Becoming an NBA All-Star: Ball Game or Social-Media Game

ELE 381
Final Report - May 2019
Oliver Schwartz, Matthew Nicholas

Our GitHub repository lives here.

Abstract

This project is our attempt to build a unique prediction engine to forecast the results of the NBA All-Star fan vote. To the naive observer, the player who wins the All-Star vote is probably the highest scoring player. There is some validity in this intuition - raw performance undoubtedly plays a part. However, what other factors influence this order? How can we weight these factors to determine the result?

Our investigation into the NBA All-Star voting system was prompted by several of its incongruities - most notably, Georgian native Zaza Pachulia breaking the top 10 in fan vote the 2016-17 season (despite having played in less than 20% of regular season games and at no point ever being considered in the top 10 NBA players). The opposite phenomenon is also something frequently observed: high performing players often don't receive fan vote totals in proportion to their statistical output.

Clearly, factors beyond pure performance can sway the fan vote. Our fundamental insight in explaining discrepancies in the NBA voting system is social media: the influence of a given player on social media platforms plays a significant role in determining their position on the fan vote. Additionally, our prediction engine builds on the intuition that we loosely model mutual respect among players by constructing a network consisting of only players - this network model can also hold predictive power for determining the player vote (where players vote on one another).

Contents

Chapter 1

Introduction and Motivation

The NBA All-Star game is a yearly exhibition match that showcases the league's top talent. The All-Star game has undergone immense changes since its inception in 1951 in terms of its rules of play, the player selection process, its popularity, and the accompanying events that go along with the exhibition. One thing that has stayed constant throughout all the changes is the pedigree associated with being selected for the All-Star game. It is considered an immense accomplishment to be selected, and every player aspires to one day be recognized as an "All-Star" amongst the best players in the world. All-Star appearances are often listed next to championship victories, All-NBA teams, MVPs, and Defensive Player of the Year awards as the most important accomplishments of a player's career. Furthermore, the number of selections becomes an important factor in determining whether a player is worthy of the Basketball Hall of Fame.

The All-Star game is supposed to include the best NBA players from each year, but every year some deserving players are left out, while other players who have had seemingly worse statistical seasons are selected. This is largely due to a selection process where fans have a large say in who is selected for the exhibition. In fact, up until 2016 the five starters for each conference were selected strictly by a fan vote.

Coaches then voted to determine the reserves to complete the rest of the two team rosters. This led to some unexpected results: in 2016 Zaza Pachulia received the fourth most votes, despite being ranked by ESPN as the 157th best player in the league. This prompted the NBA to change their selection process. Instead of starters being chosen strictly by fans, players were selected by weighting the fan vote 50%, the coach vote 25% and the player vote 25%.[2] The aim for this change was to reduce the number of inferior players from being selected to the all-star game.

The case of Zaza Pachulia in 2016 is particularly interesting, as his sharp increase in votes was not due to any significant improvement in statistics. In fact, 2016 was only a slightly above average season for Zaza, as he increased his scoring average only 0.3 points from the previous season to 8.6 points per game. This is well under his career high in 2006 where he averaged 12.2 points per game. Upon closer inspection, it turns out Zaza's high fan vote totals are largely a factor of a massive social media campaign. The president of Georgia, Zaza's native country, urged all its citizens to vote for Zaza in the All-Star game. Additionally, Internet personalities and musicians such as Hayes Grier and Wyclef Jean encouraged followers to vote for Zaza. These social media campaigns were particularly effective since fans could vote multiple times.

Cases like this, with seemingly inexplicable voting patterns, motivated us to build a model that predicts the number of All-Star fan and player votes each player will receive. One of the features for this model that we developed is a player's "importance score", which is designed to account for the large role player's social media influence has over the voting outcome. The purpose for this project is to build the most accurate predictive model, and analyze the predictive power that various social media metrics have, including our "importance score", in determining a player's All-Star fan vote and player vote totals.

Chapter 2

Importance Score

2.1 PageRank and Instagram

Before building the predictive model for determining the number of All-Star fan and player votes that each player receives, we first needed to create features that would help factor the social media presence of a player into each models prediction. We decided to create two features: the ”importance score” for each NBA player, and the raw number of followers each player has. Initially, we settled on calculating the ”importance score” by applying Google's PageRank algorithm to the graph of social media connections between players. In the later stages of our project, we decided to broaden our interpretation of importance, utilizing various other network metrics covered in class, including load, betweenness, and eigenvector centrality. These importance scores, used alongside the raw number of followers, give a good idea of a player's online presence and popularity in comparison to their NBA peers.

There are several social media platforms that would provide the data necessary to compute an importance score using PageRank for each NBA player - the most obvious choices being Facebook, Twitter, Instagram, and Snapchat. Each of these platforms represent player/fan relationships in a different way: Facebook and

Snapchat include both a ’friends’ construct as well as a ’following’ construct, while Twitter and Instagram only allow users to follow another user. This leads to different graphical representations between users in the respective social networks: friendship is bidirectional (A is friends with B, so B is also friends with A), and thus friendship graphs are undirected. On the other hand, ’following’ in a social network generates directional graphs: if A follows B, there is a directed link from A to B.

We decided to use Instagram for several reasons. The primary reason was data aggregation: following the Cambridge Analytica scandal of 2014, many social media platforms (most notably Facebook) restricted their APIs, making it far more difficult to programmatically obtain masses of data (i.e. data-scraping became much harder). However, we stumbled upon a Chrome extension which enabled us to download CSVs of all the users that a given player follows. In addition, most NBA players have Instagram meaning our dataset would be very close to complete.

Using Instagram as our data source, the network we used to generate node importance scores was a directed graph. We formally define this network as consisting of:

$$\text{A Directed Graph, } G = (V, E)$$

$$\text{Where each player is represented by a vertex, } V$$

$$\text{Where an edge, } E\text{, connects vertex A to B, if A follows B}$$

Note: our network consists only of players following other players on Instagram. Many NBA players are followed by millions of people, and our network would become unwieldy if we included all users who follow them.

In addition to PageRank, there are several ways to measure node importance in this network: eigenvector centrality, load centrality, and betweenness centrality are some obvious choices. It is not immediately apparent which of these metrics is most appropriate for our purposes. They capture different characteristics of a network: load and betweenness centrality reflect how often a node appears on shortest paths, and this value may differ from eigenvector/PageRank importance for popular players who do not follow many of their NBA peers. Because of this, our final prediction model makes use of several different metrics to determine which best represents the influence of a player on social media. Load and betweenness centrality reflect how crucial a node is to the dissemination of information through a network - hence, these measures are more intuitively understood in networks with flow of information or feedback loops (like transport networks or computer networks). Nevertheless, we include these measures of node importance to compare their predictive power to metrics like PageRank importance and eigenvector centrality.

PageRank is an iterative algorithm which uses negative feedback to determine the importance of all nodes in a network: it uses iterative multiplication of an importance vector (where each component represent the current importance of a node) by a matrix, whose entries are based on the in and out-degrees of given nodes. At each iteration of the algorithm, the output of the previous stage (the importance vector) becomes the input at the next stage. As long as the network has no dangling nodes (nodes that do not point to anything), this algorithm will converge to a stable state, yielding a final importance vector. PageRank is not a perfect model, but it balances simplicity with usefulness.

2.2 Methodology

There were several significant components involved in gathering our data. We accumulated:

1. **All-Star fan and player votes**

2. **Season-by-season and career statistics**

3. **Instagram data - raw follower count, and a list of all followers**

1. All-Star votes

All star fan and player votes are available on `basketball-reference.com` for the 2017, 2018, and 2019 All-Star games (16-17, 17-18, 18-19 NBA seasons). (As a result, our prediction engine can only be trained on these three seasons and any subsequent season where All-Star votes are available for every player. However, the engine will work on previous seasons but will not be able to compute an error function because there is insufficient voting data). After saving each webpage locally, a series of shell (.sh) and Python scripts would run to parse each HTML file and generate a series of CSVs listing (player, votes) for a given season. These files are available in the allstar-votes directory in the project repository.

Our fluency in Unix was particularly useful here. Packages like Python's BeautifulSoup are helpful, but often require more code and reading of documentation. Unix is a particularly pure way of programming, and comes equipped with practically every tool for string processing one could ever need. In particular, we used grep, sed, and sort extensively. Unix also makes reading and writing far cleaner than any scripting language. Often, our processing scripts would redirect to 10 or more temporary files as successive stages of processing took place. Finding the right tool for the task was critical, and gave us time to focus on our data analysis and machine learning models.

2. Season-by-season and career statistics

The majority of this data was downloaded from `kaggle.com`. However, we were later to discover that this dataset was somewhat incomplete: certain seasons were missing a GS ('games started') statistic which required us to find this ourselves. Fortunately, we discovered the Kaggle dataset had been scraped from `basketball-reference.com`. To rectify this missing stat, we implemented a similar system to the aforementioned HTML parsing - a series of shell and Python files. This was surprisingly challenging because it required many regular expressions and multiple redirections (piping). After many refinements, one particular shell file involved a regular expression for a player's name (which caught every player except for Nene - listed by only his first name):

$$[\text{A–Z}]\,.*[\text{a–z}]+[\text{A–Z}]*[\text{a–z}]*.*\,,\quad [\text{A–Z}]\,.*[\text{A–Z}]*.*[\text{a–z}]*.*$$

The Kaggle data is located in the kaggle directory. The logic to acquire the GS statistic is located in kaggle/gs.

3. Instagram Data

After having accumulated a complete set of 683 players, we had to manually search for them on Instagram and save their usernames.

Our instagram data consisted of two components: obtaining a list of every user a given player follows, and obtaining a raw follower count for every player. The first part involved a predominantly manual process due to the restrictions Instagram has placed on its API: specifically, we used a Chrome extension to generate a CSV of users that a given player follows - `instascraper.weebly.com`. We managed to obtain this data for approximately half of the number of players. (The fraction we obtained were primarily the better players in the league, the intuition here being that the bottom half of the league receive little to no All-Star votes anyway, so omitting them will in fact boost the accuracy of our modelling).

The second part of our data was a raw follower count for each player (this data and necessary logic is contained in ig/scripts/count). The aforementioned Chrome extension did not provide us with this information - surprising, given it was able to generate a complete list of every user a player follows. We did this programmatically with some reverse-engineered workarounds. In particular, Instagram is able to detect bots scraping pages, so returns 400 HTTP status codes (Bad Request) for such requests. By tweaking a Python script to imitate a browser, we could peruse Instagram as we pleased. This involved the following request header:

```
'User-Agent': 'Mozilla/5.0 (Macintosh; Intel Mac OS X 10_10_1)
AppleWebKit/537.36 (KHTML, like Gecko) Chrome/39.0.2171.95
Safari/537.36',
'cookie': '...'
```

We obtained the cookie by logging into Instagram and using Developer Tools to copy the cookie for our session.

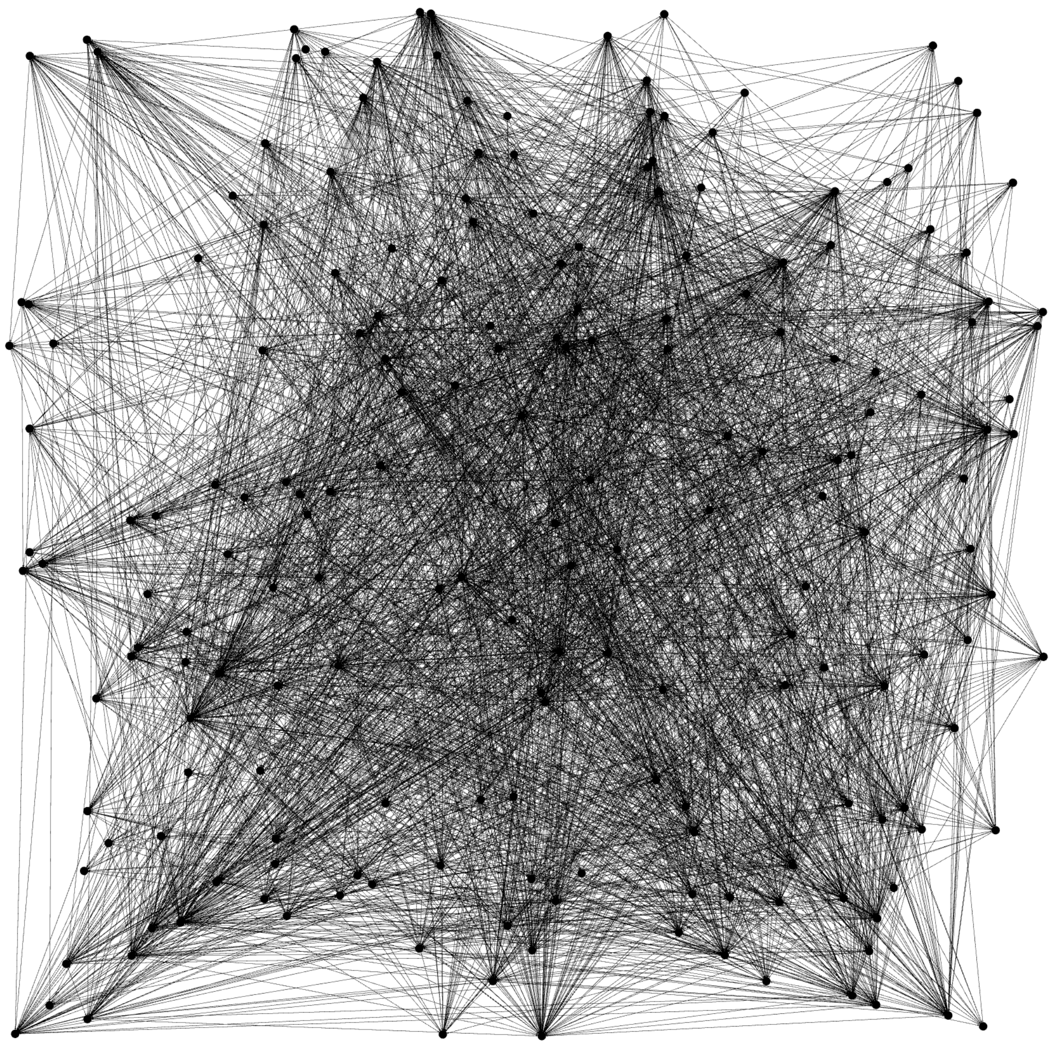
Google Trends

Following the submission of our midterm report, we pondered how to source another popularity metric from Google Trends. We found a useful repository which interfaced with Google Trends in order to automate our data collection (we actually implemented a system to pull Trends data - it lives in trends/). However, we realized that this would have little to no predictive power given the nature of Google Trends data:

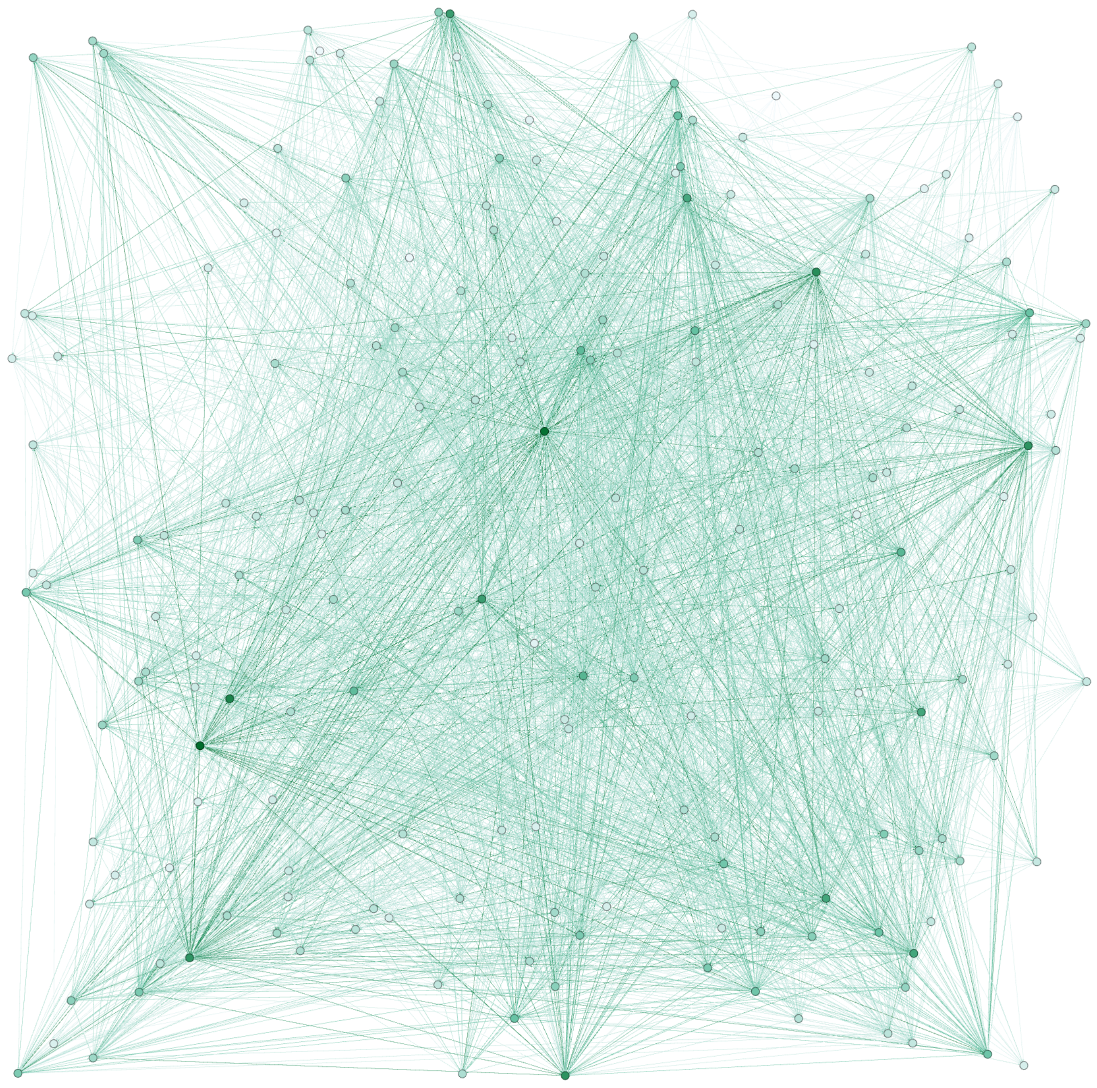
instead of a raw search count for a given time period, Trends gives search queries a score between 0 and 100 for a given time frame. This score reflects search frequency for a given term only relative to other search activity for that particular term - so ”Lebron James” could have a score of 100 for a given week and a raw search count of 1,000,000, while ”Aaron Brooks” could also have a score of 100 for the same week with a raw search count of 1,000.

2.3 Results

The logic to construct the network and compute the various network metrics we used is included in the Jupyter notebook (attached, also located in ig/playerFollowing/scripts/PageRank.ipynb). This is the network we generated (visualized in Gephi):



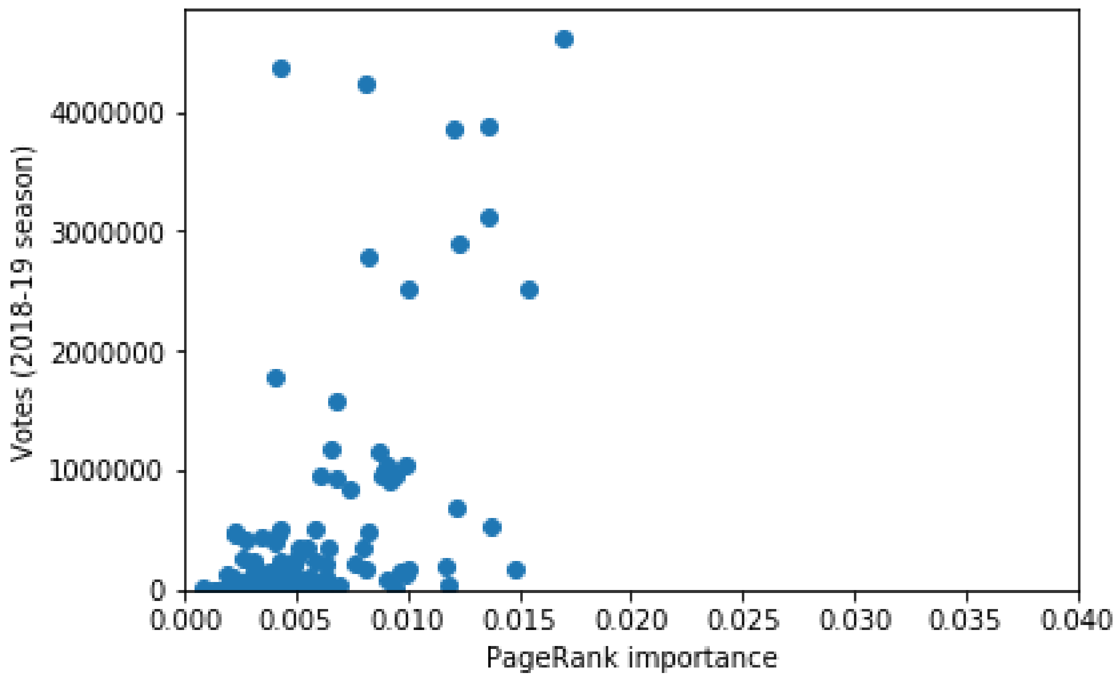
This network consists of: 198 nodes (players) and 4831 directed edges (although running Gephi on Linux distributions has problematic rendering so arrow heads are not displayed). Visualizing the same network, but instead colouring nodes in proportion to their in-degree generates:



Running PageRank on the network generates the following PageRank importance vector (from most to least important):

$$\pi = \begin{bmatrix} \text{Dwyane Wade} & 0.01727266641259745 \\ \text{Lebron James} & 0.016931067265492542 \\ \text{Russell Westbrook} & 0.01546099115290284 \\ \text{John Wall} & 0.014786288711603875 \\ \text{Kevin Durant} & 0.013960864319024541 \\ \text{Chris Paul} & 0.013752153498905602 \\ \text{Paul George} & 0.01358760213760087 \\ \text{Kyrie Irving} & 0.013572160071044495 \\ \text{James Harden} & 0.012240694876779892 \\ \text{DeMarcus Cousins} & 0.012193076851066683 \\ \text{Stephen Curry} & 0.012064110048031469 \\ ... & ... \end{bmatrix}$$

This indicates correlation between importance and All-Star votes - we would reasonably expect LeBron James, Russell Westbrook, and Kevin Durant to rank in the top 5 as they are indisputably some of the best players in the league. To quantify this, we plot importance against All-Star votes (for the 2018-19 season) and obtain this scatter plot:



Analyzing some of the other metrics generated from this network...

Eigenvector centrality:

$$\begin{bmatrix} \text{Dwyane Wade} & 0.2103654455823743 \\ \text{Lebron James} & 0.19441727320545832 \\ \text{Russell Westbrook} & 0.18393681362341363 \\ \text{John Wall} & 0.18282280494787723 \\ \text{Kyrie Irving} & 0.1824346538382556 \\ \text{Paul George} & 0.18014057828437657 \\ \text{Chris Paul} & 0.1736676671393028 \\ \text{Kevin Durant} & 0.17154174241414458 \\ \text{DeMarcus Cousins} & 0.17016345798943378 \\ \text{Carmelo Anthony} & 0.16599525847580734 \\ \text{James Harden} & 0.16472430349762554 \\ \text{J.R. Smith} & 0.1487062617284237 \\ \text{Isaiah Thomas} & 0.1483130183758508 \\ \dots & \dots \end{bmatrix}$$

Load centrality (the proportion of all shortest paths that pass through a given node):

$$\begin{bmatrix}
\text{Dwyane Wade} & 0.03832656150338334 \\
\text{Russell Westbrook} & 0.02571773129680718 \\
\text{Paul George} & 0.025629736428758317 \\
\text{Chris Paul} & 0.024248117697104082 \\
\text{Lance Stephenson} & 0.02321393174663186 \\
\text{Kevin Durant} & 0.022926307833480274 \\
\text{John Wall} & 0.021625412513498938 \\
\text{Stephen Curry} & 0.019788505280278982 \\
\text{Marc Gasol} & 0.017990127560324105 \\
\text{Victor Oladipo} & 0.01747634916087864 \\
\text{Rudy Gay} & 0.017340697172921398 \\
\text{Kyrie Irving} & 0.01690026434314827 \\
\text{Isaiah Thomas} & 0.016805578337029888 \\
... & ...
\end{bmatrix}$$

Betweenness Centrality (the extent to which a given vertex lies on shortest paths of other vertices):

$$\begin{bmatrix}
\text{Dwyane Wade} & 0.040449224593822906 \\
\text{Paul George} & 0.027683167027070587 \\
\text{Russell Westbrook} & 0.027216204007552473 \\
\text{Chris Paul} & 0.02642615100666629 \\
\text{Kevin Durant} & 0.0238044074627439 \\
\text{Lance Stephenson} & 0.022979075033234497 \\
\text{John Wall} & 0.02257730833445652 \\
\text{Stephen Curry} & 0.020297285688550718 \\
\text{Rudy Gay} & 0.018344985592303693 \\
\text{Anthony Davis} & 0.018217500537323714 \\
\text{Victor Olapido} & 0.01799599705955952 \\
\text{Kyrie Irving} & 0.017729102372486136 \\
... & ...
\end{bmatrix}$$

2.4 Analysis and Conclusions

The results above indicate that importance score has some sort of predictive power. There is a rough linear trend in the scatter plot: the more important the player, the more votes they will get. (This plot is not definitively linear: we obtained similar R^2 values fitting both logarithmic and quadtric curves to the scatter plot.) However, as we do not have a complete dataset, there is a large cluster of relatively unimportant players clustered around 0 votes. There are also several outliers: either players who aren't very important but have a high number of votes, or players who are somewhat important and do not have many votes.

All 3 additional network metrics (eigenvector, load, and betweenness centrality) roughly preserve the relative order observed in the PageRank importance - this is likely because the network is very dense and interconnected. One interesting point of note is that Lebron James' load and betweenness centrality ranks him far lower (out of the top 10) than the PageRank and eigenvector centrality. This is promising as it indicates that these metrics have some nuance. Lebron performs well in PageRank and eigenvector centrality because he has many other important players following him. However, for betweenness and load centrality, he does not follow many other players back so there are few shortest paths to other players that pass through him.

In contrast, some older players like Rudy Gay and Marc Gasol make an appearance in the top 15 or so in load and betweenness centrality because they form a glue within the network - connecting subgroups of players that are otherwise unreachable.

There are some limitations to these generated importance scores. The most obvious drawback is that some NBA players do not have Instagram, yet are still popular among fans and players alike. For instance, Kawhi Leonard (known in the NBA as *the claw*) does not have an Instagram account but is a 3x all-star, 2x defensive player of the year, and has 3 all-NBA team selections. As a result, when implementing our predictive model, our "importance score" and raw followers features will have some missing data. Additionally, we are only able to generate a single importance score instead of an importance score for each season - Instagram does not store snapshots of user following/followers as this would require an inordinate amount of memory. As a result, the importance score only reflects a players status at current times. This may bias our prediction engine to predict higher All-Star fan votes for a player who has a substantially larger social media presence now than in the 2016 season, the first season that we will use to train our models. This will be particularly noticeable for players who entered the league in the past 3 years, as well as players who made a significant jump in performance in the past 3 years, as these events tend to coincide with a corresponding spike in followers.

Chapter 3

Predictive Model

3.1 Analytics Base Table Generation

The first task in developing a predictive model is to create an Analytics Base Table (ABT). The ABT is a structure which is organized so that each row is an instance for which the model makes a prediction for.[1] In the case of our model, each row is a season for a player for which we wish to predict the number of fan All-Star votes received. Each column is a descriptive feature which is filled out for each player. In our model this includes various statistics and metrics such as points per game, rebounds per game, number of games started, our generated importance score, Instagram followers, and more raw and derived features (raw features are a direct stat while derived feature combines multiple raw features to create a new feature[1]). The last column in the ABT is the "target feature", or the feature that the model is making a prediction for. Since we wanted to predict both the number of All-Star fan votes received and the number of player votes recieved for a given player, we have two target features. For the first model this is the number of All-Star fan votes received. For the second model this is the number of player votes received. Our ABT can be found in "ABT.csv" and "ABTpvotes.csv" in the kaggle/ directory in our project repository. We generated the ABT by writing a Python script (”generateABT.py”) that read in the database of player statistics, social media metrics, and fan/player vote totals from different sources within our project repository, deleted unneeded data, formatted the data in the proper way, and then wrote the data to ABT.csv. ”generateABT.py” primarily uses the pandas module to perform data operations and formatting.

3.2 Feature Selection

In total, 64 features were generated in the ABT for input into the machine learning models. However, too many features can reduce the accuracy of certain types of machine learning models (especially similarity based models).[1] As a result, 11 features that were deemed most important were used as input into the models. These 11 features can be separated into 3 different categories:

- Health and Playing Time - this includes the features ’G’ (games played), and ’MPG’ (minutes per game). The ’G’ feature is important because it accounts for players that are injured. If a player is injured in the fourth game of the season it is possible that they still have very high ’per game’ stats, but it is unlikely that they will receive All-Star votes with playing so little of the season. The ’MPG’ feature is important since the best players (who often receive the most votes) usually play the most minutes per game.
- ’Per game’ statistics- ’PPG’ (points per game), ’RPG’ (rebounds per game), ’APG’ (assists per game), ’BLKPG’ (blocks per game), ’FG%’ (field goal percentage), ’WS’ (win share), ’VORP’ (value over replacement player). These are the most common and important basic and advanced stats that fans, players,

and analysts look at to evaluate players.

- Social Media Metrics- 'Followers' and 'PageRank'. 'Followers' corresponds to the number of Instagram followers and 'PageRank' corresponds to the importance score generated by applying Google's PageRank algorithm to the network described in Chapter 2 Section 3.

3.3 Models

The target features for both desired models are continuous. This means that the models predict a value, rather than a category. Regression models are the machine learning models which predict a value, while classification models predict a category. As a result only regression models are used for both models. The scikit-learn machine learning module was downloaded in order to use the embedded algorithms that come with it. 'RidgeRegression', 'Lasso', 'ElasticNet', 'K-NN Regressor' (k nearest neighbor), and 'RF regressor' (random forest) were used. The explanation of how each model exactly works is not included in this report due to space constraints, and the fact that the implementation of these algorithms is not part of the material for this class.

3.4 Sampling

Next, the instances in the ABT need to be separated into a training set and a testing set. The training set is used to fit each model to the unique data. The test set is the remainder of the data that is fed through the model and is used to evaluate the performance of the model with data that it hasn't seen before. There are a number of methods for separating the instances in the ABT into a training set and testing set. Random sampling separates a selected portion of the dataset into a training and test set at random. Stratified sampling separates a selected portion of the dataset into the training and test set by keeping the relative distribution of a particular descriptive feature consistent. For instance, if there are 10 girls and 90 boys in a dataset, and the test set is set to be 10%, then there will be 1 girl and 9 boys in the test set.

However, for the purpose of this model, it made sense to do a custom sampling method, where an entire years worth of data is left out. This way predictions are made for all the player in a particular season. This is how the model would be of most practical use: predicting vote totals in seasons where the all-star results are still unknown. There are only three seasons worth of data in the ABT (as outlined in Chapter 2 section 2). Before the predictive models are trained, a year is selected to be held out for the test set. For instance, if 2019 is selected, the model will be trained on data from 2017 and 2018, and then make predictions for the 2019 season.

3.5 Results

The following is a table summarizing the results for the fan vote totals by averaging the score for each metric for when the three different seasons are held out separately. "R2 CV" is the R-squared cross validation score. This gives a measure of how well the model is fit to the data (0 being lowest performing model and 1 being most accurate). By using cross validation, it ensures the score is accurate and not influenced by over-fitting. "SW" is the Shapiro-Wilk test. This test returns a "W" value between 0 and 1, with 1 meaning that the error in the model is distributed perfectly normal. It also returns a "p" value. A "p" value of less than 0.05 means that we reject the null hypothesis that the error is normally distributed. "DW" is the Durbin-Watson test. This returns a value between 0 and 4. It measures auto-correlation, which is when the model makes the same mistake over and over again (if one point is higher or lower than the previous, then the next point will be even higher or lower).

Table 3.1: Model Performance

Model	R2 CV	SW-W	SW-P	DW
Ridge	0.58	0.71	< 0.01	1.88
Lasso	0.60	0.72	< 0.01	1.99
Elastic Net	0.60	0.82	< 0.01	1.90
K-NN	0.61	0.83	< 0.01	1.90
RF	0.71	0.81	< 0.01	1.89

The R-squared cross validation scores are somewhat low, but that is expected given the nature of what we are predicting. Other models have similar r-squared values when predicting the MVP vote, and have performed well, so we can expect our models to be reasonably accurate. The error in our prediction models are random (which is good), but are not normally distributed, since our 'P' value is below 0.05 for all our models. This means that none of the errors resulting from the models are distributed normally. While this is not ideal, it does not mean that our model can't be trusted. All of the models have a DW value close to 2, so there is no auto-correlation. The following is ranking of the players with the most all-star fan votes from the RF model for the 2018 season, and the actual results of the all-star fan vote:

Rank	RF order	Actual order
1	LeBron James	LeBron James
2	Stephen Curry	Giannis Antetokounmpo
3	Kevin Durant	Stephen Curry
4	James Harden	Kevin Durant
5	Kyrie Irving	Kyrie Irving
6	Giannis Antetokounmpo	Manu Ginobili
7	Russell Westbrook	James Harden
8	Joel Embiid	Joel Embiid
9	Dwyane Wade	Russell Westbrook
10	Demarcus Cousins	Klay Thompson
11	Anthony Davis	Draymond Green
12	Chris Paul	Kristaps Porzingis
13	Klay Thompson	Anthony Davis
14	Ben Simmons	DeMar DeRozan
15	Blake Griffin	DeMarcus Cousins
16	Paul George	Paul George
17	Lonzo Ball	Kevin Love
18	Damian Lillard	Kawhi Leonard
19	Jimmy Butler	Ben Simmons
20	Demar Derozan	Victor Oladipo
21	Draymond Green	Dwyane Wade
22	Kyle Kuzma	Carmelo Anthony
23	Kristaps Porzingis	Lonzo Ball
24	John Wall	LaMarcus Aldridge
25	Devin Booker	Chris Paul

The RF model performs quite well for how difficult and seemingly random the actual order of players is. While the order is accurate, the raw number of votes have a large amount of error. This is largely due to the change in number of total votes from year to year - there is no upper limit on fan voting. For instance, in 2017 the top vote getter received 1.9 million votes, while the top vote getter in 2019 received 4.6 million. As a result, we changed the the target feature from total number of votes to a players vote share (percent of total votes). This resulted in an average R-squared score of 0.66, significantly higher than the average of 0.61 previously.

The different machine learning algorithms attained similar scores (R-squared average of 0.63) when we predicted for the number of player votes, rather than fan votes for each player. A large number of additional tests and analyses could be performed in order to analyze just how accurate each of these models is. However, since this is not a machine learning course, we will not further analyze exactly how trustworthy these predictive models are. Instead, we will examine the effect that the social media metrics have on its output.

3.6 Social Media Metrics Contribution

When the raw number of followers and the PageRank importance scores were removed from the model the R-squared score dropped to 0.44 for the fan vote predictor and 0.42 for the player vote predictor. This is significantly down from when the features were included in the model, which shows the these metrics have significant predictive power for both models. When just PageRank importance was removed from the fan vote predictor, and the number of followers was included, the R-squared average was 0.62 (which is close to how it performed originally). However, when PageRank was included, and followers removed, the R-squared average dropped to and average of 0.46. This makes intuitive sense, since the raw number of followers likely has a larger influence on the number of fan votes received in the all-star game, since most followers on Instagram are fans of the players (there are roughly 600 NBA players, and hundreds of millions of fans). The reverse is true for the player vote predictor. When PageRank was included and followers removed, the model scored much higher than the reverse (0.57 vs 0.53). This makes sense since the importance score we generated from PageRank relies solely on data about which players follow each other. A high importance score reflects a player who is followed by a lot of other important NBA players, which likely means they have the respect of many of their peers. This likely results in votes from other players.

The effect of these social media metrics can be clearly seen in the case of Dwayne

Wade. This season he received the 14th most fan votes. However, a model based upon raw performance would have put him much farther down in the rankings. In fact, when both social media metrics were removed from the model, it predicted him having the 46th most fan votes. However, our model is able to account for the social influence of a player, and successfully places Dwayne Wade in the correct vicinity, receiving the 12th most votes.

3.7 Model Shortcomings

While the addition of the social media metrics can clearly be seen through an improvement in the statistical accuracy of both predictive models, there are still factors for which the models can't account for. For example, "narrative", voter fatigue, and expectation are not built into this model. For instance, Manu Ginobili retired at the end of 2018. During that season he saw a massive spike in the fan vote and finished fourth overall, despite averaging only 8 points and 3 assists per game.

Additionally, the social media metrics generated do not always encapsulate actual importance and popularity. Some very popular players have Instagram, but are extremely inactive on the platform which results in less followers and importance - users are unlikely to follow a player who does not post content regularly. Moreover, viral trends do not necessarily correspond to a huge increase in followers. A large social media campaign was used to urge fans to vote for Zaza Pachulia. However, he never really received a large social media following, despite finishing in the top five of the fan vote. This could be because his Instagram account is relatively inactive, or because people thought a social media campaign to vote for Zaza is funny and were willing to vote for him ("for the meme!"), but didn't necessarily want to follow him. Regardless, the fan vote predictor was unable to account for his unique case.

Our models are inherently limited by our data. This is true of both our Instagram data as well as the raw playing statistics (for which we only have 3 seasons worth). There is also a fundamental disparity between our data sets: we have multiple instances of playing statistics, but we do not have Instagram metrics for every season (only contemporary data). This undoubtedly skews various predictions: the importance score for a player is same now as it was for a player's rookie year. Thus, our model may predict a much higher vote ranking in earlier seasons for players who have been in the NBA for 2-3 years. On the other hand, if we had access to Instagram data from the past, we could observe the changing influence of Instagram and its predictive power (presumably, social media influence plays more of a role in determining the All-Star vote than it did a decade ago). Put simply, if we had more data, we would be able to make predictions for more seasons, and likely more accurately.

In particular, we were limited by the tediousness of our data accumulation - if Instagram had a more functional API, this would have been very different. In the future, we could accumulate data each year to get an importance score every year. This would allow for the creation of new features that may have even more predictive power. For instance, change in importance score across years may hold significant prediction power for all-star voting. Data like this would replace our singular static network with a series of multiple networks, each representing a different point in time. This modification would allow us to build in trends, increasing accuracy for cases like Zaza (for whom we were unable to predict very well). Additionally, tracking cumulative comments and likes for given seasons would aid in measuring the change in engagement from a players followers - this helps capture sudden viral spikes in activity, where fans might comment and like on a post, but not necessarily follow the player.

Chapter 4

Conclusion

This project involved the application of several different concepts covered in class, as well as various technologies we learned along the way or were already familiar with. We acquired Instagram data for the construction of a directed graph of NBA player followings, we created an "importance score" for each player by running Google's PageRank algorithm on that directed graph (as well as other network metrics). We obtained player season statistics for each NBA player in the past three years to go along with the number of all-star fan votes they received in each of those seasons, and lastly, we formatted all of the season statistics and Instagram statistics into an ABT. Using a variety of machine learning algorithms, we observed the predictive power of the various data components we had gathered and generated. The performance of both models were improved by the addition of the generated social metrics. The model could be further improved by the addition of additional data - first in terms of additional seasons, and second in terms of time sensitive social media metrics for each season (rather than only current social media statistics).

4.1 Future Work

There are multiple areas where the network of players could be of use in future work. It would be interesting to analyze the relation between NBA salary and importance score - teams may value a players popularity in addition to performance (selling more tickets and jerseys). Additionally, this sort of data may be of use to advertising companies who are trying to select which player may be the best at promoting a particular product. Choosing players with the most influential social network may be important in yielding the best returns for advertisement. Additionally, when selecting multiple players to promote a product, choosing players that are followed by largely different users would give the product exposure to a broader user base - our network based approach would help visualize this outreach ability.

4.2 Reflections

This project was very fulfilling for both of us. Coming from different academic backgrounds, a lot of the learning involved was from each other. The scope of this project grew vastly since our initial conception. We were forced to learn many different tools, programs, and software packages. Networkx, requests, numpy, scikit, seaborn are some examples of Python packages we used. Unix commands were another crucial tool we utilized, as well as visualization tools like Gephi. No aspect of this project was straightforward: even the data aggregation involved complex logic and many hours of experimentation. It was crucial that we maintained clean, well-commented, and reusable code to ensure we could help one another solve problems. In the end, we were very pleased with how promising our final results were. We were able to translate our intuitions about NBA All-Star voting into a network model that would capture the characteristics we desired, and then see - empirically - these intuitions supported by the results of our models.

4.3 Individual Contributions

Throughout the course of the project we evenly distributed the work load between us. We divided tasks as follows:

Ollie

- Web-scraping and processing of Instagram statistics
- Network Generation and analysis

Matt

- Aggregation of data into ABT
- Creation and running of machine learning models

Shared Tasks

- Basketball statistics processing
- Parsing of HTML to generate raw player statistics
- Manual data collection of following CSVs
- Analysis of machine learning models
- Version control, keeping a well-organized repository, updating README.md

References

[1] Aoife D'Arcy John D Kelleher Brian Mac Namee. *Machine Learning for Predictive Analytics*. The MIT Press, 2015.

[2] Jeff Zillgit. *NBA makes major changes to All-Star voting format*. URL: `https://www.usatoday.com/story/sports/nba/2016/12/19/nba-all-star-2017-voting-change/95606848/`. (accessed: 4.5.2018).