NBA Player Value

Using Machine Learning to Better Run an NBA Franchise

The Problem

- General Managers are tasked with the responsibility of constructing and paying a roster composed of high-level athletes—every one of which feels he ought to be paid what he is worth.
- All too often, these GMs succumb to external pressures: to sign the player the fans want or that the media has placed on a pedestal, to pay a player more than he is worth out of fear of missing out on a player, offering too little to a player out of pride or fear (or both) and missing out—the reasons go on and on.
- None of these pressures is based in anything objective. GMs need something on which to fall to make such costly and high stakes decisions, a means of defending the decisions they make.
- As we saw in Michael Lewis' "Moneyball," professional sports leagues can be ruled by forces other than logic, but the game always has potential to be affected by it.
- Using machine learning, we will look to give any GM an upper hand who is willing to take it.

Sneak Peek: What We Learned

- Machine learning regression models have a hard time pinning down just what exactly gets a player paid.
  - Because these models were able to rely on 30+ features and still had a difficult time predicting, we must conclude that we did not have the features that would predict salary well (i.e. they aren't found on a stats sheet)
- Classifier models do not have the same trouble with predicting a team's season's success.
  - Our models performed well at predicting whether a team would finish the season ranked in the top 4 based on things like how well the team show 3-pointers.
- Feature Importance between our salary regression and our team success classifier varied enough to raise flags—GMs aren't using success predictors to determine player worth.
- The player that deserves to get paid (i.e. who is the most valuable) is the one who can shoot, both accurately and often.

Initial Objectives

Identify Successful Teams

Determine what makes them successful

Determine True Player Value

Make Recommendations

The Data

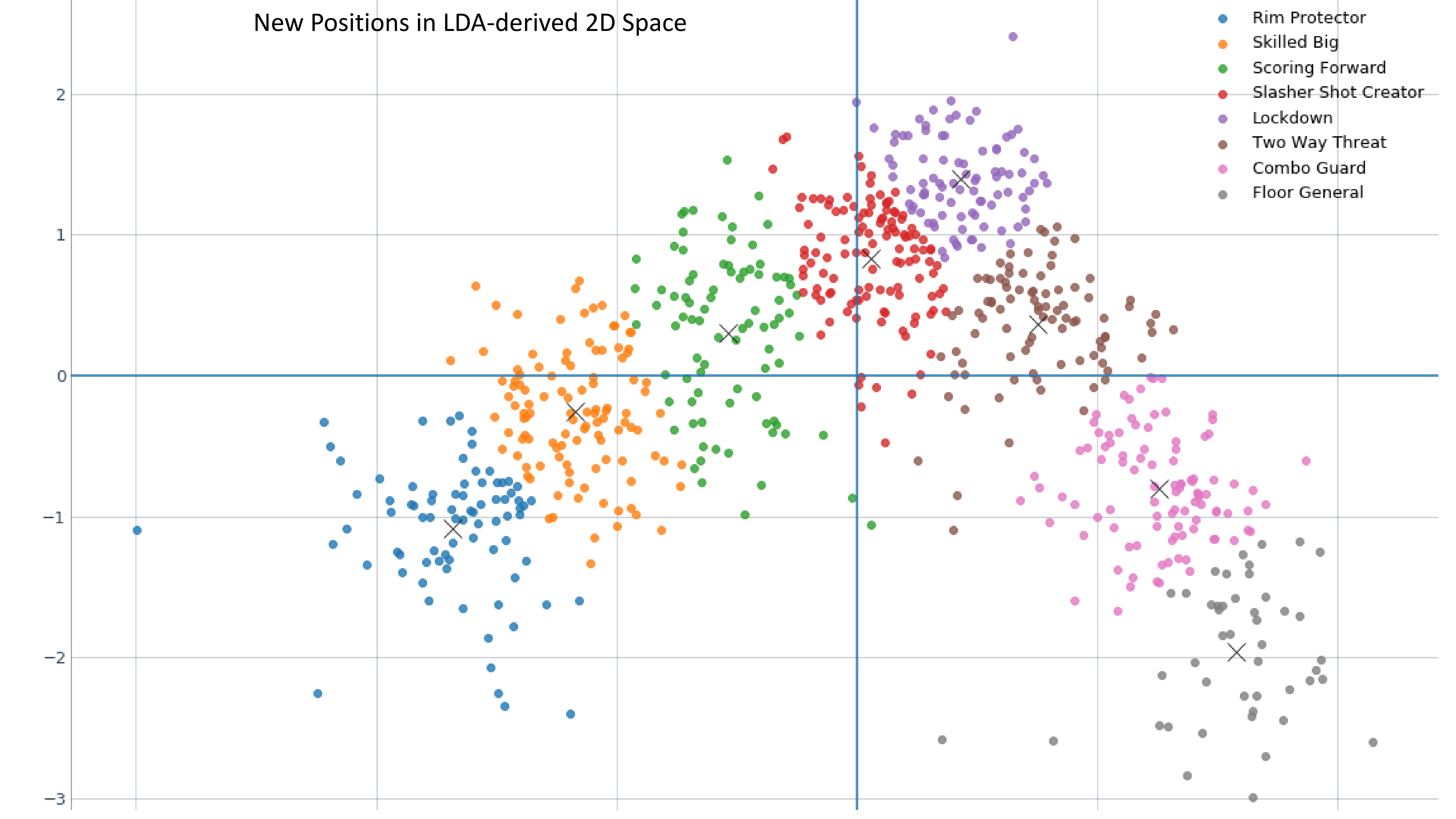
- The data was acquired via webscraping a few different websites using the BeautifulSoup package.
- Data was drawn across multiple websites and many different urls within each website.
- Player data was a composite dataset of "per 100 possessions" stats, advanced stats, and shooting stats from basketball-reference.com, as well as salary stats from hoopshype.com.
- Team data was also a composite dataset of win-loss data and advanced statistics pulled from basketball-reference.com as well.
- All webscraping code can be found here.

Data Cleaning and Wrangling

- The bulk of the data wrangling needed for the player and team data was performed during the webscraping loops—each loop spit out list of pandas DataFrames that would ultimately be concatenated.
- After the webscraping, the majority of wrangling that remained for this data was simply structuring DataFrames such that they could be merged together using pandas merge.
- The data that needed the most cleaning was the salary data from hoopshype.com. For example, a datapoint we might read as “$20,000,000” was actually written in the HTML as “\n\t\t\t$20,000,000\t\t\t\n”—a very messy string.
  - This required me to identify the characters that needed to be dropped before changing the datatype to something we could manipulate (i.e. string to float/integer).
- The data from basketball-reference.com was for the most part very clean, handling null values was an important component of this project. Depending on the feature, some data was filled using the population mean, and for some data, a null value in the wrong feature meant dropping the record entirely.
- The bulk of the data wrangling can be found here.

New Player Positions

- One of our primary objectives was to accurately determine player value. In order to best do this, it was necessary to distinguish differences between players who are listed as similar. Our hypothesis was that there is a more robust method for describing a player's position than just as one of five positions (point guard, shooting guard, small forward, power forward, or center).
  - For example, compare the play styles of Ben Simmons and Steph Curry. Both are listed as Point Guards, but their contributions could not look more different.
- In order to do this, we used our extensive player-level data (per 100 possessions stats, advanced stats, and shooting stats) to cluster players into 8 distinct positions that better described their contributions to the team.

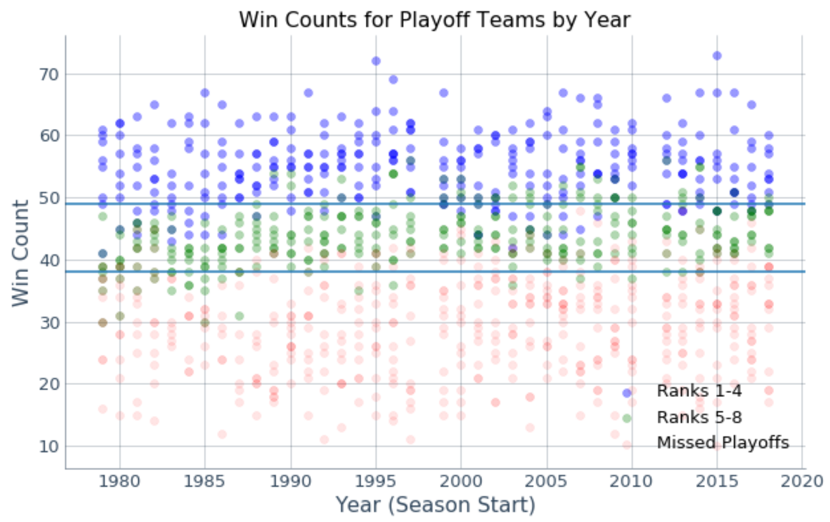


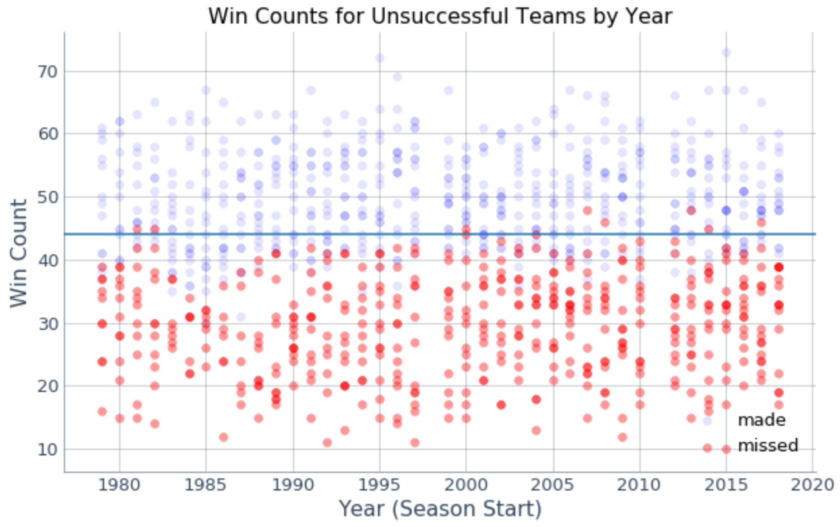
New Positions in LDA-derived 2D Space
Rim Protector
Skilled Big
Scoring Forward
Slasher Shot Creator
Lockdown
Two Way Threat
Combo Guard
Floor General
2
1
0
−1
−2
−3

Dimensionality Reduction

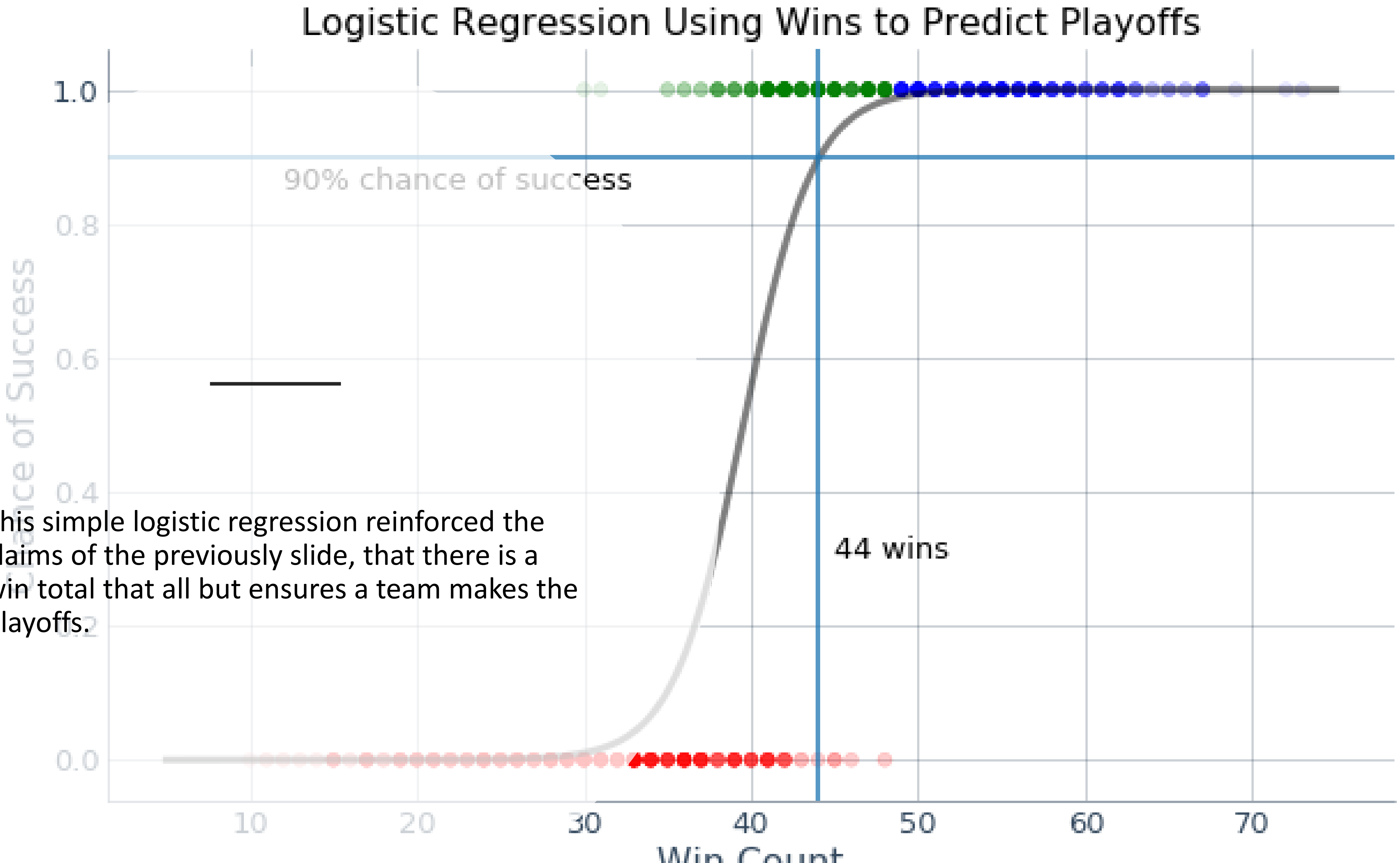
- Principal Component Analysis (PCA)
- Linear Discriminant Analysis (LDA)
- K-means clustering
- Once new labels were merged with our player-level data, we were able to add another layer to our EDA and inferential statistics as we could then consider a player’s cluster when asking questions of value.

Exploratory Data Analysis

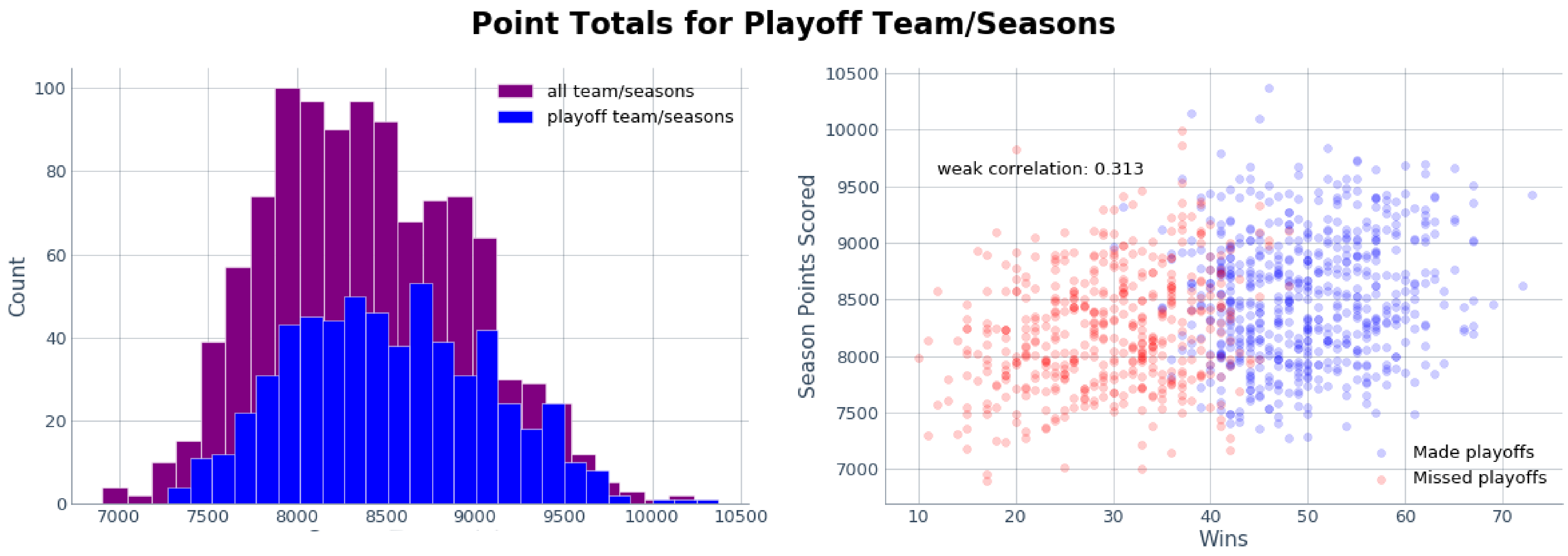




- On the team-level, we wanted to identify success, so that required of us that we define our terms.
  - "Success" for the scope of this project equated to making playoffs, or even better, making the playoffs with a good seeding (1^{st} through 4^{th}).
- Left are two similar plots that show roughly how many wins generated successful season.
  - At 38 wins, a team has a good chance of making playoffs, but at 44 wins, it's highly unlikely that a team would miss playoffs.
  - At 49 wins, a team will very likely have a good playoff seed.



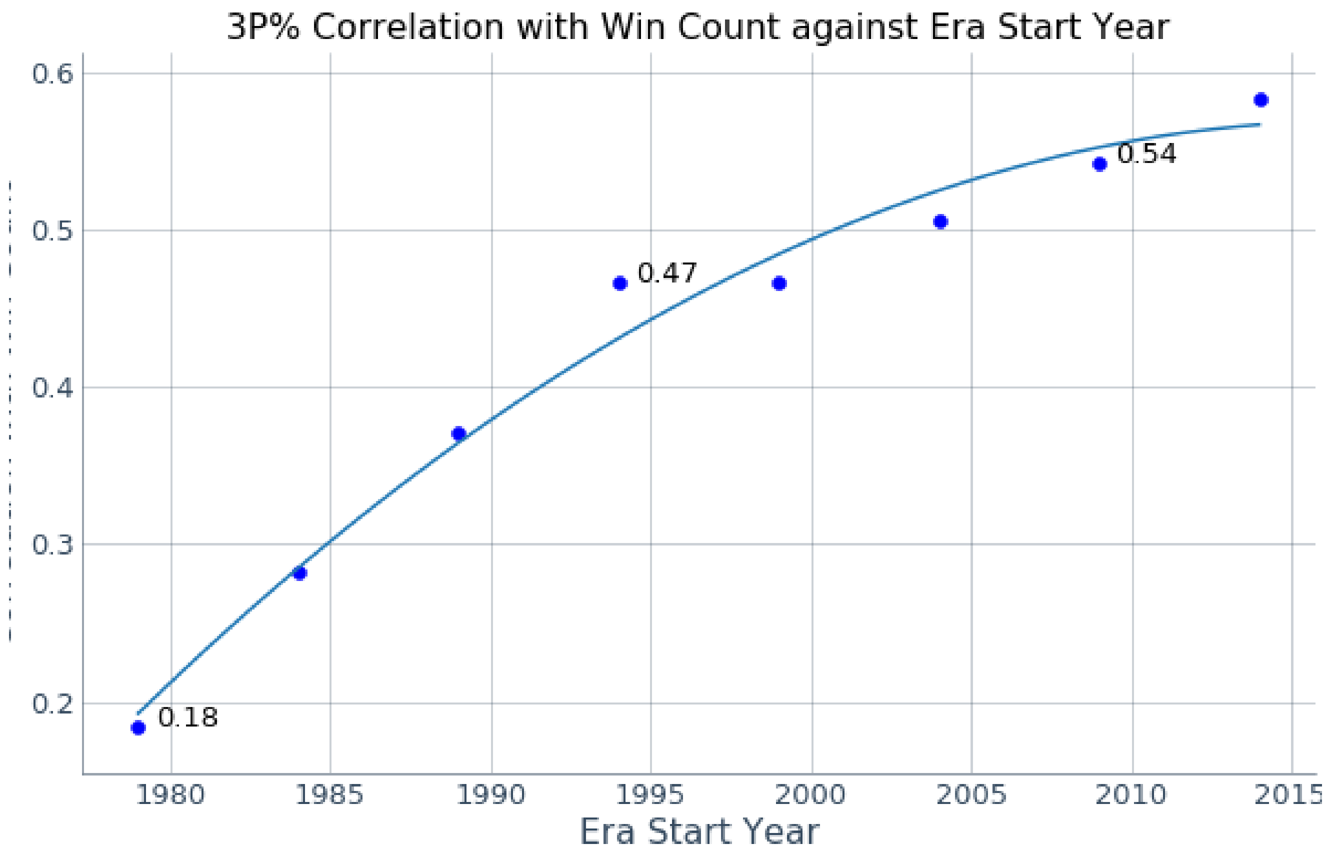
This simple logistic regression reinforced the claims of the previously slide, that there is a win total that all but ensures a team makes the playoffs.

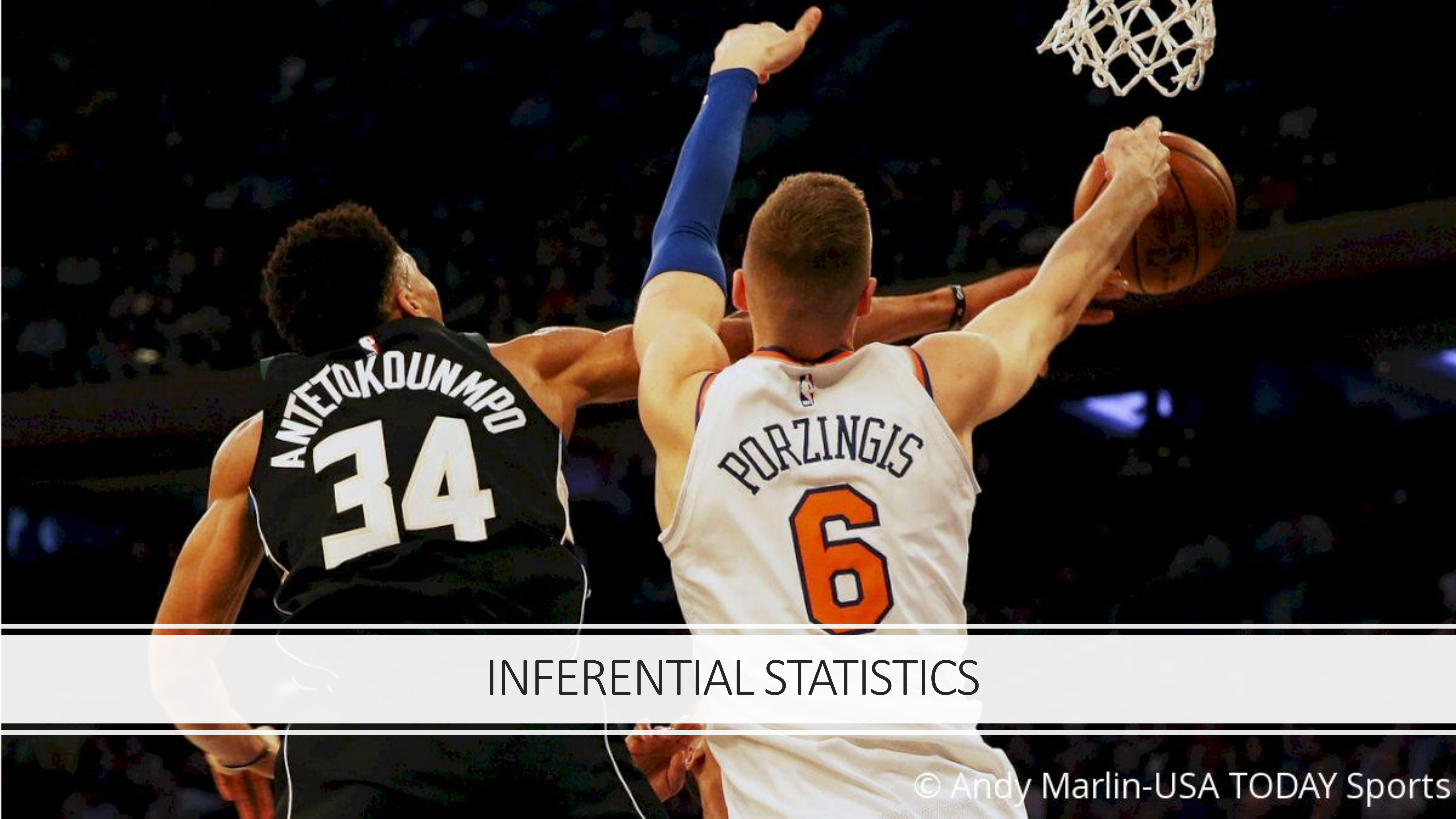


Team-level Win Correlates

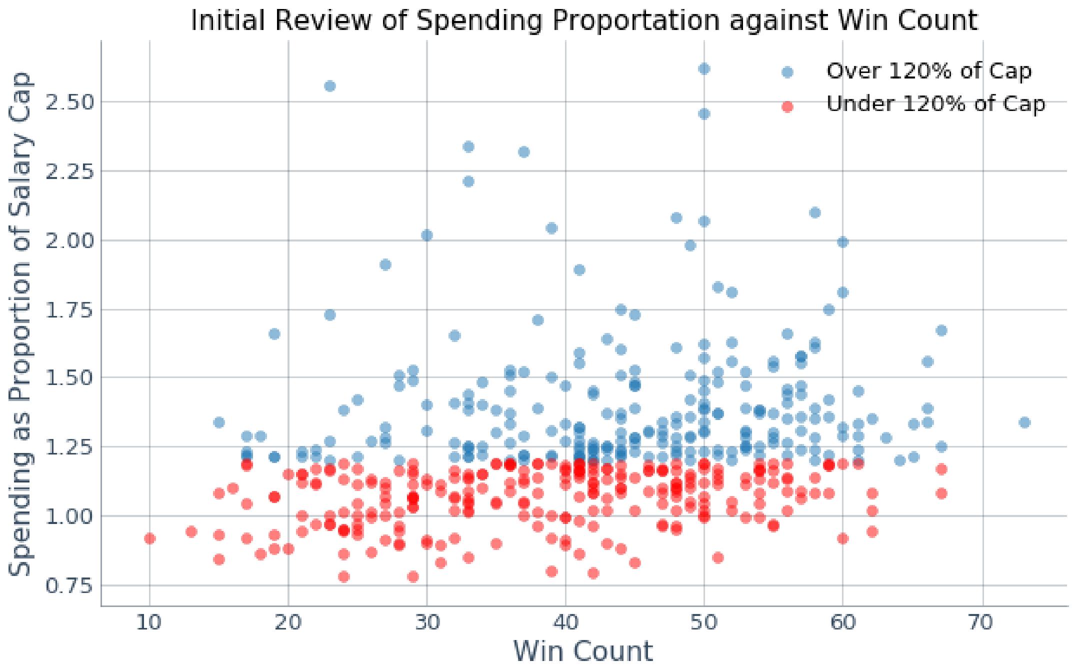
- The next step was to look for correlates of win count. “What did a team that won a bunch of games do well that season?”
- On first pass, we checked the statistic of “points scored”. This, interestingly had a very weak correlation with win count—just because a team is scoring a lot of points, doesn’t mean they’re winning games. Same could be said with the statistics of “points against”.

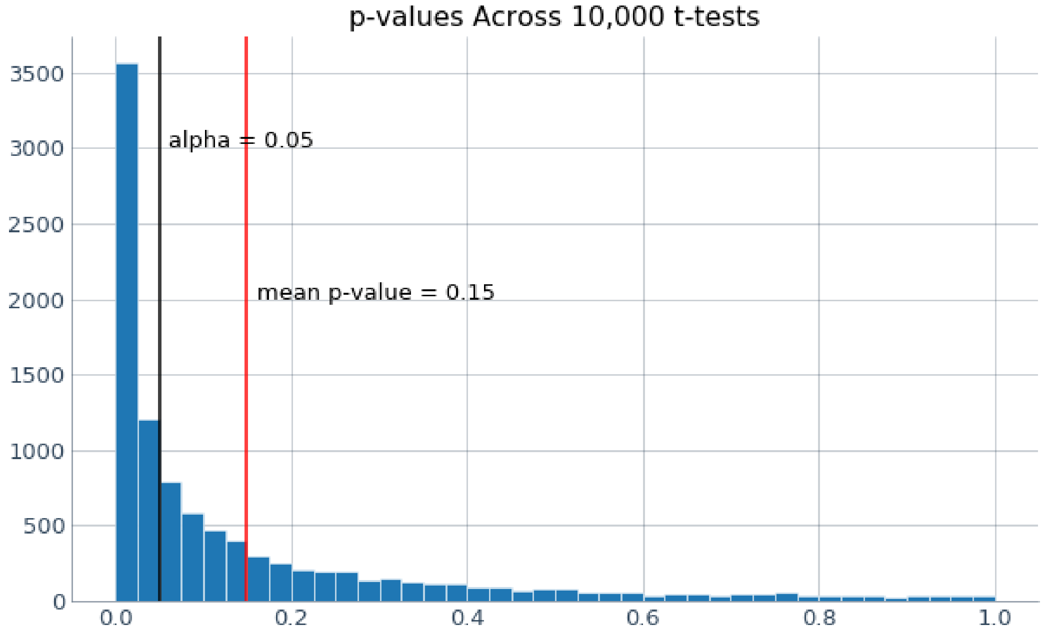
- Where we did see our most notable strong correlation was with the feature “3P%”, or 3-point percentage. Teams that were more accurate from the 3-point line won more games.
  - When we first looked at this correlate, we saw a coefficient of 0.18, but this included team data dating back to 1979.
  - When we limited our data to just team-seasons dating back 5 years (2014-2019) instead, we saw a correlation of 0.58.





INFERENTIAL STATISTICS





Spend to Win?

With our team-level inferential statistics, two sample t-tests of independence showed us that money doesn't buy wins--at least not at a statistically significant level. The differences in mean Win count for teams paying over 120% of the salary cap for any given year, and teams paying under 120% of the salary cap for any given year varied greatly--44 and 38, respectively.

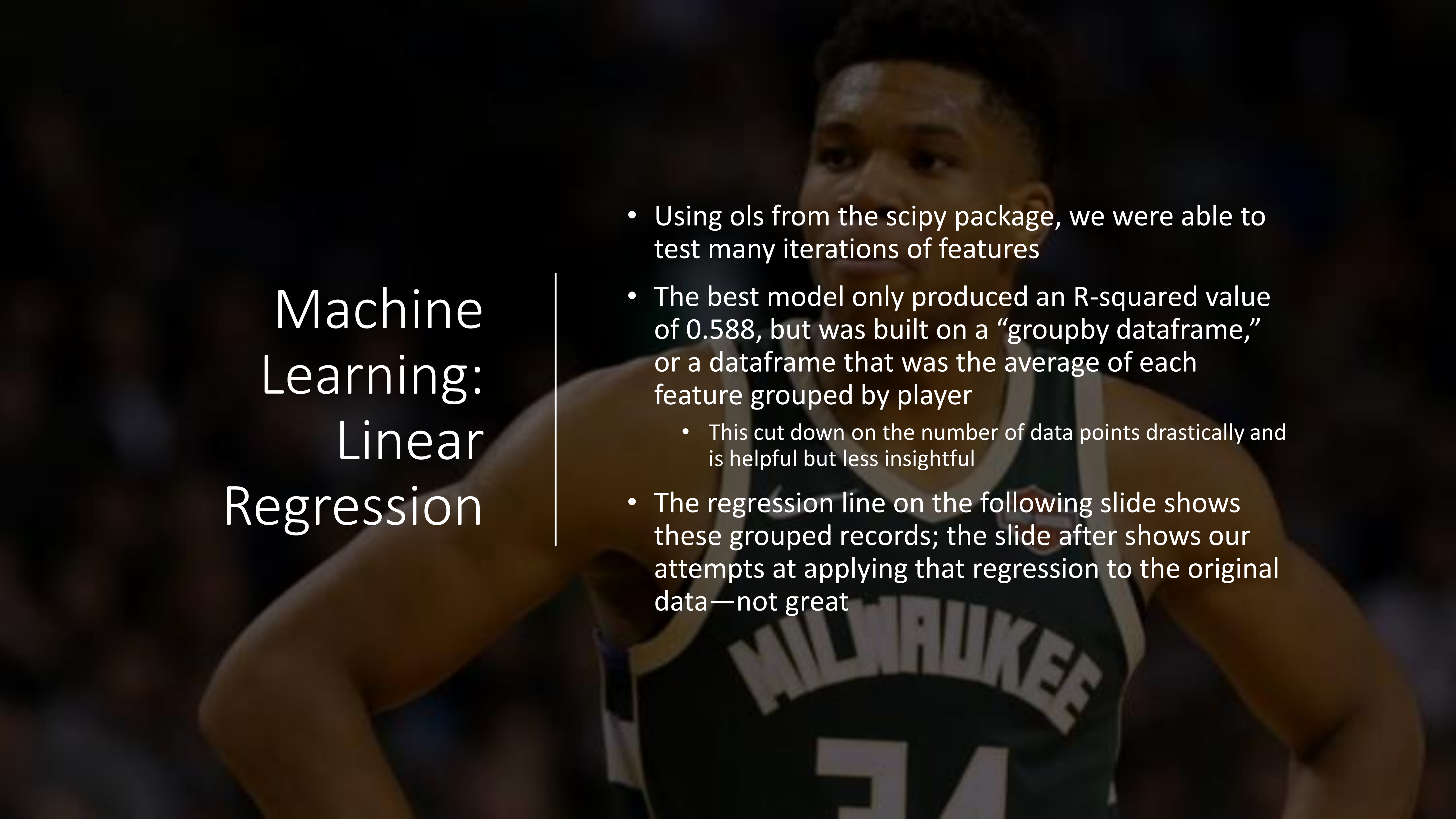
- However, when we sampled from these groups and ran a few t-tests (read: ten thousand t-tests), we saw that that this difference was not in fact significant at even a 0.10 confidence. The resultant p-values were plotted in the histogram (bottom).

Inferential Statistics cont.

Python allowed us to scan for statistically significant differences in 6 different features across 8 subsets of the data, using 10,000 t-tests each time (48,000 in total) in about 2.5 minutes.

- From this scan, we learned the following:
  - the "Lockdown" cluster was flagged for statistically lower mean Player Efficiency Rating (PER)
  - "Rim Protectors" were flagged for having statistically lower mean Usage Rate (USG%)
  - "Combo Guards" were conversely flagged for statistically higher mean Usage Rate
  - "Rim Protectors" were the only cluster with a significant difference (lower) in Points Scored (PTS)

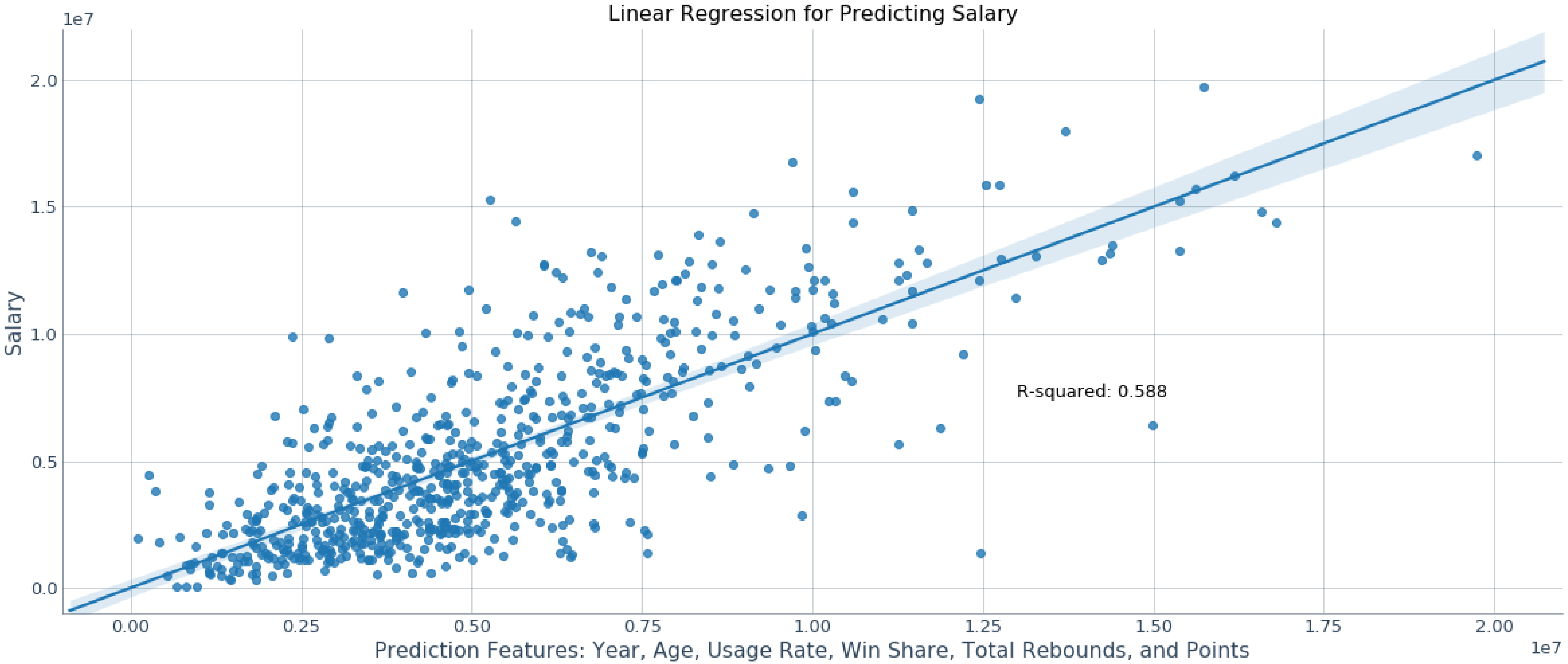
Machine Learning



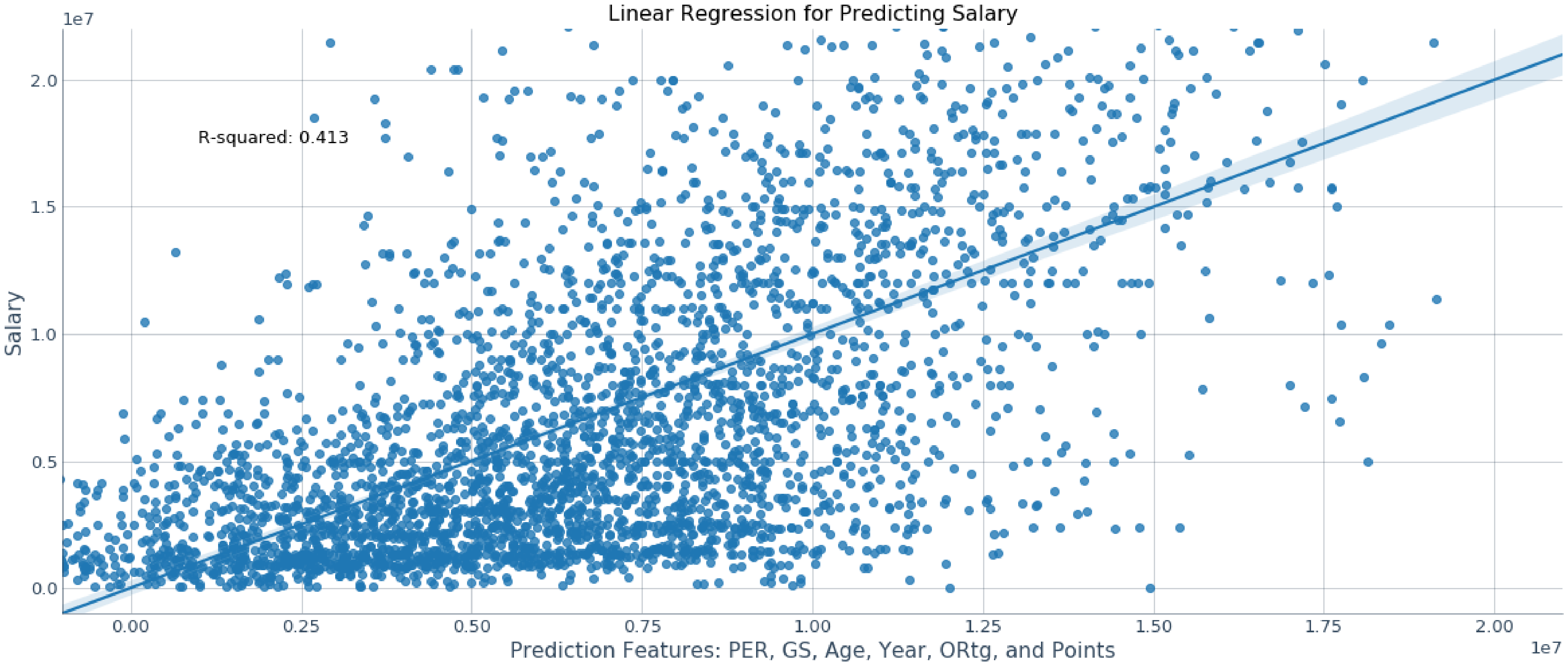
Machine Learning: Linear Regression

- Using ols from the scipy package, we were able to test many iterations of features
- The best model only produced an R-squared value of 0.588, but was built on a “groupby dataframe,” or a dataframe that was the average of each feature grouped by player
  - This cut down on the number of data points drastically and is helpful but less insightful
- The regression line on the following slide shows these grouped records; the slide after shows our attempts at applying that regression to the original data—not great

Groupby-DataFrame



Full Data



Linear Regression for Predicting Salary
1e7
2.0
1.5
1.0
0.5
0.0
Salary
R-squared: 0.413
0.00
0.25
0.50
0.75
1.00
1.25
1.50
1.75
2.00
1e7
Prediction Features: PER, GS, Age, Year, ORtg, and Points

OLS Regression Results

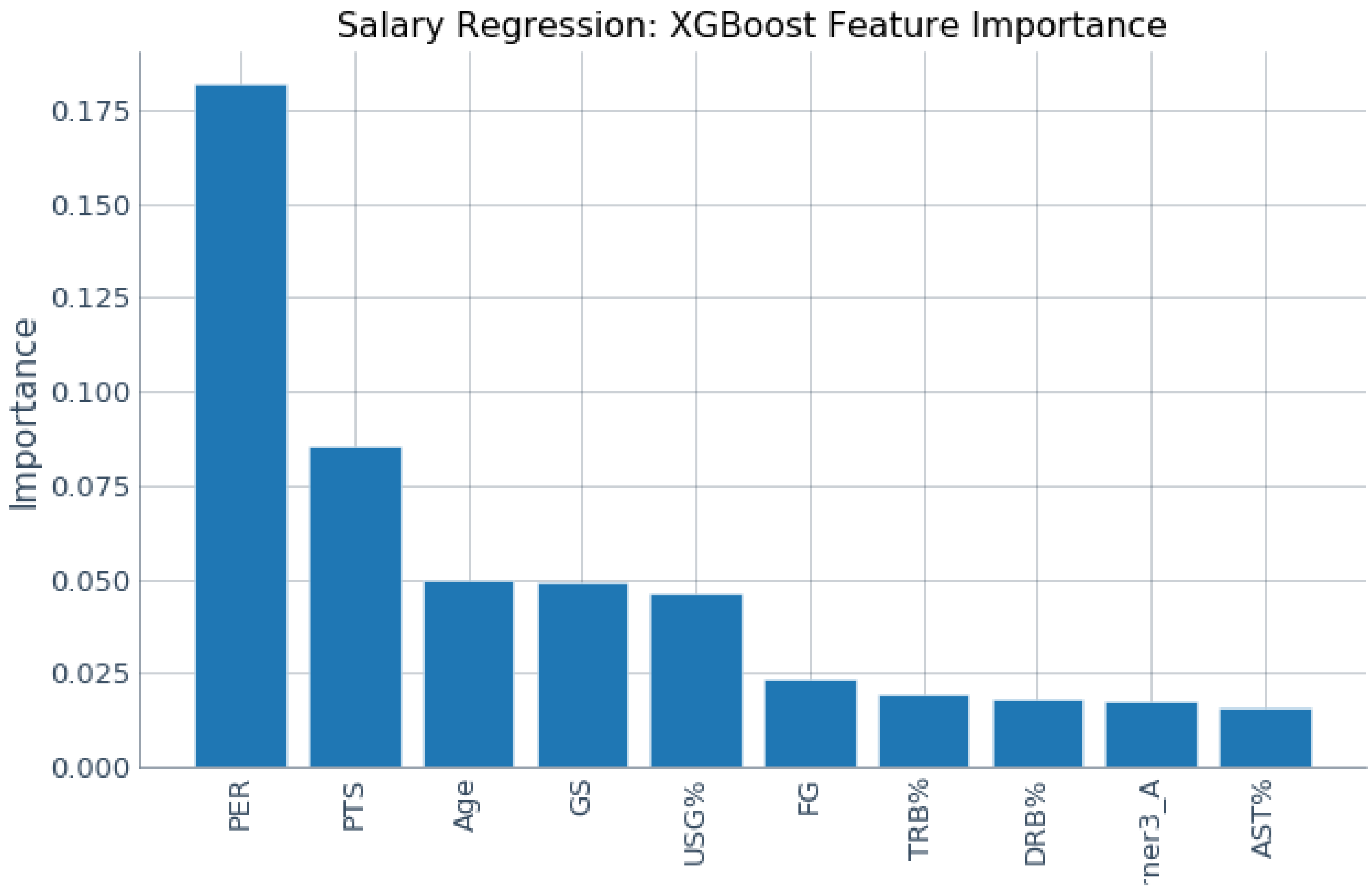
Feature	Coefficient	Std. Error	T	p-value
Intercept	-9.702e08	5.78e07	-16.795	0.000
Player Efficiency Rating (PER)	3.963e05	3.89e04	10.181	0.000
Games Started (GS)	5.401e04	3232.2	16.710	0.000
Age	5.091e05	1.91e04	26.689	0.000
Year	4.778e05	2.88e04	1603	0.000
Offensive Rating (ORtg)	-9.295e04	1.11e04	-8.409	0.000
Points (PTS)	2.29e04	2.29e04	7.995	0.000

- Our ORtg feature had a negative coefficient and probably some collinearity with PTS
- Other “value” metrics (advanced statistics such as VORP, BPM, and Usage) were put into the regression but did not help the model, only showing up with negative coefficients

- From the latter regression, we see just how difficult it is to apply a linear model to such varied data.
- If anything, this supports our underlying premise that players are paid on factors other than what they produce on the stats sheet.
- If we look towards the right of the plot, where stats are higher (read: better), we see there are points well below the regression line.
  - Such players would be easy to designate as “underpaid.”
  - Players above this line could also just as easily be considered “overpaid.”
- We also ran a Random Forest regression and an XGBoost regression and compared R-squared values across the three to find a best model. (next slide)

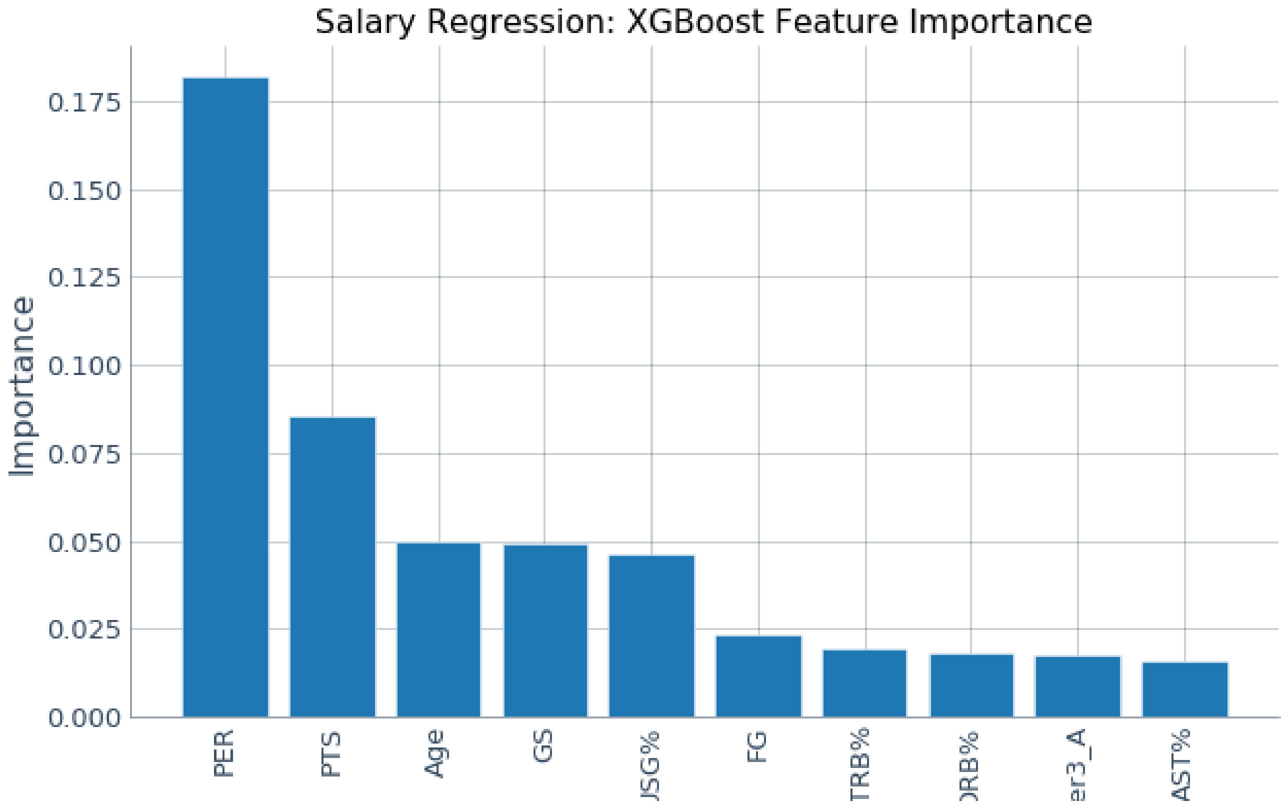
Comparing Regression Models

Regression	R-Squared
Ordinary Least Squares	0.41
Random Forest	0.51
XGBoost	0.52


Salary Regression: XGBoost Feature Importance
Importance
0.175
0.150
0.125
0.100
0.075
0.050
0.025
0.000
PER
PTS
Age
GS
USG%
FG
TRB%
DRB%
ner3_A
AST%

Understanding Player Efficiency Rating (PER)

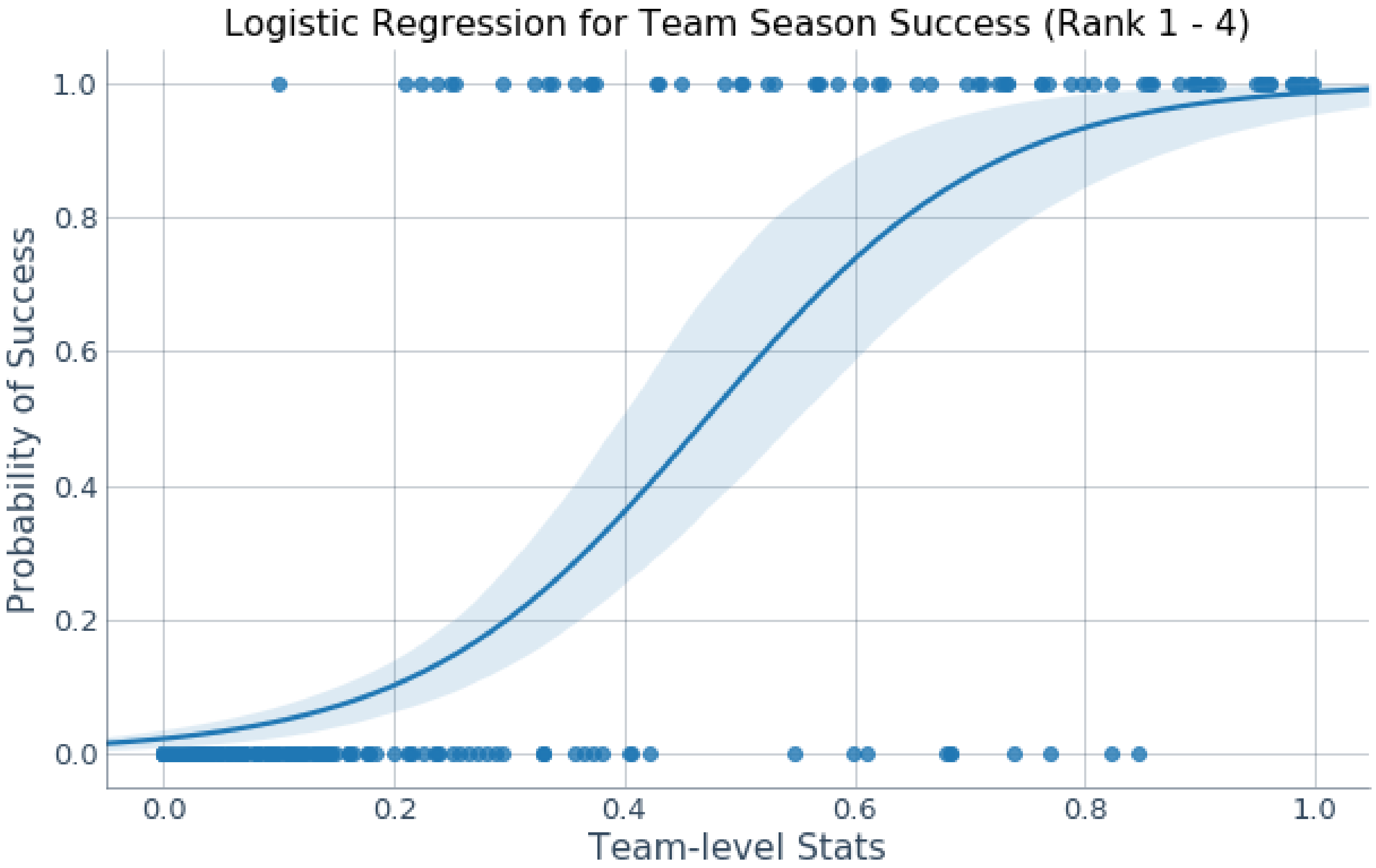
- PER strives to measure a player's per-minute performance, while adjusting for pace. A league-average PER is always 15.00, which permits comparisons of player performance across seasons.
- PER considers accomplishments, such as field goals, free throws, 3-pointers, assists, rebounds, blocks and steals, and negative results, such as missed shots, turnovers and personal fouls. The formula adds positive stats and subtracts negative ones through a statistical point value system. The rating for each player is then adjusted to a per-minute basis so that, for example, substitutes can be compared with starters in playing time debates. It is also adjusted for the team's pace. In the end, one number sums up the players' statistical accomplishments for that season.[source]
- PER is one of our better aggregate “value” statistics, but it potentially takes into account some statistics that aren’t as relevant to success as others. PER is more or less a measure of how well a player fills up a stats sheet, not a measure of whether a team has a better chance of winning.



Interpreting Regression Model Feature Importance

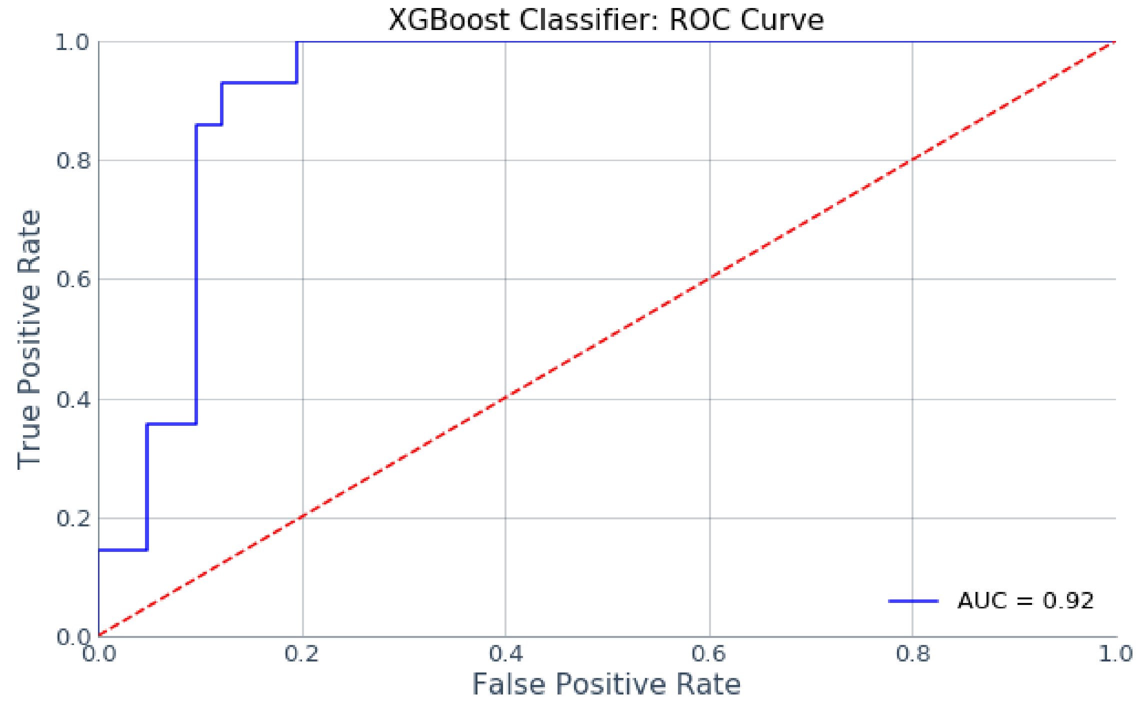
- Now that we understand PER, we can examine the other features by which player salaries can be predicted.
- Points in any sport is probably the loudest stat, understandably the most important
- Age, Games Started (GS), and Usage (USG%) are intuitive as well but don’t tell us anything about the player’s production
- FG made is comparable in importance to two rebounding metrics, “attempted corner threes,” and assist percentage.

Predicting Team Success (Classfiers)



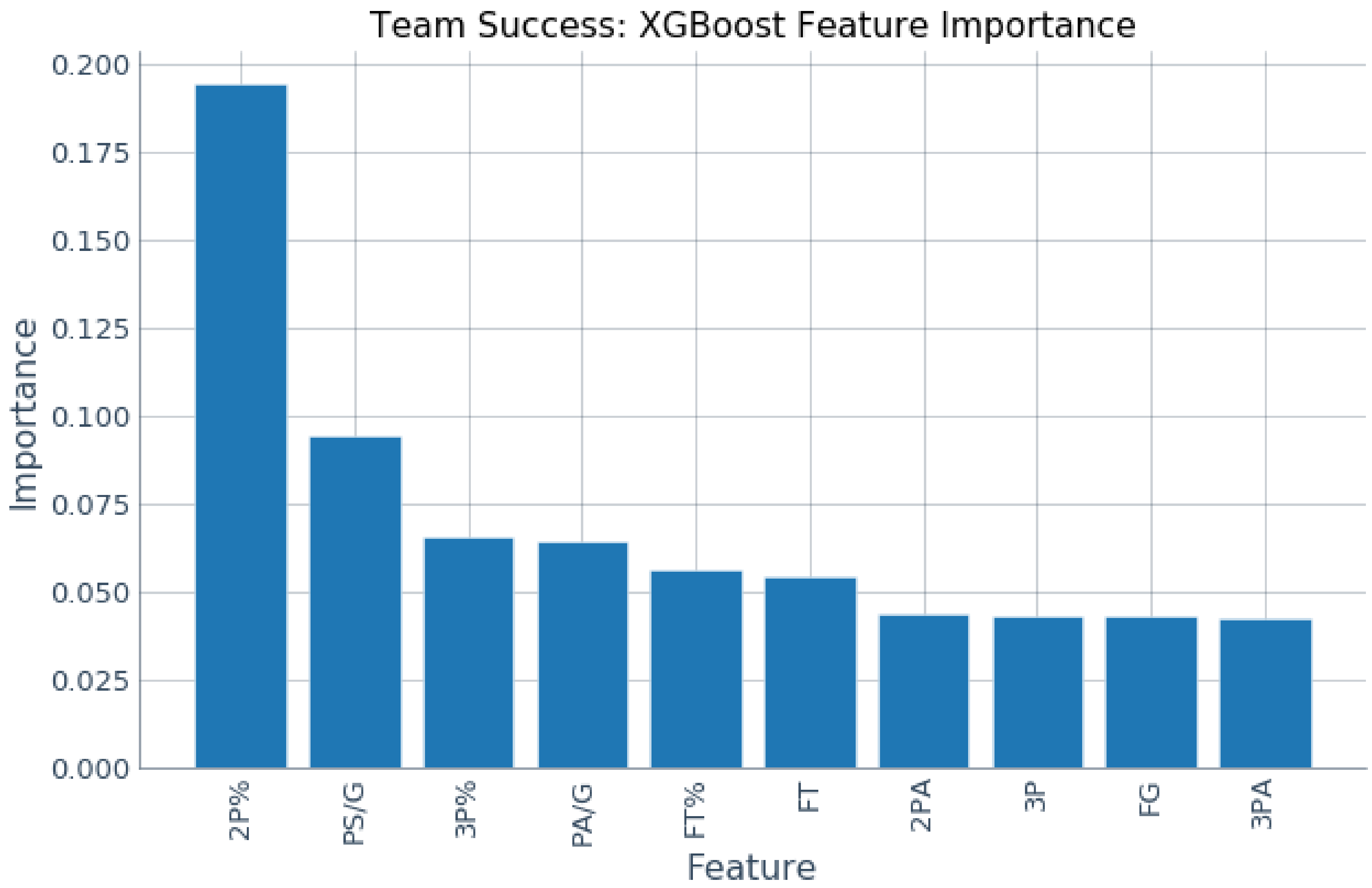
Comparing Classifier Models

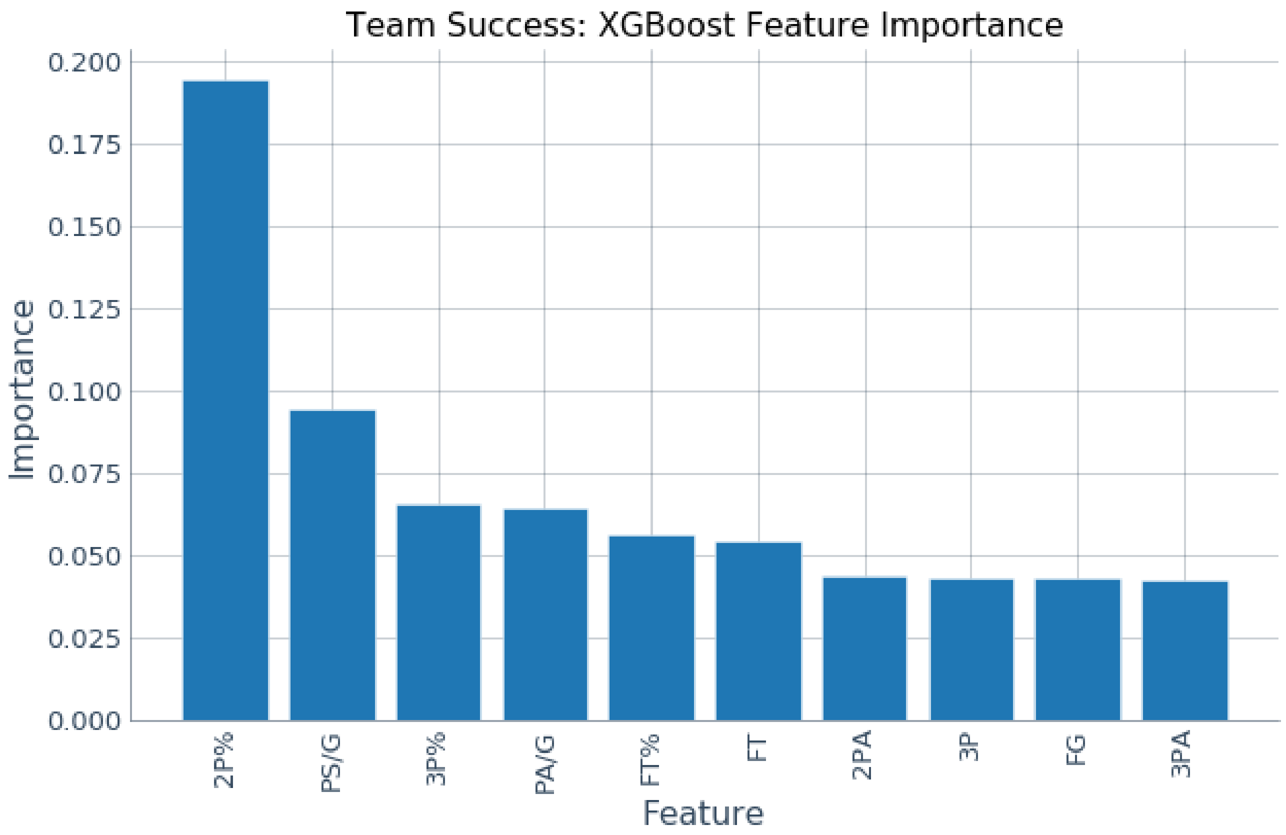
Classifier	Accuracy	ROCAUC
Logistic Regression	0.89	0.92
Random Forest	0.81	0.84
XGBoost	0.82	0.92



Assessing Our XGBoost Classifier

- With an accuracy of 0.82 and an AUC of 0.92, we have a strong model for predicting team success (defined as finishing the season ranked 1 – 4)
- This means we can interpret our feature importance plot as a legitimate ranking of which factors best help predict team success. (next slide)


Team Success: XGBoost Feature Importance
Importance
0.200
0.175
0.150
0.125
0.100
0.075
0.050
0.025
0.000
2P%
PS/G
3P%
PA/G
FT%
FT
2PA
3P
FG
3PA
Feature



Interpreting Success Classifier Feature Importance

- Like our regression models, we can examine feature importance for our classifiers; we'll look specifically our XGBoost model.
- 2-point accuracy is the heaviest influencer of the model, followed by Points Scored, and then 3-point accuracy, then Points Against.
- Importantly, there are 3 separate 3-point related statistics (3P%, 3P, and 3PA) in our top ten.
- Other than the very general PS/G and PA/G, every feature is a shooting statistic

Answering the Question: Who is Valuable?

- Three things stand out from our feature importance bar plot:
  - 2P shooters win games
  - 3P shooters win games
  - Players who make their free throws (FT) win games
- Very generally, a GM ought to pay the player who fits at least one of those. (practical advice on next slide)

Answering the Question: Who is Valuable?

- If you want to pay a big man (traditionally, a center or power forward), get one who isn't a liability when he goes to the free throw line.
- If you must pay a player who can't shoot threes, ensure that he takes high percentage twos.
- Otherwise, invest in three-point shooters.
- Defense and rebounding are much further down the priorities list.
- Don't spend money on a player because he has impressive blocks or outrebounds other players because he "wants it more" or is a "harder worker."
- Again, most simply, pay players who can shoot.

Answering the Question (cont.)

- If you want to pay a big man (traditionally, a center or power forward), get one who isn’t a liability when he goes to the free throw line.
- If you must pay a player who can’t shoot threes, ensure that he takes high percentage twos.
- Otherwise, invest in three-point shooters.
- Defense and rebounding are much further down the priorities list.
- Don’t spend money on a player because he has impressive blocks or outrebounds other players because he “wants it more” or is a “harder worker.”
- Again, most simply, pay players who can shoot.

Final Insights: Value vs. Worth

- Where are left is with a philosophical question of “Value” versus “Worth,” where the former can be defined as “the dollar amount a player contributes to his organization,” and the latter as “what someone is willing to pay for that player’s services.”

- Our regression model does a sufficient job of predicting player worth (though as we saw, there is still plenty of variance). If we can form a coherent method for assigning a dollar amount to a player’s contributions (i.e. his “value”) then we have a means of establishing if the player is over or underpaid.

- If the player’s value is greater than his worth, he is underpaid.
- If the player’s worth is greater than what he brings, his value, he is overpaid.

Conclusions and Next Steps

- Certain clusters are probably more valuable than others, objectively, meaning our LDA and clustering was at least worth it in that regard.
  - Since we know what features are strong predictors of team success, we should want to know which clusters test the strongest for those features.
- Inferential statistics showed us that, at least at the 120% of the salary cap mark, spending more doesn’t result in significantly more wins than spending less.
  - This should mean that we have room to operate, even on a “tighter” budget
- In order to establish whether a player is overpaid or underpaid, we need a coherent and intuitive algorithm for assigning value to a player’s stats. We can then weigh that against what that player is/was actually being paid (his worth).
- The differences in feature importance between what predicts player salary and what predicts team success tell the story of my hypothesis (i.e. winning games is based on a set of features different from the features on which salaries are based.)
- See the full code in [this GitHub repository](#).