final Standings",xticks=np.arange(1,13,1),ylabel="Total
Fantasy Points",
       title="Total Position Points by Final Standing - DEF")
plt.legend(["Correlation:{:f}".format(round(DEF_matrix.iloc[0,1],6))])
plt.show()
```



Research Question:

How can changing the points a tight end receives per catch effect how the position stacks up to running backs and wide receivers each week?

In [1]:
```
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
```

In [2]:
```
ls
```

```
 Volume in drive C is Windows
 Volume Serial Number is BEC0-9D55

 Directory of C:\Users\aclark5\Documents\Python\Final_Project

07/12/2018  11:01 AM    <DIR>          .
07/12/2018  11:01 AM    <DIR>          ..
07/12/2018  11:01 AM    <DIR>          .ipynb_checkpoints
07/12/2018  11:01 AM           111,053 Drafts.csv
07/12/2018  11:01 AM         1,082,671 Lineups.csv
07/12/2018  11:01 AM            84,993 Matchups.csv
07/12/2018  11:01 AM                28 POS.csv
07/12/2018  11:01 AM           148,889 Position Correlation.ipynb
07/12/2018  11:01 AM            14,540 Standings.csv
07/12/2018  11:01 AM            57,514 TE Change.ipynb
               7 File(s)      1,499,688 bytes
               3 Dir(s)  97,693,847,552 bytes free
```

In [3]:
```
all_lineups = pd.read_csv('./Lineups.csv',sep = ',')
all_lineups = all_lineups.drop(['Team'],1)
```

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

Out[24]:

	Season	Week	Owner	Starter	POS	Name	Fan Pts	% Start	Pass Comp	Pass Yds	...
0	2016-2017	1	Whitmore	1	QB	Ben Roethlisberger	32.3	92%	27.0	300.0	...
1	2016-2017	1	Whitmore	1	RB	Lamar Miller	16.5	99%	0.0	0.0	...
2	2016-2017	1	Whitmore	1	RB	Rashad Jennings	10.1	34%	0.0	0.0	...
3	2016-2017	1	Whitmore	1	WR	Brandon Marshall	4.7	98%	0.0	0.0	...
4	2016-2017	1	Whitmore	1	WR	Jeremy Maclin	14.8	77%	0.0	0.0	...

5 rows × 39 columns

In [5]:
```
all_lineups.shape
```

Out[5]: (9151, 39)

In [6]:
```
all_lineups.describe().transpose()
```

Out[6]:

	count	mean	std	min	25%	50%	75%	max
Week	9151.0	8.501147	4.611482	1.0	4.5	9.00	13.00	16.0
Starter	9151.0	0.566168	0.495630	0.0	0.0	1.00	1.00	1.0
Fan Pts	9151.0	10.704773	9.186325	0.0	3.7	9.05	15.16	64.2
Pass Comp	8730.0	2.305155	7.046716	0.0	0.0	0.00	0.00	44.0
Pass Yds	8730.0	26.882818	82.619868	0.0	0.0	0.00	0.00	513.0
Pass TD	8729.0	0.179746	0.655389	0.0	0.0	0.00	0.00	7.0
INT	8730.0	0.077892	0.368454	0.0	0.0	0.00	0.00	5.0
Rush ATT	8730.0	3.555326	6.257522	0.0	0.0	0.00	5.00	38.0
Rush Yds	8721.0	14.932003	29.072017	0.0	0.0	0.00	17.00	236.0
Rush TD	8712.0	0.103535	0.365642	0.0	0.0	0.00	0.00	3.0
Tgt	8707.0	3.559665	3.942671	0.0	0.0	2.00	6.00	33.0
Rec	8707.0	2.305386	2.652840	0.0	0.0	1.00	4.00	17.0
Rec Yds	8707.0	26.416676	35.716123	0.0	0.0	10.00	43.00	300.0
Rec TD	8707.0	0.161594	0.425691	0.0	0.0	0.00	0.00	3.0
Return Yds	8707.0	2.022970	11.387437	0.0	0.0	0.00	0.00	200.0
Return TD	8707.0	0.001608	0.040069	0.0	0.0	0.00	0.00	1.0
2PT	8563.0	0.017868	0.136815	0.0	0.0	0.00	0.00	2.0
Fum Lost	8563.0	0.054420	0.240843	0.0	0.0	0.00	0.00	3.0
0 19 Yrds	6478.0	0.001389	0.037251	0.0	0.0	0.00	0.00	1.0
20 29 Yrds	6478.0	0.043841	0.251458	0.0	0.0	0.00	0.00	3.0
30 39 Yrds	6478.0	0.048317	0.261194	0.0	0.0	0.00	0.00	3.0
40 49 Yrds	6478.0	0.047237	0.258720	0.0	0.0	0.00	0.00	4.0
50+ Yrds	6478.0	0.020222	0.153368	0.0	0.0	0.00	0.00	3.0
PAT Made	6478.0	0.209633	0.784806	0.0	0.0	0.00	0.00	7.0
PAT Miss	6478.0	0.018061	0.386767	0.0	0.0	0.00	0.00	23.0
Points VS	6539.0	2.211194	7.034105	0.0	0.0	0.00	0.00	51.0
Sack	6539.0	0.283377	2.082936	0.0	0.0	0.00	0.00	151.0
Safe	6539.0	0.004435	0.068716	0.0	0.0	0.00	0.00	2.0
Interception	6539.0	0.098027	0.433770	0.0	0.0	0.00	0.00	5.0
Fum Rec	6539.0	0.065148	0.320181	0.0	0.0	0.00	0.00	4.0
DEF TD	6539.0	0.020798	0.181414	0.0	0.0	0.00	0.00	8.0
Blk Kick	6539.0	0.015293	0.387253	0.0	0.0	0.00	0.00	30.0

In [7]:
```
lineups = all_lineups[all_lineups['POS'].isin(["RB","TE","WR"]) & all_l
ineups.Starter == 1]
```

In [16]:
```
lineups['Full_PPR'] = np.where(lineups['POS']=='TE',(lineups['Rec']*1)
+ (lineups['Rec Yds']*.1) + (lineups['Rec TD']*6),lineups['Fan Pts'])
lineups['Half_PPR'] = np.where(lineups['POS']=='TE',(lineups['Rec']*.5)
+ (lineups['Rec Yds']*.1) + (lineups['Rec TD']*6),lineups['Fan Pts'])
lineups['Eigth_PPR'] = np.where(lineups['POS']=='TE',(lineups['Rec']*.8
) + (lineups['Rec Yds']*.1) + (lineups['Rec TD']*6),lineups['Fan Pts'])
```

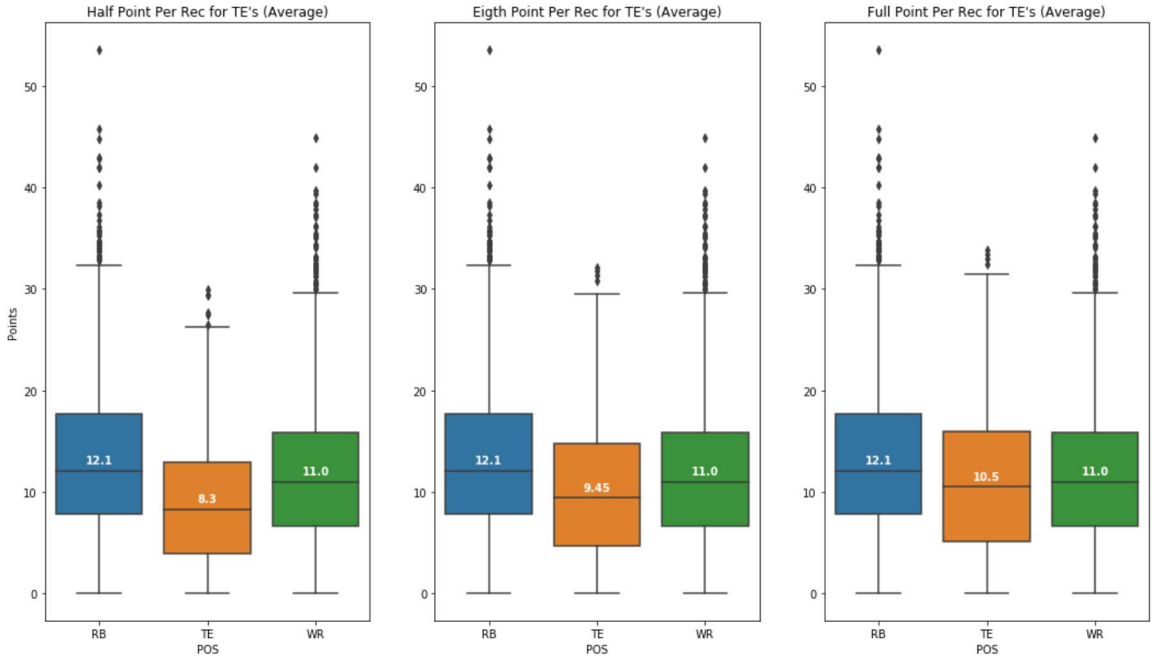
```
C:\Users\aclark5\AppData\Local\Continuum\anaconda3\lib\site-packages\
ipykernel_launcher.py:1: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: http://pandas.pydata.org/pandas
-docs/stable/indexing.html#indexing-view-versus-copy
  """Entry point for launching an IPython kernel.
C:\Users\aclark5\AppData\Local\Continuum\anaconda3\lib\site-packages\
ipykernel_launcher.py:2: 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/indexing.html#indexing-view-versus-copy

C:\Users\aclark5\AppData\Local\Continuum\anaconda3\lib\site-packages\
ipykernel_launcher.py:3: 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/indexing.html#indexing-view-versus-copy
  This is separate from the ipykernel package so we can avoid doing i
mports until
```

In [17]:
```
lineups_plt = lineups[['POS','Rec','Rec Yds','Rec TD','Full_PPR','Half_
PPR','Eigth_PPR']]
lineups_plt.head()
```

Out[17]:

	POS	Rec	Rec Yds	Rec TD	Full_PPR	Half_PPR	Eigth_PPR
1	RB	4.0	11.0	0.0	16.5	16.5	16.5
2	RB	1.0	3.0	0.0	10.1	10.1	10.1
3	WR	3.0	32.0	0.0	4.7	4.7	4.7
4	WR	5.0	63.0	1.0	14.8	14.8	14.8
5	TE	3.0	14.0	0.0	4.4	2.9	3.8

In [22]:
```
dim= (18,10)
fig, axs = plt.subplots(figsize=dim,ncols=3)

# Full point Plot

ax = sns.boxplot(x=lineups_plt['POS'], y=lineups_plt['Full_PPR'], order
=["RB","TE","WR"], ax=axs[2])
medians = lineups_plt.groupby(['POS'])['Full_PPR'].median().values
median_labels = [str(np.round(s, 2)) for s in medians]

pos = range(len(medians))
for tick,label in zip(pos,ax.get_xticklabels()):
    ax.text(pos[tick], medians[tick]+.7, median_labels[tick],
            horizontalalignment='center', size='medium', color='w', wei
ght='semibold')
ax.set_ylabel('')
ax.set_title('Full Point Per Rec for TE\'s (Average)')

# Eigth Point Plot

ax = sns.boxplot(x=lineups_plt['POS'], y=lineups_plt['Eigth_PPR'], orde
r=["RB","TE","WR"], ax=axs[1])
medians = lineups_plt.groupby(['POS'])['Eigth_PPR'].median().values
median_labels = [str(np.round(s, 2)) for s in medians]

pos = range(len(medians))
for tick,label in zip(pos,ax.get_xticklabels()):
    ax.text(pos[tick], medians[tick]+.7, median_labels[tick],
            horizontalalignment='center', size='medium', color='w', wei
ght='semibold')
ax.set_ylabel('')
ax.set_title('Eigth Point Per Rec for TE\'s (Average)')

# Half Point Plot

ax = sns.boxplot(x=lineups_plt['POS'], y=lineups_plt['Half_PPR'], order
=["RB","TE","WR"], ax=axs[0])
medians = lineups_plt.groupby(['POS'])['Half_PPR'].median().values
median_labels = [str(np.round(s, 2)) for s in medians]

pos = range(len(medians))
for tick,label in zip(pos,ax.get_xticklabels()):
    ax.text(pos[tick], medians[tick]+.7, median_labels[tick],
            horizontalalignment='center', size='medium', color='w', wei
ght='semibold')

ax.set_ylabel('Points')
ax.set_title('Half Point Per Rec for TE\'s (Average)')



plt.show()
```


Half Point Per Rec for TE's (Average)
Eigth Point Per Rec for TE's (Average)
Full Point Per Rec for TE's (Average)
Points
POS
RB
TE
WR
12.1
8.3
11.0
12.1
9.45
11.0
12.1
10.5
11.0