Predicting NBA Draftkings Scores

Supervised Learning Capstone

BY: Elijah Woolford

Research Question:

- Can we model and accurately predict player's draftkings points from 2017-18 NBA basketball data?

Gather the data: Web Scraping

- The basketball data I needed was not readily accessible from data sites such as kaggle and data.gov
- I had to scrape the data from basketball-reference.com
- Using the libraries BeautifulSoup and urlib.request, I was able to open the url https://www.basketball-reference.com and parse through the html code to get the necessary tables.

Gather the data: Web Scraping

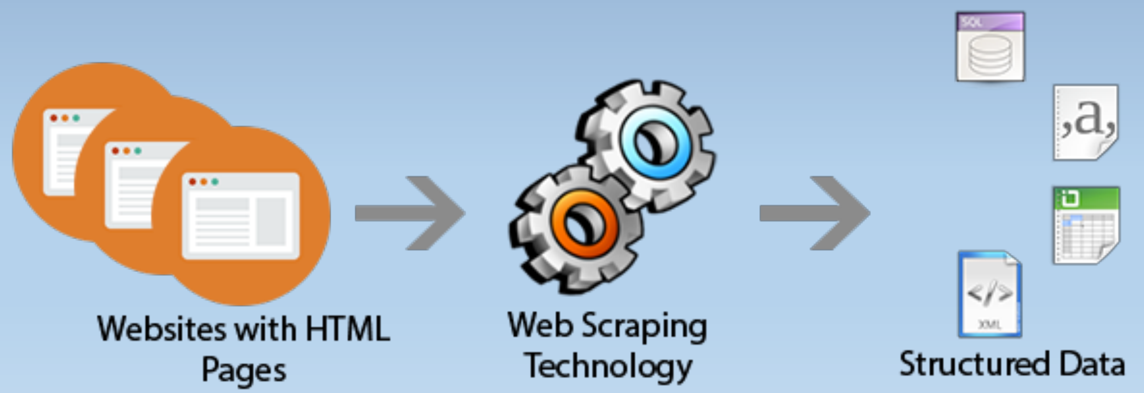
Player Per Game

Share & more ▾ Glossary Hide Partial Rows

Rk	Player	Pos	Age	Tm	G	GS	MP	FG	FGA	FG%	3P	3PA	3P%	2P	2PA	2P%	eFG%	FT	FTA	FT%	ORB	DRB	TRB	AST	STL	BLK	TOV	PF	PS/G
1	Alex Abrines	SG	24	OKC	75	8	15.1	1.5	3.9	.395	1.1	2.9	.380	0.4	0.9	.443	.540	0.5	0.6	.848	0.3	1.2	1.5	0.4	0.5	0.1	0.3	1.7	4.7
2	Quincy Acy	PF	27	BRK	70	8	19.4	1.9	5.2	.356	1.5	4.2	.349	0.4	1.0	.384	.496	0.7	0.9	.817	0.6	3.1	3.7	0.8	0.5	0.4	0.9	2.1	5.9
3	Steven Adams	C	24	OKC	76	76	32.7	5.9	9.4	.629	0.0	0.0	.000	5.9	9.3	.631	.629	2.1	3.8	.557	5.1	4.0	9.0	1.2	1.2	1.0	1.7	2.8	13.9
4	Bam Adebayo	C	20	MIA	69	19	19.8	2.5	4.9	.512	0.0	0.1	.000	2.5	4.8	.523	.512	1.9	2.6	.721	1.7	3.8	5.5	1.5	0.5	0.6	1.0	2.0	6.9
5	Arron Afflalo	SG	32	ORL	53	3	12.9	1.2	3.1	.401	0.5	1.3	.386	0.7	1.7	.413	.485	0.4	0.5	.846	0.1	1.2	1.2	0.6	0.1	0.2	0.4	1.1	3.4
6	Cole Aldrich	C	29	MIN	21	0	2.3	0.2	0.7	.333	0.0	0.0		0.2	0.7	.333	.333	0.1	0.3	.333	0.1	0.6	0.7	0.1	0.1	0.0	0.0	0.5	0.6
7	LaMarcus Aldridge	C	32	SAS	75	75	33.5	9.2	18.0	.510	0.4	1.2	.293	8.8	16.7	.526	.520	4.5	5.3	.837	3.3	5.2	8.5	2.0	0.6	1.2	1.5	2.2	23.1
8	Jarrett Allen	C	19	BRK	72	31	20.0	3.3	5.5	.589	0.1	0.2	.333	3.2	5.3	.599	.596	1.6	2.0	.776	2.0	3.4	5.4	0.7	0.4	1.2	1.1	2.0	8.2
9	Kadeem Allen	PG	25	BOS	18	1	5.9	0.3	1.2	.273	0.0	0.6	.000	0.3	0.6	.545	.273	0.4	0.5	.778	0.2	0.4	0.6	0.7	0.2	0.1	0.5	0.8	1.1
10	Tony Allen	SF	36	NOP	22	0	12.4	2.0	4.1	.484	0.2	0.5	.333	1.8	3.6	.506	.505	0.5	1.0	.524	0.9	1.2	2.1	0.4	0.5	0.1	0.9	2.2	4.7
11	Al-Farouq Aminu	PF	27	POR	69	67	30.0	3.3	8.4	.395	1.8	4.9	.369	1.5	3.5	.432	.503	0.9	1.2	.738	1.4	6.2	7.6	1.2	1.1	0.6	1.1	2.0	9.3
12	Justin Anderson	SF	24	PHI	38	0	13.7	2.3	5.3	.431	0.9	2.7	.330	1.4	2.6	.535	.515	0.7	1.0	.737	0.7	1.8	2.4	0.7	0.4	0.2	0.4	1.4	6.2
13	Kyle Anderson	SF	24	SAS	74	67	26.7	3.1	5.9	.527	0.3	0.8	.333	2.9	5.1	.556	.549	1.4	2.0	.712	1.1	4.2	5.4	2.7	1.6	0.8	1.3	1.5	7.9
14	Ryan Anderson	PF	29	HOU	66	50	26.1	3.1	7.3	.431	2.0	5.1	.386	1.2	2.1	.539	.568	1.1	1.4	.774	1.4	3.6	5.0	0.9	0.4	0.3	0.6	1.9	9.3
15	Ike Anigbogu	C	19	IND	11	0	2.7	0.4	0.8	.444	0.0	0.0		0.4	0.8	.444	.444	0.5	0.5	.833	0.5	0.4	0.8	0.0	0.1	0.3	0.2	0.1	1.2
16	Giannis Antetokounmpo	PF	23	MIL	75	75	36.7	9.9	18.7	.529	0.6	1.9	.307	9.3	16.8	.554	.545	6.5	8.5	.760	2.1	8.0	10.0	4.8	1.5	1.4	3.0	3.1	26.9
17	Carmelo Anthony	PF	33	OKC	78	78	32.1	6.1	15.0	.404	2.2	6.1	.357	3.9	8.9	.437	.476	1.9	2.5	.767	0.9	4.9	5.8	1.3	0.6	0.6	1.3	2.5	16.2
18	OG Anunoby	SF	20	TOR	74	62	20.0	2.2	4.7	.471	1.0	2.7	.371	1.2	2.0	.604	.577	0.5	0.8	.629	0.6	1.9	2.5	0.7	0.7	0.2	0.6	1.8	5.9
19	Ryan Arcidiacono	PG	23	CHI	24	0	12.7	0.7	1.7	.415	0.4	1.3	.290	0.3	0.4	.800	.524	0.2	0.3	.833	0.0	1.0	1.0	1.5	0.5	0.0	0.5	0.8	2.0
20	Trevor Ariza	SF	32	HOU	67	67	33.9	4.0	9.7	.412	2.5	6.9	.368	1.5	2.8	.519	.542	1.1	1.3	.854	0.5	3.9	4.4	1.6	1.5	0.2	0.8	2.0	11.7

Gather the data: Web Scraping

- This link contained the table I needed with all the players and links to their stats for the 2017-2018 season.
- Again using beautifulsoup I was able to find the table and get the extension for each link for each player.



Gather the data: Web Scraping

2017-18 Regular Season

Share & more ▾ Glossary

Rk	G	Date	Age	Tm		Opp		GS	MP	FG	FGA	FG%	3P	3PA	3P%	FT	FTA	FT%	ORB	DRB	TRB	AST	STL	BLK	TOV	PF	PTS	GmSc	+/-
1	1	2017-10-19	24-079	OKC		NYK	W (+21)	0	24:15	1	5	.200	1	5	.200	0	0		0	3	3	0	1	0	2	3	3	-1.4	+23
2	2	2017-10-21	24-081	OKC	@	UTA	L (-9)	0	29:04	2	6	.333	1	4	.250	2	2	1.000	2	2	4	1	1	0	0	1	7	6.9	+6
3	3	2017-10-22	24-082	OKC		MIN	L (-2)	0	14:20	2	2	1.000	0	0		0	0		0	0	0	0	1	0	0	4	4	2.8	+13
4	4	2017-10-25	24-085	OKC		IND	W (+18)	0	13:26	2	5	.400	1	1	1.000	0	0		1	1	2	1	1	0	0	3	5	3.8	+5
5	5	2017-10-27	24-087	OKC	@	MIN	L (-3)	0	8:27	0	1	.000	0	1	.000	0	0		0	0	0	0	0	0	0	3	0	-1.9	+9
6	6	2017-10-28	24-088	OKC	@	CHI	W (+32)	0	18:09	2	8	.250	1	5	.200	0	0		0	2	2	1	2	0	1	2	5	1.7	-1
7	7	2017-10-31	24-091	OKC	@	MIL	W (+19)	0	27:11	2	4	.500	2	2	1.000	0	0		0	1	1	1	1	0	2	4	6	2.4	+11
8	8	2017-11-03	24-094	OKC		BOS	L (-7)	0	12:41	2	5	.400	2	5	.400	0	0		0	1	1	0	0	1	0	3	6	3.1	+6
9	9	2017-11-05	24-096	OKC	@	POR	L (-4)	0	14:36	0	1	.000	0	1	.000	0	0		0	0	0	1	0	0	0	1	0	-0.4	+11
10	10	2017-11-07	24-098	OKC	@	SAC	L (-8)	0	5:13	1	3	.333	0	2	.000	0	0		1	0	1	0	0	0	0	1	2	0.6	-4
11	11	2017-11-09	24-100	OKC	@	DEN	L (-8)	0	12:02	0	0		0	0		1	1	1.000	0	1	1	0	0	0	0	0	1	1.3	-1
12	12	2017-11-10	24-101	OKC		LAC	W (+9)	0	20:42	4	5	.800	3	4	.750	3	4	.750	1	2	3	0	0	0	0	1	14	12.6	-2
13	13	2017-11-12	24-103	OKC		DAL	W (+13)	0	14:34	1	2	.500	1	1	1.000	0	0		1	1	2	0	0	0	1	2	3	1.2	-1
14	14	2017-11-15	24-106	OKC		CHI	W (+13)	0	23:16	1	5	.200	1	5	.200	1	2	.500	0	4	4	0	0	0	0	1	4	1.3	+13
15	15	2017-11-17	24-108	OKC	@	SAS	L (-3)	0	17:47	4	8	.500	1	5	.200	0	0		1	0	1	0	1	0	1	1	9	5.3	0
16	16	2017-11-20	24-111	OKC	@	NOP	L (-7)	0	16:25	2	6	.333	1	4	.250	2	2	1.000	0	1	1	1	0	0	0	1	7	4.2	-17
17	17	2017-11-22	24-113	OKC		GSW	W (+17)	0	6:07	0	1	.000	0	0		0	0		0	0	0	0	1	0	0	2	0	-0.5	+6
18	18	2017-11-24	24-115	OKC		DET	L (-1)	0	8:53	0	2	.000	0	1	.000	0	0		0	1	1	1	1	0	0	0	0	0.6	+1
19	19	2017-11-25	24-116	OKC	@	DAL	L (-16)	0	1:50	0	1	.000	0	1	.000	0	0		0	0	0	0	0	0	0	0	0	-0.7	-1
20		2017-11-29	24-120	OKC	@	ORL	L (-13)	Did Not Play																					

Gather the data: Web Scraping

- With all the links to each of the player's datatables, I removed any duplicate links and once again used beautiful soup to parse through each link and find the table with the data.
- With each table I iterated through each row and column and saved the data in a list as comma separated strings.
- Finally I wrote each player line to a csv file to be imported as a dataframe.

Feature Creation:

- After scraping the data and doing some cleaning, it was time to choose and create features.
- First I wanted to see what columns in the data table are correlated with DK points.

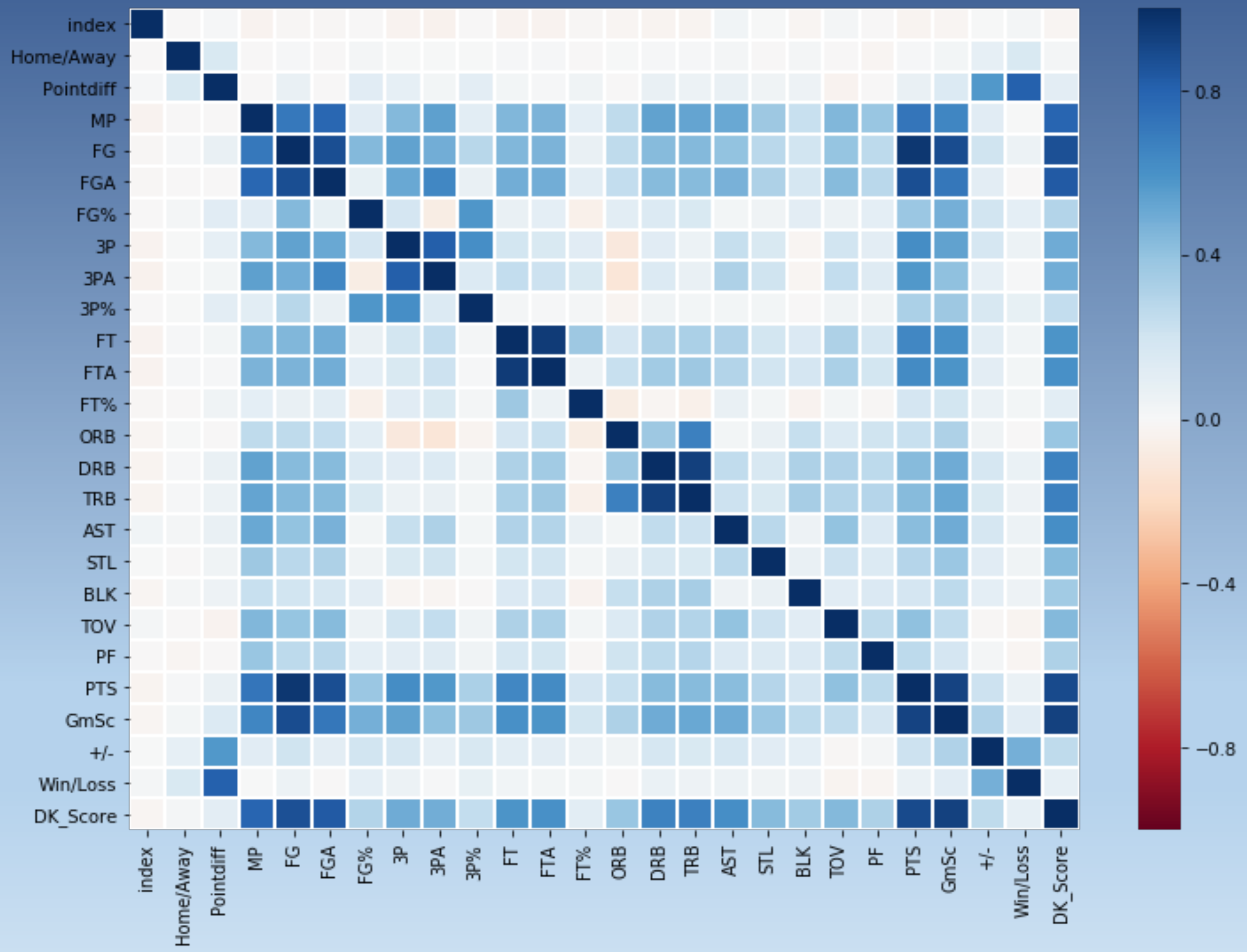
Feature Creation:

Draftkings NBA Scoring System:

- Point +1 Pt
- Made 3pt Shot +0.5 Pts
- Rebound +1.25 Pts
- Assist +1.5 Pts
- Steal +2 Pts
- Block +2 Pts
- Turnover -0.5 pts

- Double-Double {Max 1 Per Player: Points, Rebounds, Assists, Blocks, Steals} +1.5 Pts
- Triple-Double {Max 1 Per Player: Points, Rebounds, Assists, Blocks, Steals} +3 Pts

Feature Creation:



Feature Creation:

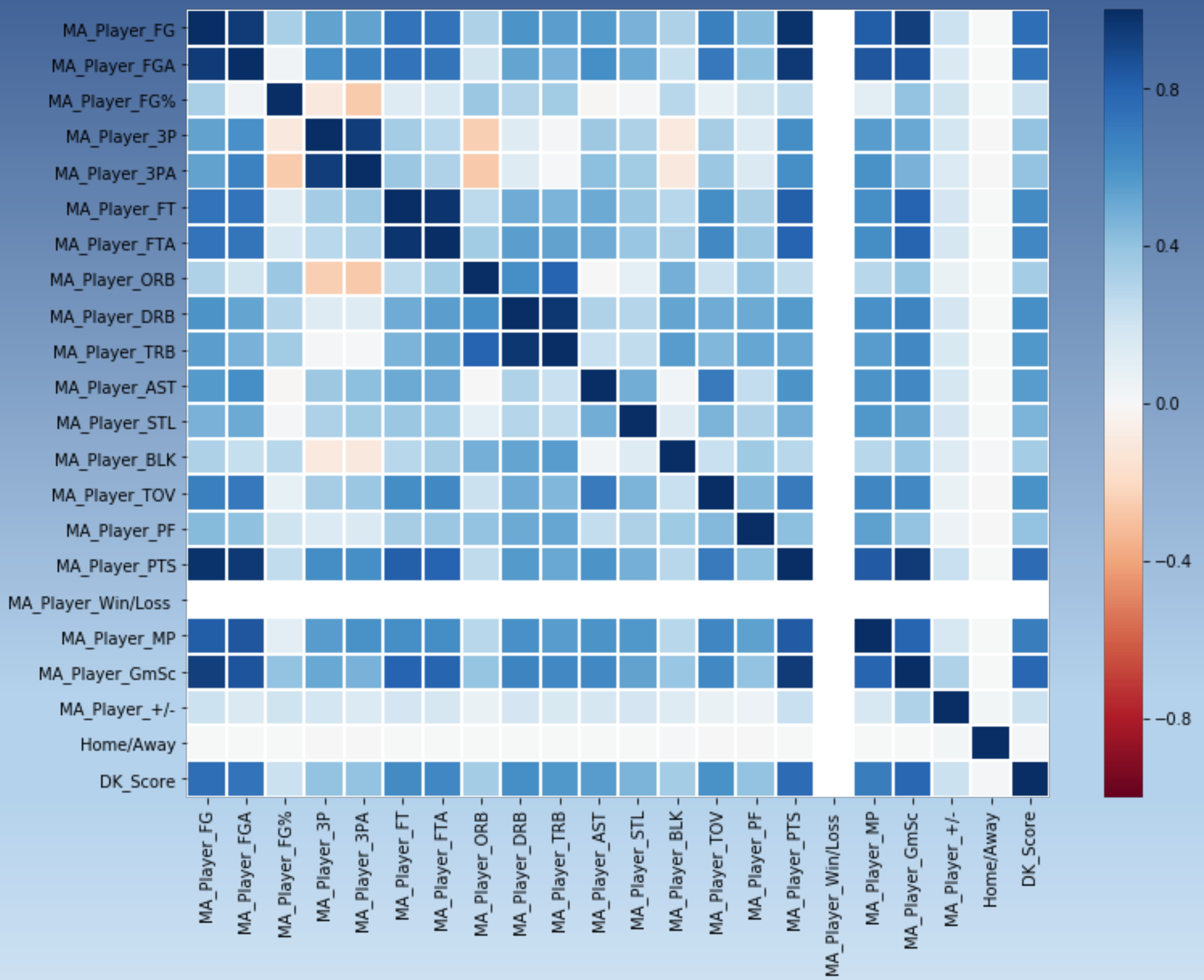
- Looking at the features we see that most of the columns have a positive correlation with DK scores.
- The ones that stand out the most are FG, FGA, MP, PTS, and GmSc
- But we run into issues because we won’t have any of the columns before the game is played in order to predict the DK score of the player

Feature Creation:

- So to solve this I decided to use the moving averages on the previous games to try to predict what the actual game columns would be.

- A lot of players’ draftkings points are based on streaks. So using a moving average would help catch those streaks (indications of higher averages) and help accurately predict these scores.

Feature Creation:



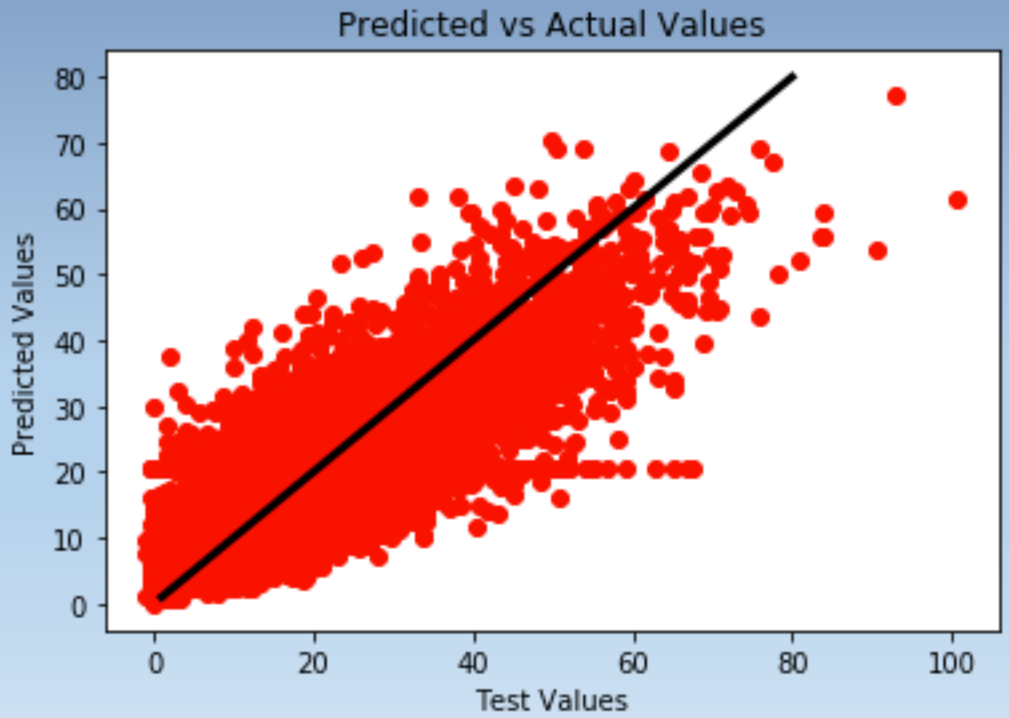
Modeling:

- I decided to try out two different regression models to see if performance differs between the models:
- Linear Regression
- Ridge Regression

Modeling: Linear Regression

- Using Linear regression with the created features we see that there is some signal for the predicted test values compared to the actual test values but could still use some improving.

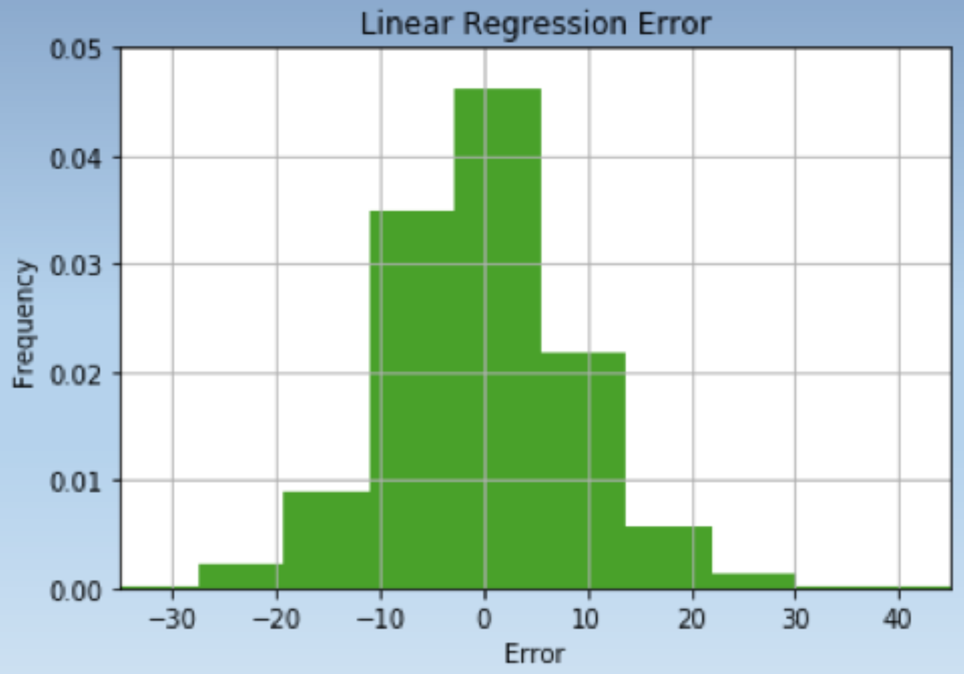
Cross Val Scores:
0.59734155, 0.60200847,
0.60762156, 0.60145765,
0.61501582, 0.59486526



Modeling: Linear Regression

- Here are some more metrics I used to further evaluate the model:

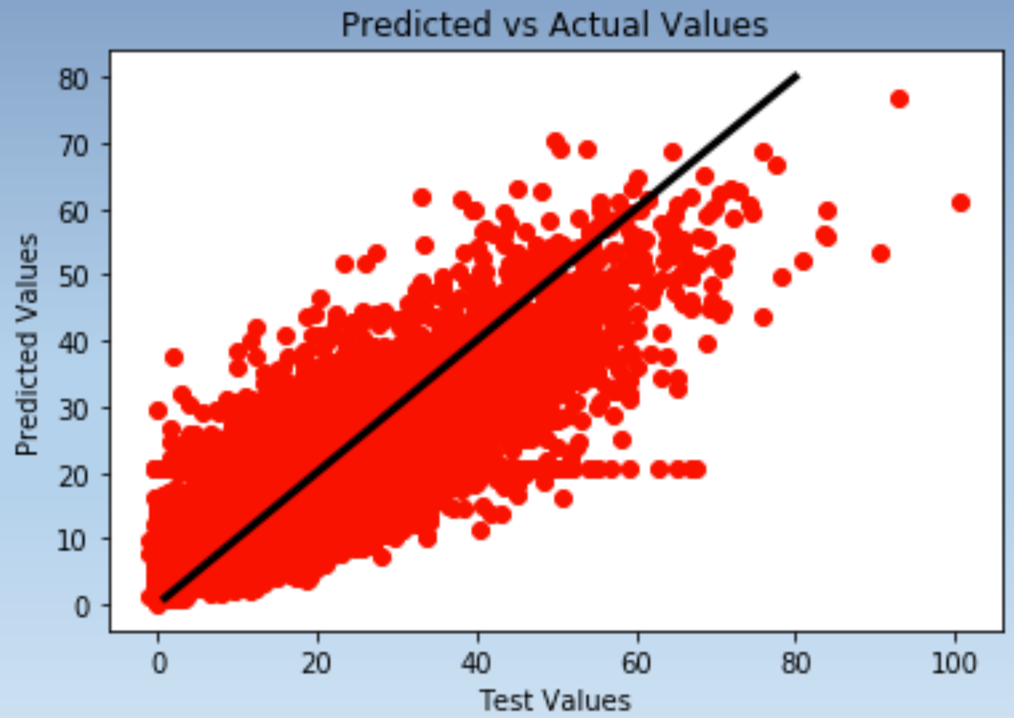
- Actual test value mean: 20.903089887640448
- Predicted test value mean: 20.975292680914517
- Mean absolute error: 6.8039056518838406



Modeling: Ridge Regression

- Like Linear Regression, ridge regression shows signals but need for improvement.
- Although these features are pretty correlated, adjusting the alpha dose not significantly change much of the performance.

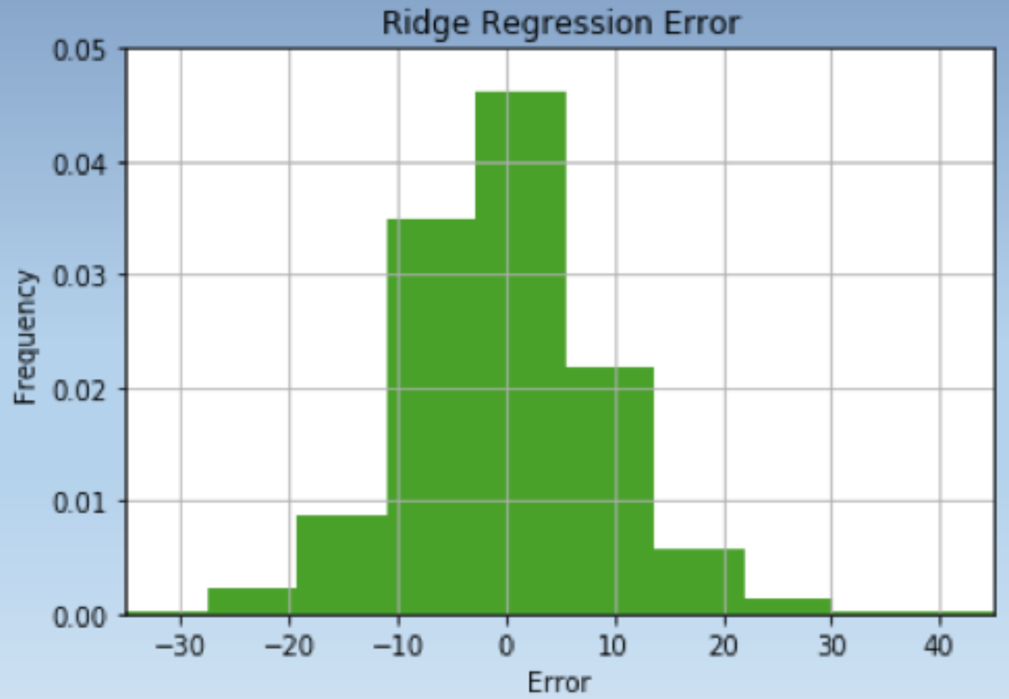
Cross Val Scores:
0.59734041, 0.60138892,
0.60818722, 0.60157627,
0.61567429, 0.59491193



Modeling: Ridge Regression

- The Linear Regression model and the Ridge Regression almost had identical performances

- Actual test value mean : 20.903089887640448
- Predicted test value mean: 20.985472179811175
- Mean absolute error: 6.800937471272061



Improvements:

- Going back to feature creation we can create more features to further highlight some of the columns that are highly correlated with DK score.
- Also, accounting for player coming off and getting injuries can help since historically a player performs a little under average when this happens.
- Scraping some player advanced stats from basketball-reference.com and calculating more moving averages should improve these models.
- Incorporating team defensive features could also improve things.

Further Research:

- This project can go a step further with incorporating draftkings' salaries and lineup construction.
- Draftkings assigns a particular price tag to each player and gives a $50,000 salary cap that a person can't exceed.
- We have to pick 8 players while staying under that salary cap.

Further Research:

- I will incorporate classifier models to help classify over/under priced players based off of salary.
- Also, I would like to build a lineup generator that further classifies the players to choose the highest possible scoring lineup at the "cheapest" salary


THANKS FOR LISTENING
ANY QUESTIONS?