NBA Longevity Analysis

One of the most important things an NBA front office does is predict player ability. In this notebook, I will attempt to determine which factors influence a basketball player's chances of a long NBA career using data available at the end of each player's first season. I will explore common factors that players with long NBA careers share, and explore available data to test whether those observed factors make a difference when extrapolated over a large set of players.

The dataset I'll be working with includes an interesting amount of detail about National Basketball Association (NBA) players with careers spanning from 1996 to 2021 downloaded from kaggle, compiled by the user Justinas Cirtautas.

In [27]:
```
import pandas as pd
from bokeh.io import show
import holoviews as hv
import numpy as np
hv.extension('bokeh')


df_nba = pd.read_csv('all_seasons.csv')
display(df_nba.head())
```

	Unnamed: 0	player_name	team_abbreviation	age	player_height	player_weight	college	country	draft_year	draft_round	...	pts
0	0	Dennis Rodman	CHI	36.0	198.12	99.790240	Southeastern Oklahoma State	USA	1986	2	...	5.7
1	1	Dwayne Schintzius	LAC	28.0	215.90	117.933920	Florida	USA	1990	1	...	2.3
2	2	Earl Cureton	TOR	39.0	205.74	95.254320	Detroit Mercy	USA	1979	3	...	0.8
3	3	Ed O'Bannon	DAL	24.0	203.20	100.697424	UCLA	USA	1995	1	...	3.7
4	4	Ed Pinckney	MIA	34.0	205.74	108.862080	Villanova	USA	1985	1	...	2.4

5 rows × 22 columns

Interesting! Looks like we have 22 columns defining data about individual players that made NBA rosters. The first looks to be a index column, let's redefine df_nba to use that as the index column rather than creating our own upon import.

In [28]:
```
df_nba = pd.read_csv('all_seasons.csv', index_col = 0)
display(df_nba.head())
```

	player_name	team_abbreviation	age	player_height	player_weight	college	country	draft_year	draft_round	draft_number	...	pts
0	Dennis Rodman	CHI	36.0	198.12	99.790240	Southeastern Oklahoma State	USA	1986	2	27	...	5.7
1	Dwayne Schintzius	LAC	28.0	215.90	117.933920	Florida	USA	1990	1	24	...	2.3
2	Earl Cureton	TOR	39.0	205.74	95.254320	Detroit Mercy	USA	1979	3	58	...	0.8
3	Ed O'Bannon	DAL	24.0	203.20	100.697424	UCLA	USA	1995	1	9	...	3.7
4	Ed Pinckney	MIA	34.0	205.74	108.862080	Villanova	USA	1985	1	10	...	2.4

5 rows × 21 columns

Much better. Now, we're interested in the factors that may play into a player's longevity. We have an 'age' column, let's find the ten oldest players that have played in the NBA.

In [29]:
```
display(df_nba.sort_values(by = 'age', ascending = False).head(n = 10))
```

	player_name	team_abbreviation	age	player_height	player_weight	college	country	draft_year	draft_round	draft_number	...
4698	Kevin Willis	DAL	44.0	213.36	111.130040	Michigan State	USA	1984	1	11	...
270	Robert Parish	CHI	43.0	215.90	110.676448	Centenary (LA)	USA	1976	1	8	...
10818	Vince Carter	ATL	43.0	198.12	99.790240	North Carolina	USA	1998	1	5	...
5680	Dikembe Mutombo	HOU	43.0	218.44	117.933920	Georgetown	Congo	1991	1	4	...
4892	Dikembe Mutombo	HOU	42.0	218.44	117.933920	Georgetown	Congo	1991	1	4	...
12149	Udonis Haslem	MIA	42.0	203.20	106.594120	Florida	USA	Undrafted	Undrafted	Undrafted	...
10240	Vince Carter	ATL	42.0	198.12	99.790240	North Carolina	USA	1998	1	5	...
3795	Kevin Willis	ATL	42.0	213.36	111.130040	Michigan State	USA	1984	1	11	...
1291	Herb Williams	NYK	41.0	210.82	117.933920	Ohio State	USA	1981	1	14	...
2818	John Stockton	UTA	41.0	185.42	79.378600	Gonzaga	USA	1984	1	16	...

10 rows × 21 columns

Looks like our analysis is capturing multiple of our NBA veterans' seasons. Let's capture one row per player.

In [30]:
```
display(df_nba.sort_values(by = 'age', ascending = False).drop_duplicates(['player_name']).head(n = 10))
```

	player_name	team_abbreviation	age	player_height	player_weight	college	country	draft_year	draft_round	draft_number	...
4698	Kevin Willis	DAL	44.0	213.36	111.130040	Michigan State	USA	1984	1	11	...
270	Robert Parish	CHI	43.0	215.90	110.676448	Centenary (LA)	USA	1976	1	8	...
10818	Vince Carter	ATL	43.0	198.12	99.790240	North Carolina	USA	1998	1	5	...
5680	Dikembe Mutombo	HOU	43.0	218.44	117.933920	Georgetown	Congo	1991	1	4	...
12149	Udonis Haslem	MIA	42.0	203.20	106.594120	Florida	USA	Undrafted	Undrafted	Undrafted	...
1291	Herb Williams	NYK	41.0	210.82	117.933920	Ohio State	USA	1981	1	14	...
2818	John Stockton	UTA	41.0	185.42	79.378600	Gonzaga	USA	1984	1	16	...
10292	Dirk Nowitzki	DAL	41.0	213.36	111.130040	None	Germany	1998	1	9	...
732	Charles Jones	HOU	41.0	205.74	97.522280	Albany State (GA)	USA	1979	8	165	...
11799	Joe Johnson	BOS	41.0	200.66	108.862080	Arkansas	USA	2001	1	10	...

10 rows × 21 columns

But age is only one (flawed) way of measuring career longeivity. A more complete metric for what we're looking for is number of NBA seasons played.

In [ ]:
```
df_season_count = df_nba.groupby(['player_name']).size().reset_index(name='season_count')
print('Top 10 players by seasons played:')
display(df_season_count.sort_values(by = 'season_count', ascending = False).head(n = 10))
```

Cool! We have a lot of recognizable stars from recent years. Then lets add these counts value back into the original dataframe, associated to each of the player's entries. Note- if running the below cell multiple times, the below line will only run the first time, to prevent duplication of 'season_count'.

In [45]:
```
if 'season_count' not in df_nba.columns:
    df_nba = df_nba.merge(df_season_count, how='left', on='player_name')

# And, lets save and display that merged dataframe, keeping the player's first season as the entry.
# First, lets sort the dataframe by player name and season to get each player's seasons collected, with their rookie season fir
df_nba = df_nba.sort_values(by = ['player_name', 'season'], ascending = True)
# Now, let's drop non-rookie seasons to avoid our friend Vince Carter being the only 22 entries we see.
df_first_season = df_nba.drop_duplicates(['player_name'], keep='first').sort_values(by = 'season_count', ascending = False)
print('Season count applied to the dataframe:')
display(df_first_season.head(n = 10))

# Let's double check how many players exist in our dataframe
print('Number of players who played one season in the NBA by length of first_season df: {}'.format(len(df_first_season)))
print('Unique player_name values from the original dataframe: {}'.format(len(df_nba['player_name'].unique())))
```

```
Season count applied to the dataframe:
```

	player_name	team_abbreviation	age	player_height	player_weight	college	country	draft_year	draft_round	draft_number	...	re
1005	Vince Carter	TOR	22.0	200.66	97.522280	North Carolina	USA	1998	1	5	...	5.
1238	Dirk Nowitzki	DAL	21.0	213.36	107.501304	None	Germany	1998	1	9	...	3.
1757	Jamal Crawford	CHI	21.0	195.58	90.718400	Michigan	USA	2000	1	8	...	1.
337	Kevin Garnett	MIN	21.0	210.82	99.790240	None	USA	1995	1	5	...	8.
342	Kobe Bryant	LAL	18.0	200.66	90.718400	None	USA	1996	1	13	...	1.
2357	Tyson Chandler	CHI	19.0	215.90	106.594120	None	USA	2001	1	2	...	4.
1170	Paul Pierce	BOS	21.0	200.66	99.790240	Kansas	USA	1998	1	10	...	6.
3120	LeBron James	CLE	19.0	203.20	108.862080	None	USA	2003	1	1	...	5.
3306	Carmelo Anthony	DEN	20.0	203.20	99.790240	Syracuse	USA	2003	1	3	...	6.
567	Tim Duncan	SAS	22.0	213.36	112.490816	Wake Forest	US Virgin Islands	1997	1	1	...	11.

10 rows × 22 columns

```
Number of players who played one season in the NBA by length of first_season df: 2463
Unique player_name values from the original dataframe: 2463
```

Checks out. However, there are data limitations with approaching measuring longeivity by the seasons that exist in our dataframe. What comes to mind initially is that players whose careers dont fit neatly within the timespan the data skew our eventual dependent variable, season_count. Let's filter out players whose draft years are earlier than the first year of data, and who played in the last season of data.

In [46]:
```
# First year of data should be the 1996-97 season, but lets check:
print('We shouldnt have any draft years before: {}'.format(df_nba['season'].min()))
df_nba_filtered = df_nba[df_nba['draft_year'] >= df_nba['season'].min()]
print('We dont have any draft years before: {}'.format(df_nba_filtered['draft_year'].min()))

# And whats the last year of data? We should not capture data from players who may play more seasons, so while not a completely
# exhaustive measure, let's filter players who played in the last season of data
print('Latest season in original data is: {}'.format(df_nba_filtered['season'].max()))
df_last_season = df_nba_filtered[df_nba_filtered['season'] == df_nba_filtered['season'].max()]
```

```
df_nba_filtered = df_nba_filtered[~df_nba_filtered['player_name'].isin(df_last_season['player_name'])]
print('Latest season in filtered data is: {}'.format(df_nba_filtered['season'].max()))
# Let's double check if Lebron James, currently active NBA player, is in our filtered dataframe for any of his seasons.
print('\nIs Lebron still around?')
display(df_nba_filtered[df_nba_filtered['player_name'] == 'LeBron James'])
print('Nope!')

# Great! Let's filter the dataframe further to only include the first season of our remaining players, as we did previously
print('\nRookie seasons of players with complete careers within data:')
df_nba_filtered = df_nba_filtered.sort_values(by = ['player_name', 'season'], ascending = True)
df_nba_filtered = df_nba_filtered.drop_duplicates(['player_name'], keep='first').sort_values(by = 'season_count', ascending = F
display(df_nba_filtered.head(n = 10))
```

```
We shouldnt have any draft years before: 1996-97
We dont have any draft years before: 1997
Latest season in original data is: 2021-22
Latest season in filtered data is: 2020-21

Is Lebron still around?
```

player_name	team_abbreviation	age	player_height	player_weight	college	country	draft_year	draft_round	draft_number	...	reb	ast

0 rows × 22 columns

```
Nope!

Rookie seasons of players with complete careers within data:
```

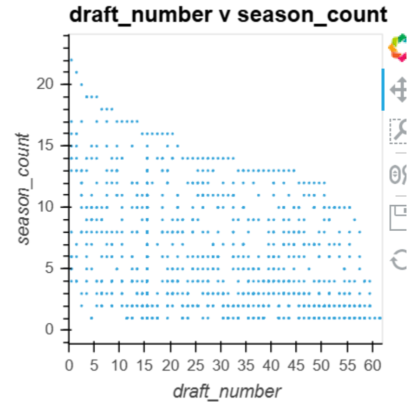
	player_name	team_abbreviation	age	player_height	player_weight	college	country	draft_year	draft_round	draft_number	...	re
1005	Vince Carter	TOR	22.0	200.66	97.522280	North Carolina	USA	1998	1	5	...	5.
1238	Dirk Nowitzki	DAL	21.0	213.36	107.501304	None	Germany	1998	1	9	...	3.
1757	Jamal Crawford	CHI	21.0	195.58	90.718400	Michigan	USA	2000	1	8	...	1.
1382	Jason Terry	ATL	22.0	187.96	78.017824	Arizona	USA	1999	1	10	...	2.
2357	Tyson Chandler	CHI	19.0	215.90	106.594120	None	USA	2001	1	2	...	4.
1170	Paul Pierce	BOS	21.0	200.66	99.790240	Kansas	USA	1998	1	10	...	6.
567	Tim Duncan	SAS	22.0	213.36	112.490816	Wake Forest	US Virgin Islands	1997	1	1	...	11.
1158	Nazr Mohammed	PHI	21.0	208.28	108.862080	Kentucky	USA	1998	1	29	...	1.
2233	Pau Gasol	MEM	21.0	213.36	102.965384	None	Spain	2001	1	3	...	8.
2315	Tony Parker	SAS	20.0	187.96	80.285784	None	France	2001	1	28	...	2.

10 rows × 22 columns

Visually inspecting the top results, seems like each of these players were drafted relatively highly (by draft number). Many seem to have not attended college (this could be for a number of reasons), and more than expected are not from the United States.

In [47]:
```
# Let's explore a few of these variables. First, let's graph the relationship between seasons_count and draft_number.
scatter = hv.Scatter(df_nba_filtered[['draft_number','season_count']], label='draft_number v season_count')
scatter = scatter.opts(xticks=10, size=1)
scatter
```

Out[47]:



That's a lot of points, and it doesn't fully capture all of the data. Let's layer on a regression line to make some more sense of it. But first, we're going to need to further clean draft_number to become a numeric field. We also see that some entries in draft_number are 'Undrafted'. For ease of analysis but at the risk of a less than statistically rigorous treatment of the data, we're going to code 'Undrafted' as the max + 1 of the draft_number column.

In [34]:
```
df_further_filtered = df_nba_filtered[df_nba_filtered['draft_number'] != 'Undrafted']
max_draft_num = pd.to_numeric(df_further_filtered['draft_number']).max()
print('Our max draft position is: {}'.format(max_draft_num))

# Great, let's apply that result to df_nba_filtered, transforming 'Undrafted' to undrafted_draft_num
df_nba_filtered['draft_number'] = df_nba_filtered['draft_number'].apply(lambda x: str(max_draft_num + 1) if x == 'Undrafted' el
# Cool, let's check if this worked.
print('New max after transformation is: {}'.format(pd.to_numeric(df_nba_filtered['draft_number']).max()))
print('\nAnyone still "Undrafted"?')
display(df_nba_filtered[df_nba_filtered['draft_number'] == 'Undrafted'])
print('Nope!')
# What about one of the players who went undrafted previously, is he labeled properly?
print('\nWhat about Ben Wallace?')
display(df_nba_filtered[df_nba_filtered['player_name'] == 'Ben Wallace'])
print('Properly labeled!')
```

```
Our max draft position is: 82
New max after transformation is: 83

Anyone still "Undrafted"?
```

player_name	team_abbreviation	age	player_height	player_weight	college	country	draft_year	draft_round	draft_number	...	reb	ast

0 rows × 22 columns

```
Nope!

What about Ben Wallace?
```

	player_name	team_abbreviation	age	player_height	player_weight	college	country	draft_year	draft_round	draft_number	...	reb	a
115	Ben Wallace	WAS	22.0	205.74	108.86208	Virginia Union	USA	Undrafted	Undrafted	83	...	1.7	(

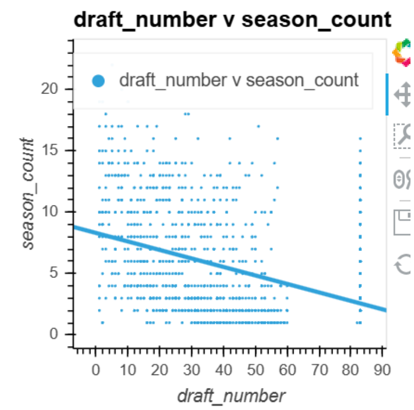
1 rows × 22 columns

```
Properly labeled!
```

We'll now make the df_nba_filtered column numeric and graph the relationship

In [35]:
```
df_nba_filtered['draft_number'] = pd.to_numeric(df_nba_filtered['draft_number'])
scatter = hv.Scatter(df_nba_filtered[['draft_number','season_count']], label='draft_number v season_count')
scatter = scatter.opts(xticks=10, size=1)
scatter * hv.Slope.from_scatter(scatter)
```

Out[35]:



Excellent! Seems there is definitely a relationship between the draft position and seasons played. As draft_number increases, average season_count decreases. We'll look to explore other relationships.

Let's clean our dataset a bit further to make things digestible. Goal is to either make each variable analyzed numeric or categorical. Let's start with college and country of birth. I'd like to make each of these binary or dummy variables. For college, the binary elements will be college, or no college. For country, the binary split will be USA or not USA.

In [36]:
```
# Let's start with whether a player played basketball in college. 0 represents No, 1 represents Yes.
df_nba_filtered['college'] = np.where(df_nba_filtered['college'] == 'None', 0, 1)

# Great, let's move on to country. 0 represents non-USA country of origin, 1 represents USA as a country of origin.
df_nba_filtered['country'] = np.where(df_nba_filtered['country'] != 'USA', 0, 1)
print('\nCollege and country coded as dummy variables:')
display(df_nba_filtered[['player_name','age','college','country','season_count']].head())
```

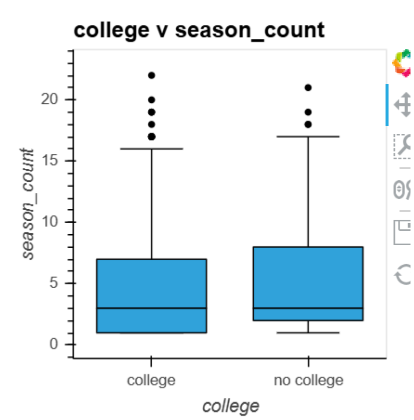
```
College and country coded as dummy variables:
```

	player_name	age	college	country	season_count
1005	Vince Carter	22.0	1	1	22
1238	Dirk Nowitzki	21.0	0	0	21
1757	Jamal Crawford	21.0	1	1	20
1382	Jason Terry	22.0	1	1	19
2357	Tyson Chandler	19.0	0	1	19

Let's visualize these newly coded relationships. We'll start with comparing the season_count for players who played in college to those that did not:

In [37]:
```
df_nba_filtered_plot = df_nba_filtered.copy()
df_nba_filtered_plot['college'] = np.where(df_nba_filtered_plot['college'] == 0, 'no college', 'college')
college_boxwhisker = hv.BoxWhisker((df_nba_filtered_plot['college'],df_nba_filtered_plot['season_count']),
              'college', 'season_count', label = 'college v season_count')
college_boxwhisker
```

Out[37]:

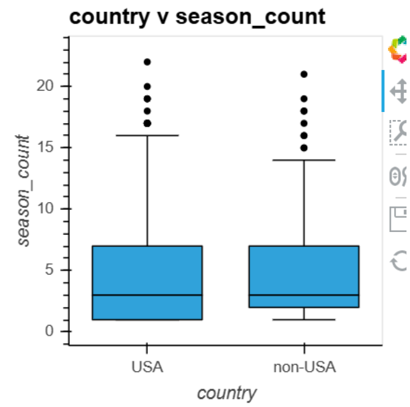


Looks like collegiate players have a shorter career than their non-collegiate NBA peers at most comparison points above the 25th percentile.

Let's see what impact USA and non-USA origins have:

In [38]:
```
df_nba_filtered_plot['country'] = np.where(df_nba_filtered_plot['country'] == 0, 'non-USA', 'USA')
country_boxwhisker = hv.BoxWhisker((df_nba_filtered_plot['country'],df_nba_filtered_plot['season_count']),
              'country', 'season_count', label = 'country v season_count')
country_boxwhisker
```

Out[38]:



Interesting. Seems like the relationship is similar to the collegiate analysis at the lower end of the distribution, but at the upper levels of the data USA players have longer careers.

Now that we've looked at a few variables directly, let's keep get our table ready for the multiple linear regression analysis. We'll convert the last remaining object type, draft_year, to an int, converting 'Undrafted' column names to their corresponding years, with the resulting column being first_season.

In [39]:
```
df_nba_filtered['first_season'] = df_nba_filtered['season'].str[:4].astype('int64')
```

We'll need to remove columns that are correlated directly, so let's remove draft_year (correlated directly with first_season), draft_round (correlated directly with draft_number). team_abbreviation is also removed for simplicity's sake.

In [40]:
```
if 'season' in df_nba_filtered.columns:
    df_nba_filtered = df_nba_filtered.drop(['season', 'draft_year', 'draft_round', 'team_abbreviation', 'player_name'], axis=1)
print('\nDataframe ready for analysis:')
display(df_nba_filtered.head())
```

```
Dataframe ready for analysis:
```

	age	player_height	player_weight	college	country	draft_number	gp	pts	reb	ast	net_rating	oreb_pct	dreb_pct	usg_pct	ts_pct
1005	22.0	200.66	97.522280	1	1	5	50	18.3	5.7	3.0	1.2	0.061	0.127	0.257	0.516
1238	21.0	213.36	107.501304	0	0	9	47	8.2	3.4	1.0	-1.5	0.052	0.139	0.223	0.492
1757	21.0	195.58	90.718400	1	1	8	61	4.6	1.5	2.3	-14.1	0.010	0.092	0.178	0.442
1382	22.0	187.96	78.017824	1	1	10	81	8.1	2.0	4.3	-7.0	0.014	0.083	0.191	0.497
2357	19.0	215.90	106.594120	0	1	2	71	6.1	4.8	0.8	-7.1	0.097	0.186	0.164	0.543

Ok- we're ready for the grand finale. We have each of our variables prepared and appropriately coded, now its time to see which of these variables out of a player's first NBA season are actually impactful on season_count.

In [41]:
```
dependent_matrix = df_nba_filtered['season_count']
coefficient_matrix = df_nba_filtered.copy()
if 'season_count' in coefficient_matrix.columns:
    coefficient_matrix = coefficient_matrix.drop(['season_count'], axis=1)
solution_array = np.linalg.lstsq(coefficient_matrix, dependent_matrix, rcond=None)
results_data = {'solution': solution_array[0], 'residual': solution_array[3]}
results_df = pd.DataFrame(results_data, index=coefficient_matrix.columns)
display(results_df)
```

	solution	residual
age	-0.227284	76964.466715
player_height	0.040884	1278.270784
player_weight	-0.011803	820.200953
college	0.150368	735.377887
country	0.315312	545.216394
draft_number	-0.026629	175.938554
gp	0.049218	105.672314
pts	0.005609	73.437218
reb	0.337637	32.934728
ast	0.398629	27.139606
net_rating	0.011656	16.959571
oreb_pct	2.584431	9.127305
dreb_pct	-1.750185	4.830387
usg_pct	3.771839	2.478831
ts_pct	0.434993	2.097376
ast_pct	0.632961	1.889774
first_season	0.000211	1.577368

Perhaps predictably, somewhat strange and not particularly satisfying results with that many variables in the matrix. Lets try with fewer variables, eliminating most of the rookie season stats aside from games played.

In [42]:
```
coefficient_matrix = df_nba_filtered.copy()
if 'season_count' in coefficient_matrix.columns:
    coefficient_matrix = coefficient_matrix.drop(['season_count', 'pts', 'reb', 'ast', 'net_rating',
                                                   'oreb_pct', 'dreb_pct', 'usg_pct', 'ts_pct', 'ast_pct'], axis=1)
solution_array = np.linalg.lstsq(coefficient_matrix, dependent_matrix, rcond=None)
results_data = {'solution': solution_array[0], 'residual': solution_array[3]}
results_df = pd.DataFrame(results_data, index=coefficient_matrix.columns)
display(results_df)
```

	solution	residual
age	-0.218188	76963.930402
player_height	0.022104	1274.148209
player_weight	-0.004750	774.942651
college	0.098787	544.590451
country	0.354821	175.655173
draft_number	-0.030555	73.454170
gp	0.068692	17.100817
first_season	0.002350	9.135913

Still not particularly clear. Additionally, age seems to be poorly fitted in both analyses, generating a huge residual compared to the other variables. The residual components are a little suspect in general given their cascading values and age's large share in both analyses.

Overall, this analysis still yields some interesting results. One of the relationships with the largest solution values is age, with the data showing a strong relationship between younger rookie seasons and overall career length. This makes sense for a number of reasons, younger players that are NBA-ready have proven their level of potential above their peers. Does this mean that the average player more likely to make a long NBA career if they enter the league early? I don't believe so, but what this relationship does show is a selection bias toward talented players entering the league early.

Another interesting component of analysis is the impact that our dummy variables, college and country, have on overall career length. We may be seeing similar reasons for those values being strong predictors of eventual career length to age. A player who skips college may already be seen as NBA-ready, and a player able to make enough of a splash in their country of origin to make it on an NBA team's radar already may have a higher level of talent in some shape or form.

This was a rich set of data to analyze, and much more could be done than I did. Covariance between these variables would be an interesting next step; do players with non-USA origins have a different physical profile, or do they take longer in the league to establish themselves? I would also love to extend analysis to variables I excluded for simplicity's sake. In particular, the drafting team's impact on a player's career length would be very intriguing- are certain teams better or worse at developing players once they are on rosters?

In [ ]:

```

```