kmeans_player_positions

March 25, 2018

```
In [1]: import numpy
        import pandas

In [2]: from nba_py.player import *
        from nba_py.team import *

In [3]: PlayerList(season='2015-16').info().info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 103 entries, 0 to 102
Data columns (total 13 columns):
PERSON_ID                   103 non-null int64
DISPLAY_LAST_COMMA_FIRST    103 non-null object
DISPLAY_FIRST_LAST          103 non-null object
ROSTERSTATUS                103 non-null int64
FROM_YEAR                   103 non-null object
TO_YEAR                     103 non-null object
PLAYERCODE                  103 non-null object
TEAM_ID                     103 non-null int64
TEAM_CITY                   103 non-null object
TEAM_NAME                   103 non-null object
TEAM_ABBREVIATION           103 non-null object
TEAM_CODE                   103 non-null object
GAMES_PLAYED_FLAG           103 non-null object
dtypes: int64(3), object(10)
memory usage: 6.5+ KB
```

```
In [4]: last_season = "2007-08"
        warriors = "1610612744"
        cavs = "1610612739"
        thunder = "1610612760"
        raps = "1610612761"
        player = 2544
        first_game = "1610612744"
        adv = 'Advanced'
        permode = 'Per100Possessions'
```

```
In [5]: import time

In [6]: import matplotlib.pyplot as plt
        %matplotlib inline
        import seaborn as sns

In [7]: teams = TeamList().info().head(30)

In [8]: teams.head()
```

Out[8]:

	LEAGUE_ID	TEAM_ID	MIN_YEAR	MAX_YEAR	ABBREVIATION
0	00	1610612737	1949	2017	ATL
1	00	1610612738	1946	2017	BOS
2	00	1610612739	1970	2017	CLE
3	00	1610612740	2002	2017	NOP
4	00	1610612741	1966	2017	CHI

```
In [9]: team_ids = teams['TEAM_ID']

In [10]: rosters = []
         for team in teams['TEAM_ID']:
             rosters.append(TeamCommonRoster(team, season=last_season).roster().drop(['SCHOOL'


In [ ]: team_abrv = []
        abbrvs = teams['ABBREVIATION']

In [ ]: for i in range(0,30):
            team_abrv.append(abbrvs.iloc[i])

In [ ]: team_abrv[28]

In [ ]: def compute_stats(player_id, season):
            stats = PlayerGeneralSplits(player_id,measure_type= adv, per_mode = permode, seasor

In [15]: def save_team(roster):
             stats = roster['PLAYER_ID'].apply(lambda player: compute_stats(player, last_seasor
             pandas.concat([roster, stats], axis=1).to_csv('team.csv')

In [16]: rosters[29].head()
```

Out[16]:

	TeamID	SEASON	PLAYER	NUM	POSITION	HEIGHT	WEIGHT	AGE	EXP	\
0	1610612766	2007	Derek Anderson	1	G	6-5	195	33.0	10	
1	1610612766	2007	Gerald Wallace	3	F	6-7	220	25.0	6	
2	1610612766	2007	Jared Dudley	4	F	6-7	225	22.0	R	
3	1610612766	2007	Nazr Mohammed	6	C	6-10	250	30.0	9	
4	1610612766	2007	Earl Boykins	11	G	5-5	133	32.0	9	

	PLAYER_ID
0	1507
1	2222
2	201162
3	1737
4	1863

```
In [ ]: all_teams = []
        for i in range(0,30):
            time.sleep(5)
            stats = rosters[i]['PLAYER_ID'].apply(lambda player: compute_stats(player, last_sea
            all_teams.append(pandas.concat([rosters[i], stats], axis=1))

In [20]: totalPlayers.head(30)
```

Out[20]:

	AGE	AST	BLK	DREB	EXP	FG3A	FG3M	FG3_PCT	\
0	25.0	1.539474	0.565789	2.618421	3	0.789474	0.289474	0.192987	
1	27.0	5.780488	0.219512	3.463415	6	5.414634	2.060976	0.358744	
2	23.0	2.017857	0.000000	0.803571	R	0.607143	0.125000	0.083321	
3	22.0	3.358025	2.802469	6.246914	3	1.222222	0.308642	0.161506	
4	24.0	0.203125	0.062500	0.343750	R	0.062500	0.000000	0.000000	
5	30.0	6.041667	0.083333	3.000000	9	5.166667	1.937500	0.349000	
6	30.0	NaN	NaN	NaN	6	NaN	NaN	NaN	
7	22.0	1.530864	0.938272	6.629630	R	0.061728	0.000000	0.000000	
8	25.0	0.800000	0.142857	0.571429	2	2.428571	0.828571	0.250714	
9	22.0	1.712500	0.412500	4.225000	2	0.125000	0.012500	0.012500	
10	24.0	0.580645	0.209677	2.564516	4	0.048387	0.000000	0.000000	
11	24.0	0.074074	0.000000	0.296296	1	0.703704	0.259259	0.160481	
12	23.0	0.000000	0.142857	0.657143	1	0.057143	0.000000	0.000000	
0	24.0	0.267857	0.285714	2.357143	1	0.017857	0.000000	0.000000	
1	32.0	3.436620	1.253521	7.323944	12	0.154930	0.000000	0.000000	
2	22.0	5.103896	0.168831	3.168831	1	0.246753	0.064935	0.051948	
3	22.0	0.405797	0.289855	1.637681	R	0.000000	0.000000	0.000000	
4	22.0	0.866667	0.000000	0.466667	R	0.800000	0.200000	0.166667	
5	32.0	3.082192	0.219178	2.643836	11	6.191781	2.465753	0.395027	
6	38.0	3.890909	0.109091	2.163636	14	1.381818	0.418182	0.155455	
7	30.0	4.537500	0.450000	4.475000	9	4.562500	1.787500	0.354038	
8	31.0	1.540541	0.256757	3.945946	8	3.770270	1.432432	0.318919	
9	26.0	1.520000	0.280000	1.786667	3	0.760000	0.240000	0.126667	
10	23.0	1.076923	1.461538	4.217949	4	0.012821	0.000000	0.000000	
11	30.0	0.833333	0.166667	1.187500	6	0.958333	0.312500	0.179167	
12	30.0	1.948718	0.128205	1.897436	7	3.820513	1.500000	0.379936	
13	33.0	0.136364	0.272727	1.045455	10	0.000000	0.000000	0.000000	
14	38.0	0.555556	0.444444	2.166667	14	0.055556	0.000000	0.000000	
0	22.0	2.500000	0.241379	1.793103	1	4.620690	2.034483	0.413017	
1	24.0	1.607843	0.117647	1.823529	4	2.235294	0.666667	0.237608	

	FGA	FGM	...	POSITION	PTS	REB	SEASON	STL	\
0	7.539474	4.302632	...	G-F	11.815789	4.907895	2007	0.934211	
1	18.256098	7.890244	...	G	21.695122	4.475610	2007	1.024390	
2	4.232143	1.696429	...	G	4.196429	1.017857	2007	0.517857	
3	13.987654	6.395062	...	F	17.209877	8.234568	2007	1.518519	
4	0.765625	0.328125	...	G	0.921875	0.750000	2007	0.187500	
5	12.562500	5.166667	...	G	13.895833	3.333333	2007	1.125000	
6	NaN	NaN	...	G	NaN	NaN	2007	NaN	

7	8.246914	4.111111	...	C-F	10.135802	9.691358	2007	0.740741	
8	5.142857	1.857143	...	G	5.714286	0.685714	2007	0.200000	
9	11.475000	5.300000	...	F	14.812500	5.712500	2007	1.012500	
10	3.951613	1.725806	...	C	5.193548	4.000000	2007	0.387097	
11	1.629630	0.592593	...	G-F	1.518519	0.333333	2007	0.111111	
12	0.857143	0.342857	...	F	1.000000	1.200000	2007	0.085714	
0	4.803571	2.750000	...	F	7.946429	4.053571	2007	0.267857	
1	13.943662	7.521127	...	F	18.830986	9.225352	2007	1.408451	
2	9.259740	4.558442	...	G	10.571429	4.181818	2007	1.675325	
3	3.202899	1.550725	...	F	4.536232	3.014493	2007	0.449275	
4	2.600000	0.933333	...	G	2.133333	0.533333	2007	0.333333	
5	13.506849	6.013699	...	G	17.438356	3.671233	2007	0.890411	
6	9.672727	4.236364	...	G	11.200000	2.472727	2007	0.672727	
7	13.725000	6.362500	...	F	19.625000	5.137500	2007	1.262500	
8	5.594595	2.337838	...	F	7.364865	4.351351	2007	0.972973	
9	5.186667	2.253333	...	G	6.586667	2.240000	2007	0.826667	
10	4.461538	2.743590	...	C	6.948718	6.076923	2007	0.397436	
11	1.958333	0.604167	...	F-C	1.833333	1.645833	2007	0.187500	
12	6.807692	2.782051	...	G	7.487179	2.141026	2007	0.756410	
13	1.045455	0.545455	...	C-F	1.772727	1.681818	2007	0.136364	
14	2.277778	0.777778	...	F-C	2.166667	3.777778	2007	0.277778	
0	8.017241	3.465517	...	G	10.431034	2.310345	2007	0.810345	
1	7.901961	2.862745	...	G-F	7.431373	2.509804	2007	0.568627	

	TOV	TeamID	WEIGHT	WINS	WL
0	1.289474	1610612737	210	33.0	NaN
1	2.719512	1610612737	235	37.0	NaN
2	1.000000	1610612737	195	25.0	NaN
3	3.024691	1610612737	235	37.0	NaN
4	0.140625	1610612737	210	31.0	NaN
5	2.479167	1610612737	190	24.0	NaN
6	NaN	1610612737	170	0.0	NaN
7	1.691358	1610612737	245	37.0	NaN
8	0.600000	1610612737	175	12.0	NaN
9	1.587500	1610612737	230	35.0	NaN
10	1.112903	1610612737	280	28.0	NaN
11	0.222222	1610612737	195	12.0	NaN
12	0.314286	1610612737	230	13.0	NaN
0	0.767857	1610612738	240	46.0	NaN
1	1.943662	1610612738	220	57.0	NaN
2	1.909091	1610612738	171	62.0	NaN
3	0.942029	1610612738	289	56.0	NaN
4	0.333333	1610612738	170	13.0	NaN
5	1.739726	1610612738	205	58.0	NaN
6	1.763636	1610612738	185	29.0	NaN
7	2.762500	1610612738	235	64.0	NaN
8	0.878378	1610612738	217	60.0	NaN
9	1.453333	1610612738	213	61.0	NaN

```
10  1.602564  1610612738     264  62.0 NaN
11  0.541667  1610612738     235  40.0 NaN
12  0.974359  1610612738     175  62.0 NaN
13  0.227273  1610612738     278  17.0 NaN
14  0.555556  1610612738     239  16.0 NaN
0   1.310345  1610612739     194  32.0 NaN
1   1.078431  1610612739     239  28.0 NaN

[30 rows x 34 columns]
```

```
In [19]: totalPlayers = pandas.concat(all_teams)
```

```
In [21]: totalPlayers.to_csv('players_0708.csv')
```

```
In [22]: significantPlayers = totalPlayers[totalPlayers['MIN'] > 10.0]
```

```
In [23]: significantPlayers = significantPlayers[significantPlayers['GP'] > 40]
```

```
In [24]: significantPlayers['FG3_PCT_ADJ'] = significantPlayers['FG3_PCT'].apply(lambda x: x i
```

```
In [25]: significantPlayers.to_csv('sig_players_0708.csv')
```

```
In [26]: all_players = []
         players_0708 = pandas.read_csv('players_0708.csv')
         all_players.append(players_0708)

         players_0809 = pandas.read_csv('players_0809.csv')
         all_players.append(players_0809)

         players_0910 = pandas.read_csv('players_0910.csv')
         all_players.append(players_0910)

         players_1011 = pandas.read_csv('players_1011.csv')
         all_players.append(players_1011)

         players_1112 = pandas.read_csv('players_1112.csv')
         all_players.append(players_1112)

         players_1213 = pandas.read_csv('players_1213.csv')
         all_players.append(players_1213)

         players_1314 = pandas.read_csv('players_1314.csv')
         all_players.append(players_1314)

         players_1415 = pandas.read_csv('players_1415.csv')
         all_players.append(players_1415)

         players_1516 = pandas.read_csv('players_1516.csv')
         all_players.append(players_1516)
```

```
        players_1617 = pandas.read_csv('players_1617.csv')
        all_players.append(players_1617)

In [27]: all_players_df = pandas.concat(all_players)

In [13]: all_players_df = pandas.read_csv('all_players_0717')

In [14]: all_players_df.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 4422 entries, 0 to 4421
Data columns (total 37 columns):
Unnamed: 0        4422 non-null int64
AGE               4422 non-null float64
AST               4345 non-null float64
BLK               4345 non-null float64
DREB              4345 non-null float64
EXP               4422 non-null object
FG3A              4345 non-null float64
FG3M              4345 non-null float64
FG3_PCT           4345 non-null float64
FGA               4345 non-null float64
FGM               4345 non-null float64
FG_PCT            4345 non-null float64
FTA               4345 non-null float64
FTM               4345 non-null float64
FT_PCT            4345 non-null float64
GP                4422 non-null float64
HEIGHT            4422 non-null object
LOSES             4422 non-null float64
MIN               4345 non-null float64
NUM               4411 non-null float64
OREB              4345 non-null float64
PF                4345 non-null float64
PLAYER            4422 non-null object
PLAYER_ID         4422 non-null int64
PLUS_MINUS        4345 non-null float64
POSITION          4422 non-null object
PTS               4345 non-null float64
REB               4345 non-null float64
SEASON            4422 non-null int64
STL               4345 non-null float64
TOV               4345 non-null float64
TeamID            4422 non-null int64
Unnamed: 0.1      4422 non-null int64
Unnamed: 0.1.1    451 non-null float64
WEIGHT            4422 non-null int64
WINS              4422 non-null float64
```

```
WL                0 non-null float64
dtypes: float64(27), int64(6), object(4)
memory usage: 1.2+ MB
```

```
In [15]: columns = ['Unnamed: 0', 'Unnamed: 0.1', 'WL']
         all_players_df.drop(columns, inplace=True, axis=1)
```

```
In [16]: all_players_df = all_players_df.dropna()
         all_players_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 446 entries, 3971 to 4421
Data columns (total 34 columns):
AGE               446 non-null float64
AST               446 non-null float64
BLK               446 non-null float64
DREB              446 non-null float64
EXP               446 non-null object
FG3A              446 non-null float64
FG3M              446 non-null float64
FG3_PCT           446 non-null float64
FGA               446 non-null float64
FGM               446 non-null float64
FG_PCT            446 non-null float64
FTA               446 non-null float64
FTM               446 non-null float64
FT_PCT            446 non-null float64
GP                446 non-null float64
HEIGHT            446 non-null object
LOSES             446 non-null float64
MIN               446 non-null float64
NUM               446 non-null float64
OREB              446 non-null float64
PF                446 non-null float64
PLAYER            446 non-null object
PLAYER_ID         446 non-null int64
PLUS_MINUS        446 non-null float64
POSITION          446 non-null object
PTS               446 non-null float64
REB               446 non-null float64
SEASON            446 non-null int64
STL               446 non-null float64
TOV               446 non-null float64
TeamID            446 non-null int64
Unnamed: 0.1.1    446 non-null float64
WEIGHT            446 non-null int64
WINS              446 non-null float64
```

```
dtypes: float64(26), int64(4), object(4)
memory usage: 115.0+ KB
```

```
In [17]: sigplayers = all_players_df[all_players_df['MIN'] > 25.0]
```

```
In [18]: sigplayers = all_players_df[all_players_df['GP'] > 30]
```

```
In [19]: sigplayers.to_csv('sig_players_0717')
```

```
In [20]: sigplayers.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 375 entries, 3971 to 4421
Data columns (total 34 columns):
AGE               375 non-null float64
AST               375 non-null float64
BLK               375 non-null float64
DREB              375 non-null float64
EXP               375 non-null object
FG3A              375 non-null float64
FG3M              375 non-null float64
FG3_PCT           375 non-null float64
FGA               375 non-null float64
FGM               375 non-null float64
FG_PCT            375 non-null float64
FTA               375 non-null float64
FTM               375 non-null float64
FT_PCT            375 non-null float64
GP                375 non-null float64
HEIGHT            375 non-null object
LOSES             375 non-null float64
MIN               375 non-null float64
NUM               375 non-null float64
OREB              375 non-null float64
PF                375 non-null float64
PLAYER            375 non-null object
PLAYER_ID         375 non-null int64
PLUS_MINUS        375 non-null float64
POSITION          375 non-null object
PTS               375 non-null float64
REB               375 non-null float64
SEASON            375 non-null int64
STL               375 non-null float64
TOV               375 non-null float64
TeamID            375 non-null int64
Unnamed: 0.1.1    375 non-null float64
WEIGHT            375 non-null int64
WINS              375 non-null float64
```

```
dtypes: float64(26), int64(4), object(4)
memory usage: 96.7+ KB
```

```
In [21]: sigplayers['AST_PER_MIN'] = sigplayers['AST']/sigplayers['MIN']
```

```
C:\Users\tkauk\Anaconda3\lib\site-packages\ipykernel\__main__.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
  if __name__ == '__main__':
```

```
In [22]: sigplayers['BLK_PER_MIN'] = sigplayers['BLK']/sigplayers['MIN']
```

```
C:\Users\tkauk\Anaconda3\lib\site-packages\ipykernel\__main__.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
  if __name__ == '__main__':
```

```
In [23]: sigplayers['PTS_PER_MIN'] = sigplayers['PTS']/sigplayers['MIN']
```

```
C:\Users\tkauk\Anaconda3\lib\site-packages\ipykernel\__main__.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
  if __name__ == '__main__':
```

```
In [24]: sigplayers['STL_PER_MIN'] = sigplayers['STL']/sigplayers['MIN']
```

```
C:\Users\tkauk\Anaconda3\lib\site-packages\ipykernel\__main__.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
  if __name__ == '__main__':
```

```
In [25]: sigplayers['TOV_PER_MIN'] = sigplayers['TOV']/sigplayers['MIN']
```

```
C:\Users\tkauk\Anaconda3\lib\site-packages\ipykernel\__main__.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
  if __name__ == '__main__':
```

```
In [26]: sigplayers['FGA_PER_MIN'] = sigplayers['FGA']/sigplayers['MIN']
```

```
C:\Users\tkauk\Anaconda3\lib\site-packages\ipykernel\__main__.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
  if __name__ == '__main__':
```

```
In [27]: sigplayers['FG3A_PER_MIN'] = sigplayers['FG3A']/sigplayers['MIN']
```

```
C:\Users\tkauk\Anaconda3\lib\site-packages\ipykernel\__main__.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
  if __name__ == '__main__':
```

```
In [28]: sigplayers['FTA_PER_MIN'] = sigplayers['FTA']/sigplayers['MIN']
```

```
C:\Users\tkauk\Anaconda3\lib\site-packages\ipykernel\__main__.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
  if __name__ == '__main__':
```

```
In [29]: sigplayers['REB_PER_MIN'] = sigplayers['REB']/sigplayers['MIN']
```

```
C:\Users\tkauk\Anaconda3\lib\site-packages\ipykernel\__main__.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
  if __name__ == '__main__':
```

```
In [30]: bigmen_list = []
         bigmen_list.append(sigplayers[sigplayers['POSITION'] == 'F'])
         bigmen_list.append(sigplayers[sigplayers['POSITION'] == 'F-C'])
         bigmen_list.append(sigplayers[sigplayers['POSITION'] == 'C-F'])
         bigmen_list.append(sigplayers[sigplayers['POSITION'] == 'C'])

         sig_bigmen = pandas.concat(bigmen_list)
```

```
In [31]: sig_guards.info()
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-31-84e55bc4b38e> in <module>()
----> 1 sig_guards.info()

NameError: name 'sig_guards' is not defined
```

```
In [301]: sig_guards['TALL'] = sig_guards['HEIGHT'].apply(lambda x:True if (x.split('-')[0] ==
```

```
In [ ]:
```

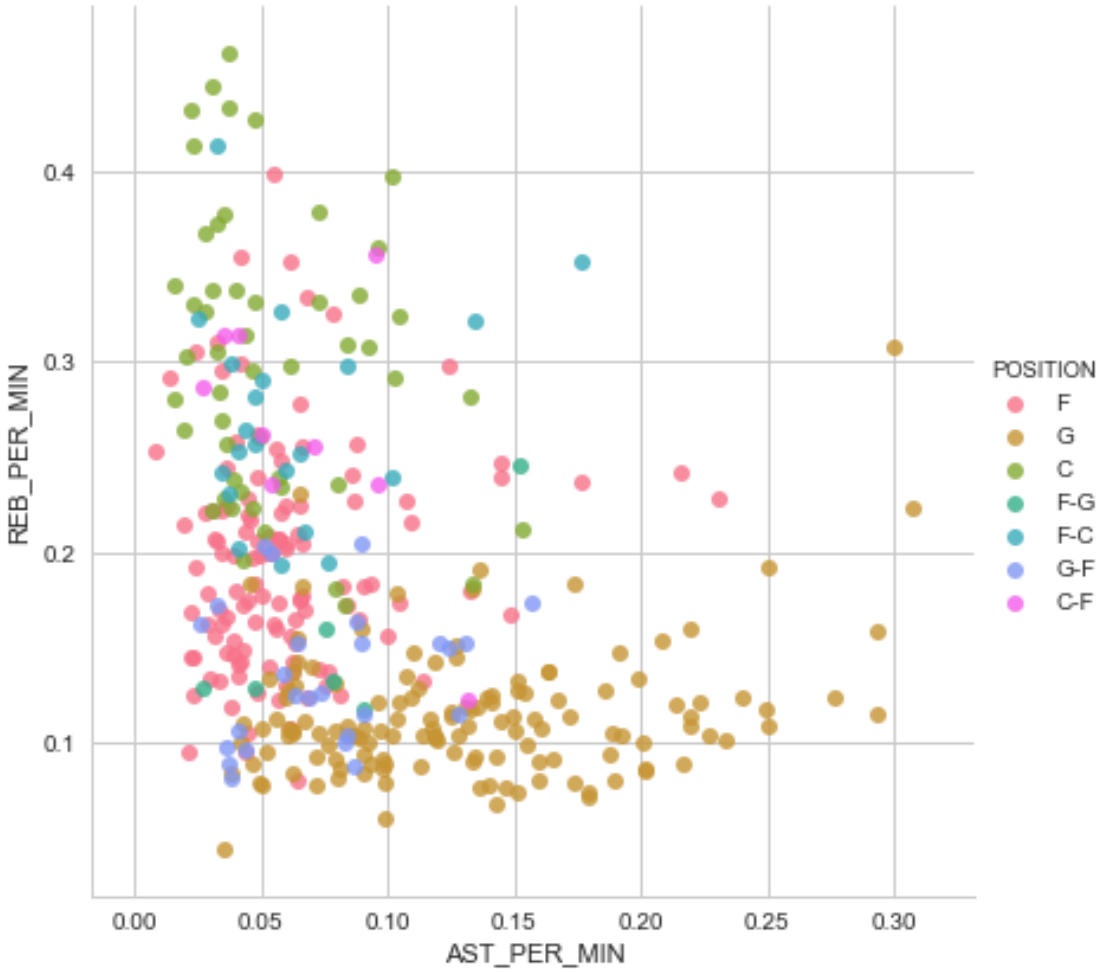
```
In [185]: sigplayers.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 2327 entries, 0 to 450
Data columns (total 41 columns):
AGE             2327 non-null float64
AST             2327 non-null float64
BLK             2327 non-null float64
DREB            2327 non-null float64
EXP             2327 non-null object
FG3A            2327 non-null float64
FG3M            2327 non-null float64
FG3_PCT         2327 non-null float64
FGA             2327 non-null float64
FGM             2327 non-null float64
FG_PCT          2327 non-null float64
FTA             2327 non-null float64
FTM             2327 non-null float64
FT_PCT          2327 non-null float64
GP              2327 non-null float64
HEIGHT          2327 non-null object
LOSES           2327 non-null float64
MIN             2327 non-null float64
NUM             2327 non-null float64
OREB            2327 non-null float64
PF              2327 non-null float64
PLAYER          2327 non-null object
PLAYER_ID       2327 non-null int64
PLUS_MINUS      2327 non-null float64
POSITION        2327 non-null object
PTS             2327 non-null float64
```

```
REB             2327 non-null float64
SEASON          2327 non-null int64
STL             2327 non-null float64
TOV             2327 non-null float64
TeamID          2327 non-null int64
WEIGHT          2327 non-null int64
WINS            2327 non-null float64
BLK_PER_MIN     2327 non-null float64
PTS_PER_MIN     2327 non-null float64
STL_PER_MIN     2327 non-null float64
TOV_PER_MIN     2327 non-null float64
FGA_PER_MIN     2327 non-null float64
FG3A_PER_MIN    2327 non-null float64
FTA_PER_MIN     2327 non-null float64
REB_PER_MIN     2327 non-null float64
dtypes: float64(33), int64(4), object(4)
memory usage: 727.2+ KB
```

```
In [32]: sns.set_style('whitegrid')
         sns.lmplot('AST_PER_MIN','REB_PER_MIN',data=sigplayers, hue='POSITION',
                    size=6,aspect=1,fit_reg=False)
```

```
Out[32]: <seaborn.axisgrid.FacetGrid at 0xd086df0>
```




```
In [33]: sigplayers['POSITION'] = sigplayers['POSITION'].apply(lambda x: 'G' if (x == 'G-F') el

C:\Users\tkauk\Anaconda3\lib\site-packages\ipykernel\__main__.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
  if __name__ == '__main__':
```

```
In [34]: sigplayers['POSITION'] = sigplayers['POSITION'].apply(lambda x: 'C' if (x == 'C-F') el

C:\Users\tkauk\Anaconda3\lib\site-packages\ipykernel\__main__.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
```

```
  if __name__ == '__main__':
```

```
In [35]: from sklearn.cluster import KMeans
```

```
In [36]: sigplayers['POSITION'] = sigplayers['POSITION'].apply(lambda x: 'F' if (x == 'F-C') el
```

```
C:\Users\tkauk\Anaconda3\lib\site-packages\ipykernel\__main__.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
  if __name__ == '__main__':
```

```
In [37]: sigplayers['POSITION'] = sigplayers['POSITION'].apply(lambda x: 'F' if (x == 'F-G') el
```

```
C:\Users\tkauk\Anaconda3\lib\site-packages\ipykernel\__main__.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
  if __name__ == '__main__':
```

```
In [39]: kmeans = KMeans(n_clusters=5)
```

```
In [40]: cols = ['LOSES','PLAYER','PLAYER_ID','POSITION','SEASON','TeamID','WINS','HEIGHT','EXI
         sig_players= sigplayers.drop(labels = cols, axis = 1)
```

```
In [42]: import pandas as pd
         from sklearn import preprocessing


         min_max_scaler = preprocessing.MinMaxScaler()
         np_scaled = min_max_scaler.fit_transform(sig_players)
         df_normalized = pd.DataFrame(np_scaled)
```

```
In [43]: kmeans.fit(sig_players)
```

```
Out[43]: KMeans(algorithm='auto', copy_x=True, init='k-means++', max_iter=300,
             n_clusters=5, n_init=10, n_jobs=1, precompute_distances='auto',
             random_state=None, tol=0.0001, verbose=0)
```

```
In [44]: kmeans.cluster_centers_
```

```
Out[44]: array([[  3.92414691e+00,   6.02231848e-01,   4.64404260e+00,
                   4.52686263e+00,   1.67601452e+00,   3.06325902e-01,
                   1.44800697e+01,   6.72383958e+00,   4.62834178e-01,
                   4.52289746e+00,   3.63162459e+00,   6.71973706e-01,
```

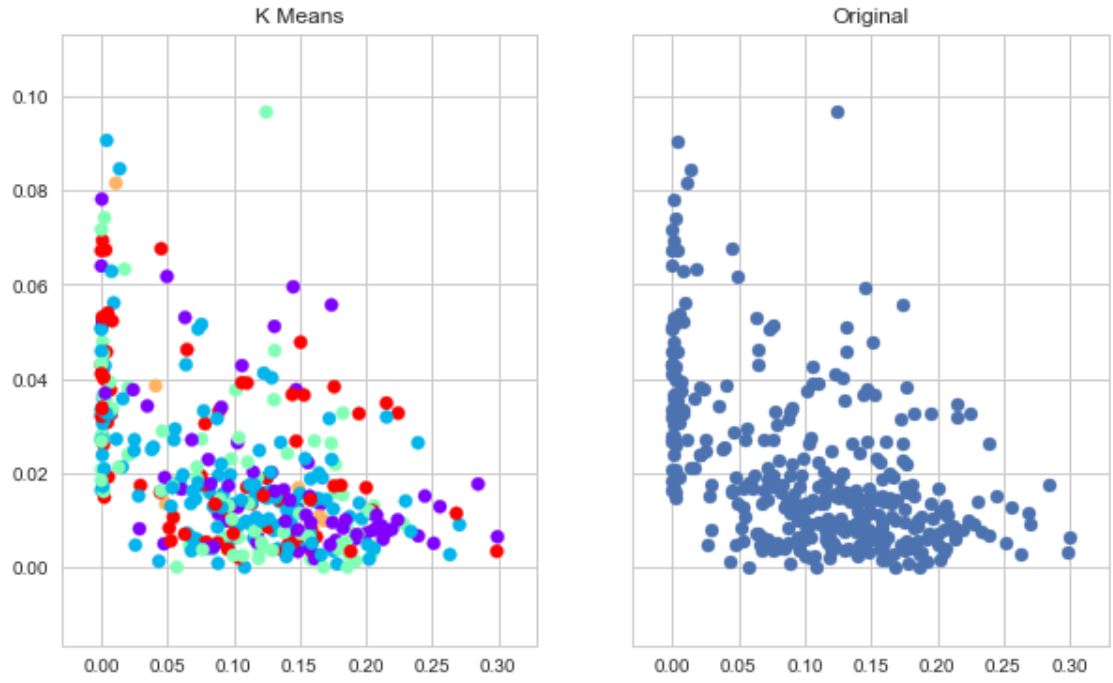
```
   7.25789474e+01,   3.27235824e+01,   1.02105263e+01,
   1.25914929e+00,   2.26816438e+00,   1.48351173e+00,
   1.87553183e+01,   5.90341119e+00,   1.11032939e+00,
   2.18429860e+00,   4.92105263e+00,   1.18599943e-01,
   1.84572159e-02,   5.69091677e-01,   3.38475535e-02,
   6.62496679e-02,   4.40422728e-01,   1.38074737e-01,
   1.36292147e-01,   1.80775488e-01],
[  1.86670393e+00,   4.10293858e-01,   2.89962105e+00,
   2.19243237e+00,   7.74245570e-01,   2.30571040e-01,
   6.68523868e+00,   3.02018139e+00,   4.31204880e-01,
   1.55903030e+00,   1.15678989e+00,   3.81314252e-01,
   7.39925926e+01,   2.20328977e+01,   1.01111111e+01,
   9.25956763e-01,   1.91367131e+00,  -2.20840772e-01,
   7.97139824e+00,   3.82557781e+00,   6.99889250e-01,
   1.08484716e+00,   5.28888889e+00,   8.38280471e-02,
   1.90940325e-02,   3.62663601e-01,   3.16425291e-02,
   4.94238333e-02,   3.04514844e-01,   9.93068775e-02,
   7.08509548e-02,   1.74955331e-01],
[  1.37384702e+00,   3.16662918e-01,   2.09265004e+00,
   1.60482154e+00,   5.53813063e-01,   1.79107772e-01,
   5.20527930e+00,   2.27791925e+00,   3.78595477e-01,
   1.35250186e+00,   9.95968582e-01,   3.14165324e-01,
   4.40243902e+01,   1.59177262e+01,   1.12073171e+01,
   6.78669366e-01,   1.48586814e+00,  -9.85314030e-01,
   6.10562014e+00,   2.77131941e+00,   5.05059390e-01,
   8.98948461e-01,   5.48780488e+00,   8.26401028e-02,
   2.00796908e-02,   3.69777896e-01,   3.18497580e-02,
   5.45431640e-02,   3.19729989e-01,   9.97823131e-02,
   8.05892867e-02,   1.71866440e-01],
[  1.22950750e+00,   4.77514029e-01,   2.68848124e+00,
   1.93929970e+00,   6.74869658e-01,   1.96325761e-01,
   5.09202616e+00,   2.44175024e+00,   4.33956384e-01,
   1.26309983e+00,   9.10006155e-01,   3.08177705e-01,
   6.42857143e+01,   1.83954979e+01,   9.02857143e+01,
   9.35050741e-01,   1.79235160e+00,   1.11887501e+00,
   6.46837630e+00,   3.62353198e+00,   5.96170567e-01,
   8.22767656e-01,   1.38571429e+01,   6.75533991e-02,
   2.53120708e-02,   3.41860662e-01,   3.13929986e-02,
   4.63047416e-02,   2.78462645e-01,   9.75265903e-02,
   6.45653502e-02,   1.96051708e-01],
[  1.58002569e+00,   4.78943803e-01,   3.08721228e+00,
   2.11264374e+00,   7.45652866e-01,   2.08563374e-01,
   6.90550750e+00,   3.20142159e+00,   4.32362016e-01,
   1.72367467e+00,   1.32941238e+00,   3.95593501e-01,
   6.42800000e+01,   2.08620014e+01,   3.66400000e+01,
   9.44979976e-01,   1.81732435e+00,  -3.40198060e-01,
   8.47790843e+00,   4.03219225e+00,   6.04774035e-01,
   1.01344228e+00,   1.22933333e+01,   6.98794646e-02,
```

```
                2.41422895e-02,   3.89546426e-01,   2.86476884e-02,
                4.76066596e-02,   3.18802318e-01,   9.55142019e-02,
                7.91335998e-02,   1.99068444e-01]])
```

In [46]:
```
labels = kmeans.predict(sig_players)
```

In [47]:
```
centroids = kmeans.cluster_centers_
```

In [51]:
```
f, (ax1, ax2) = plt.subplots(1, 2, sharey=True,figsize=(10,6))
ax1.set_title('K Means')
ax1.scatter(sigplayers['FG3A_PER_MIN'],sigplayers['BLK_PER_MIN'],c=kmeans.labels_,cma
ax2.set_title("Original")
ax2.scatter(sigplayers['FG3A_PER_MIN'],sigplayers['BLK_PER_MIN'],cmap=sigplayers['POS
plt.savefig('K_means_graph.png')
```



In [52]:
```
sigplayers['REAL_POSITION'] = labels
```

```
C:\Users\tkauk\Anaconda3\lib\site-packages\ipykernel\__main__.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
  if __name__ == '__main__':
```

In [54]:
```
sigplayers['REAL_POSITION'].value_counts()
```

```
Out[54]: 1    135
         2     82
         0     76
         4     75
         3      7
         Name: REAL_POSITION, dtype: int64

In [56]: sig_players_name = sigplayers[['PLAYER','POSITION','REAL_POSITION']]

In [298]: sig_bigmen_name.head(100)

Out[298]:                 PLAYER TRUE_POS  REAL_POSITION
          3           Josh Smith        F              1
          9      Marvin Williams        F              1
          12       Solomon Jones        C              0
          13           Leon Powe        C              0
          14       Kevin Garnett        F              1
          16          Glen Davis        F              1
          20         Paul Pierce        F              1
          21         James Posey        F              1
          31    Wally Szczerbiak        F              1
          38        LeBron James        F              1
          50     Peja Stojakovic        F              1
          52          David West        F              1
          53       Julian Wright        C              0
          55          Ryan Bowen        C              0
          58      Andres Nocioni        F              1
          61           Luol Deng        F              1
          67        Tyrus Thomas        F              1
          71         Drew Gooden        C              0
          80         Malik Allen        F              1
          82        Brandon Bass        F              1
          84       Dirk Nowitzki        F              1
          86        Juwan Howard        C              0
          90       Kenyon Martin        F              1
          92         Bobby Jones        C              0
          94     Carmelo Anthony        F              1
          95      Eduardo Najera        F              1
          101    Stephen Jackson        F              1
          102    Mickael Pietrus        F              1
          110        Matt Barnes        F              1
          113     Brandan Wright        C              0
          ..                ...      ...            ...
          272       Reggie Evans        F              1
          296      Raef LaFrentz        C              0
          301      Travis Outlaw        F              1
          302         James Jones        C              0
          318  Metta World Peace        C              0
```

322	Ime Udoka	F	1
326	Bruce Bowen	F	1
337	Mickael Gelabale	C	0
341	Jeff Green	F	1
348	Chris Wilcox	C	0
350	Chris Bosh	F	1
356	Joey Graham	C	0
359	Jason Kapono	F	1
360	Jamario Moon	F	1
361	Kris Humphries	C	0
363	Carlos Boozer	F	1
367	Matt Harpring	F	1
375	Andrei Kirilenko	F	1
379	Hakim Warrick	F	1
380	Rudy Gay	F	1
385	Brian Cardinal	C	0
386	Andre Brown	C	0
391	Caron Butler	F	1
392	Antawn Jamison	F	1
393	Dominic McGuire	F	1
396	Darius Songaila	F	1
398	Andray Blatche	F	1
403	Walter Herrmann	C	0
405	Jarvis Hayes	F	1
407	Tayshaun Prince	F	1

[100 rows x 3 columns]