NBA Predictions with Combined Player Statistics

December 12, 2019

Jamison Moody & Mingyan Zhao

1 Motivation and Overview of the Data

Predicting basketball games is a popular pastime for enthusiasts of the sport as well as an important job in sports analytics. NBA front offices want to know if a certain player will give them the unique edge they are looking for to make it to the next round in the playoffs. There are many factors to consider when predicting wins: players, teamwork, chemistry, and variations in individual performance all come into play.

1.1 Our Objective

Much work has been done previously to predict wins. There are matchup predictors on popular sports websites like espn.com and fivethirtyeight.com and research papers have been written addressing the subject. For instance [1] highlights how OLS regression can be used to predict wins by incorporating strength of teams and other important factors including home team advantage, whether or not a team played back-to-back games, etc. Another example of game prediction comes from [2] where the authors use neural networks to help in their predictions.

When we approached our model we wanted to do something a little bit different. Instead of focusing solely on team performance/stats in predicting wins, we wanted the ability to remove and replace players on a team and see how that affects team performance. This would greatly help game prediction because often teams go into a new season with a new combination of players or players get injured throughout the season.

More specifically we had the following questions: How can we swap out players to predict team performance? How do we adjust team stats so if a new player is hogging the ball (and minutes) it will diminish the effect of the other players around him? Can we simulate games by drawing from an empirical distribution of player stats and then combining them into team stats?

We believe this model will fit into current research by offering a unique perspective on game prediction. It will offer the ability to look at different combinations of players to find the optimal lineups. The model also gives the ability to see how playing time, shot attempts, and free throw attempts of players affects team performance. This could be especially useful for coaches, who have control of how long to play specific players (playing time) and how much they should encourage players to shoot the ball (shot attempts) and how much to drive the ball (which leads to free throw attempts).

By converting player stats into team stats, we can effectively simulate games by drawing from an empirical distribution of past player performance and then combining our draws into team data. We can then use the team data to predict head to head matchups. This is the overall idea of our model.

1.2 Overview of the data

In order to create a model with these characteristics, we had to figure out how to turn player stats into team stats and what type of data we wanted to look at. In professional basketball, there are two types of stats: traditional and advanced. Traditional stats can be easily measured during the game like shots attempted and three-point shooting percentage. Advanced stats can be thought of as hand-engineered features made from traditional stats. For example effective field goal percentage, an advanced stat, takes into account that three-point field goals are worth more than two-point field goals and is normalized by field goals attempted. Advanced stats are useful because they are normalized and usually percentages. When predicting team wins, we decided to use advanced stats because of these favorable qualities. However, we need traditional and advanced player stats to create the advanced team stats.

Thus, our dataset has two parts, both from basketball-reference.com: a popular, valid source for team and player basketball stats.

1. `Merged Player and Team Stats (Advanced and Traditional):` We use this table to draw from an empirical distribution. Once specific stats are drawn, we can convert them to team stats. This first part of the dataset covers the past two seasons (including playoffs). Drawing player statistics from too far in the past will not give a good representation of the player's current ability (the player is likely to improve or decline over long periods of time).

DATE	NAME	TEAM NAME	OPP TEAM NAME	TEAM SCORE	OPP TEAM SCORE	W/L TEAM	SP	FG	FGA	...	ORB% OPP TEAM	DRB% OPP TEAM	TRB% OPP TEAM	AST% OPP TEAM	STL% OPP TEAM	BLK% OPP TEAM	TOV% OPP TEAM	USG% OPP TEAM	ORtg OPP TEAM	DRtg OPP TEAM
2017-10-17	Jaylen Brown	BOS	CLE	99	102	0	2376	11	23	...	19.6	82.0	52.1	50.0	3.0	7.1	15.3	100.0	102.7	99.7
2017-10-17	Kyrie Irving	BOS	CLE	99	102	0	2361	8	17	...	19.6	82.0	52.1	50.0	3.0	7.1	15.3	100.0	102.7	99.7
2017-10-17	Jayson Tatum	BOS	CLE	99	102	0	2192	5	12	...	19.6	82.0	52.1	50.0	3.0	7.1	15.3	100.0	102.7	99.7
2017-10-17	Al Horford	BOS	CLE	99	102	0	1927	2	7	...	19.6	82.0	52.1	50.0	3.0	7.1	15.3	100.0	102.7	99.7
2017-10-17	Gordon Hayward	BOS	CLE	99	102	0	315	1	2	...	19.6	82.0	52.1	50.0	3.0	7.1	15.3	100.0	102.7	99.7

2. `Merged Team Stats and Opposing Team Stats (Advanced):` This table goes back seven seasons. Since this data is is not labeled by team or player, we can use as much of it as we want. This is the data we use to train our logistic regression model.

TEAM NAME	OPP TEAM NAME	DATE	W/L TEAM	SP TEAM	FG TEAM	FGA TEAM	FG% TEAM	3P TEAM	3PA TEAM	...	ORB% OPP TEAM	DRB% OPP TEAM	TRB% OPP TEAM	AST% OPP TEAM	STL% OPP TEAM	BLK% OPP TEAM	TOV% OPP TEAM	USG% OPP TEAM	ORtg OPP TEAM
ORL	IND	2013-10-29	0	2880.0	36	93	0.387	9	19	...	27.8	72.3	53.0	50.0	4.2	24.3	19.0	100.0	102.1
CHI	MIA	2013-10-29	0	2880.0	35	83	0.422	7	26	...	14.3	76.1	49.4	70.3	10.3	12.3	17.5	100.0	110.2
LAC	LAL	2013-10-29	0	2880.0	41	83	0.494	8	21	...	37.5	77.3	56.5	54.8	8.0	9.7	15.3	100.0	115.6
IND	NOP	2013-10-30	1	2880.0	30	70	0.429	10	23	...	32.6	82.4	53.8	50.0	7.7	10.6	13.7	100.0	99.4
LAL	GSW	2013-10-30	0	2880.0	35	89	0.393	8	18	...	18.9	82.0	55.2	73.9	7.8	12.7	13.5	100.0	121.9

2 Data Collection and Cleaning

2.1 Data Scraping

We first got the links of all the websites that have player and games stats. Then we run through all the websites to get the player stats and team stats. We loaded data into a pandas dataframe. We then combined each player's stats with their team and opposing team's data and saved the resulting file. We also saved team data as an another file.

Here is some simplified code:

```
# Get all the urls of all the games in last five years
month0 = ['october','november','december','january', 'february','march','april','may','june']
month1 = ['december','january', 'february','march','april','may','june']
years = [str(i) for i in range(2010,2018)][::-1]

# Set the urls
urls = ['https://www.basketball-reference.com/leagues/NBA_2018_games-'+k+'.html' for k in montl
urls = []

# Loop through all the years
for j in years:
    # 2012 had less games, so it is an exception
    if j == '2012':
        month = month1
    else:
        month = month0
    for k in month:
        urls.append('https://www.basketball-reference.com/leagues/NBA_'+j+'_games-'+k+'.html')

# Save URLs for all games in each month
websites = []
browser = webdriver.Chrome()
for url in urls:
    browser.get(url)
    soup = BeautifulSoup(browser.page_source, "html.parser")
    query = soup.find_all(string='Box Score')
    tags  = ['https://www.basketball-reference.com'+ a.parent.attrs['href'] for a in query]
    websites += tags
```

Here is the code for scraping team data from the urls.

```
def game_stats(date):

    soup1 = BeautifulSoup(browser.page_source, "html.parser")
    # Team info
    infos = soup1.find(class_='suppress_all sortable stats_table now_sortable').tbody.find_all

    # Get the different tables
```

```
    tables = soup1.find_all(class_='sortable stats_table now_sortable')

    # Get the team names and scores
    team_name1 = infos[1].find_all('td')[0].text
    score1 = infos[1].find_all('td')[-1].text
    team_name2 = infos[2].find_all('td')[0].text
    score2 = infos[2].find_all('td')[-1].text

    # Check and set the win condition and convert to binary output
    if int(score1) > int(score2):
        win1 = 1
        win2 = 0
    else:
        win1 = 0
        win2 = 1

    # Save team stats
    team_basic = []
    for k in tables[0].tfoot.find_all('td')[:-1]:
        team_basic.append(k.text)
    team_adv = []
    for k in tables[1].tfoot.find_all('td')[1:]:
        team_adv.append(k.text)
    op_basic = []
    for k in tables[2].tfoot.find_all('td')[:-1]:
        op_basic.append(k.text)
    op_adv = []
    for k in tables[3].tfoot.find_all('td')[1:]:
        op_adv.append(k.text)
    t = np.array([team_name1,team_name2,date,win1] + team_basic+ team_adv)
    o = np.array([team_name2,team_name1,date,win2] + op_basic+ op_adv)
    return t, o
```

We scraped the player data in a similar way.

2.2 Data Cleaning

Most of data has the right format when we saved them in pandas dataframe. We made sure that all the quantitative variables were floats. We changed the dates in the tables to pandas datetime index and minutes played to seconds played. We also changed most variable names so they were consistent and we could merge player and team data.

```
# Fix player stat data
player_stats["date"] = pd.to_datetime(player_stats["date"], format="%Y%m%d")

# Convert from minutes played to seconds played (will be useful later)
player_stats['MP'] = [int(a)*60 + int(b) for a, b in player_stats['MP'].str.split(':')]
```

```
# Create consistent column names
player_stats = player_stats.rename({"win": "W/L TEAM", "date":"DATE", "name":"NAME", "team":"TE

# Put the dates in a correct format
team_stats["date"] = pd.to_datetime(team_stats["date"], format="%Y%m%d")

# Convert minutes played to seconds played
team_stats['MP'] = team_stats["MP"]*60/5

# Change the column names to be consistent
team_stats = team_stats.rename({"date":"DATE", "team":"TEAM NAME", "op_team":"OPP TEAM NAME",

# Merge the player and team stats into one table
all_stats = pd.merge(player_stats, team_stats, on=["TEAM NAME", "OPP TEAM NAME", "DATE", "W/L T
```

3 Model Selection and Feature Engineering

3.1 Model Selection

We determined that logistic regression was useful our application because we were trying to predict the chance of a team winning (which is fit on binary output). We knew we wanted to use advanced, normalized stats as input so we looked at the Akaike Information Criterion (AIC) of models with different feature combinations.

After looking at different combinations of features, we had the following results:

	Features	AIC
Best 10 Features	$TS\%_{team}, eFG\%_{team}, ORB\%_{team}, DRB\%_{team}, BLK\%_{team}, TOV\%_{team}, TS\%_{opp}, AST\%_{opp}, BLK\%_{opp}, TOV\%_{opp}$	1261.39
Best 9 Features	$TS\%_{team}, eFG\%_{team}, ORB\%_{team}, DRB\%_{team}, BLK\%_{team}, TOV\%_{team}, TS\%_{opp}, BLK\%_{opp}, TOV\%_{opp}$	1262.01
Best 8 Features	$TS\%_{team}, eFG\%_{team}, ORB\%_{team}, DRB\%_{team}, BLK\%_{team}, TOV\%_{team}, TS\%_{opp}, TOV\%_{opp}$	1263.54
Best 7 Features	$TS\%_{team}, eFG\%_{team}, ORB\%_{team}, DRB\%_{team}, TOV\%_{team}, TS\%_{opp}, TOV\%_{opp}$	1265.42
Best 6 Features[1]	$TS\%_{team}, ORB\%_{team}, DRB\%_{team}, TOV\%_{team}, TS\%_{opp}, TOV\%_{opp}$	1268.93
Best 5 Features	$TS\%_{team}, DRB\%_{team}, TOV\%_{team}, TS\%_{opp}, TOV\%_{opp}$	2414.50

The features chosen for our model denoted in by (1). Notice that this combination of features does not have the lowest AIC. However it isn't too far away from the other models and we wanted to have a small number of features. The less features, the less variation when we convert player stats into team stats. We didn't feel that adding the extra features would help our model substantially.

Here are the results of Logisitic Regression (with L1 Regularization) for our chosen features:

```
best_features_aic = ['TS% TEAM', 'TOV% TEAM', 'DRB% TEAM', 'ORB% TEAM', 'TS% OPP TEAM', 'TOV% O
X = team_stats.filter(best_features_aic, axis=1)
X = sm.add_constant(X,1)
y = team_stats["W/L TEAM"]
```

```
# Run logistic regression
results = sm.Logit(y, X.astype(float)).fit_regularized(disp=0)
```

```
                           Logit Regression Results
==============================================================================
Dep. Variable:                W/L TEAM   No. Observations:                 7642
Model:                           Logit   Df Residuals:                     7635
Method:                            MLE   Df Model:                            6
Date:                 Thu, 12 Dec 2019   Pseudo R-squ.:                  0.8790
Time:                         13:42:43   Log-Likelihood:                -627.47
converged:                       True   LL-Null:                       -5185.9
Covariance Type:            nonrobust   LLR p-value:                     0.000
==============================================================================
                     coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
const           -40.2455      1.963    -20.501      0.000     -44.093     -36.398
TS% TEAM        146.8957      5.721     25.678      0.000     135.683     158.108
TOV% TEAM        -1.1725      0.049    -23.715      0.000      -1.269      -1.076
DRB% TEAM        40.6685      1.850     21.988      0.000      37.043      44.294
ORB% TEAM        38.4502      1.777     21.637      0.000      34.967      41.933
TS% OPP TEAM   -146.9732      5.728    -25.661      0.000    -158.199    -135.747
TOV% OPP TEAM     1.1717      0.050     23.519      0.000       1.074       1.269
==============================================================================

Possibly complete quasi-separation: A fraction 0.47 of observations can be
perfectly predicted. This might indicate that there is complete
quasi-separation. In this case some parameters will not be identified.

AIC: 1268.9303726786206
```

3.2 Feature Engineering

We made a few new features to help with converting player stats into team stats. They include:

`Percentage of Field Goals Attempted`: $FGA\%_i = \frac{FGA_i}{FGA_{team}}$

`Percentage of Free Throws Attempted`: $FTA\%_i = \frac{FTA_i}{FTA_{team}}$

`Percentage of Game Played`: $GP\%_i = \frac{SP_i}{SP_{team}}$

`Turnover Shot Ratio`: $TSr_i = \frac{TOV_i}{FGA_i}$

When we convert player stats to team stats, we made it so $FGA\%_i$, $FTA\%_i$, and $GP\%_i$ can be manually adjusted.

We calculated the team stats from the player stats in the following way. Here an i subscript refers to player i, a *team* subscript refers to the team as whole, *opp* refers to the opposing team, and N is the number of players that played during the given game. Here FGA refers to field goals attempted (the number of shots during game play), FTA refers to free throws attempted, SP refers to seconds played during the game, and TOV refers to the number of turnovers committed during the game.

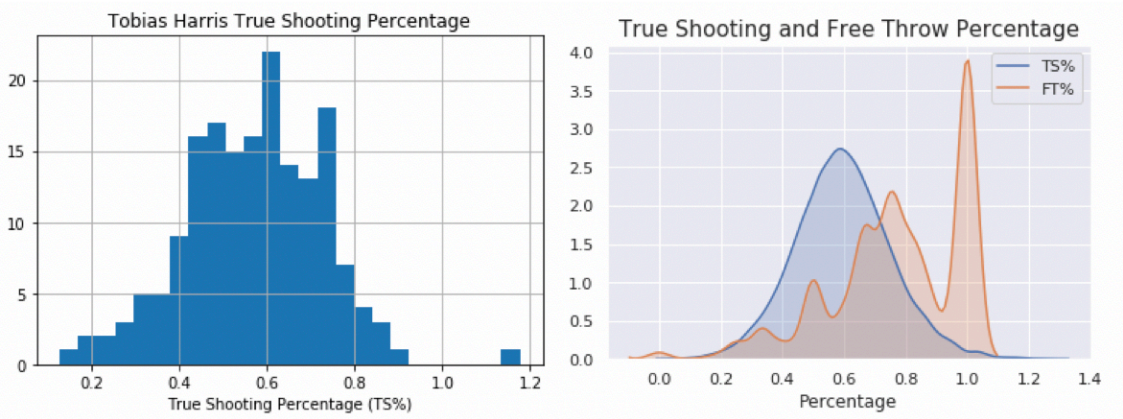
Variable Name	Abbreviation	How to Calculate from Team Statistics	How to Calculate from Player Statistics	Definition
Team True Shooting Percentage	$TS\%_{team}$	$\frac{eFG\%_{team}+\frac{1}{2}\cdot FT\%_{team}\cdot FTAr_{team}}{1+0.44\cdot FTAr_{team}}$		The measure of shooting efficiency that takes into account field goals, 3-point field goals, and free throws.
Team Effective Field Goal Percentage	$eFG\%_{team}$	$\frac{FG_{team}+\frac{1}{2}\cdot 3P_{team}}{FGA_{team}}$	$\sum_{i=1}^{N} FGA\%_i \cdot eFG\%_i$	Adjusted field goal percentage that accounts for three-pointers.
Team Free Throw Percentage	$FT\%_{team}$	$\frac{FT_{team}}{FTA_{team}}$	$\sum_{i=1}^{N} FTA\%_i \cdot FT\%_i$	The ratio of free throws made to attempted free throws.
Team Free Throw Rate	FTr_{team}	$\frac{FTA_{team}}{FGA_{team}}$	$\sum_{i=1}^{N} FGA\%_i \cdot FTr_i$	The ratio of attempted free throws to attempted field goals.
Team Defensive Rebonds Percentage	$DRB\%_{team}$	$\frac{DRB_{team}}{DRB_{team}+ORB_{opp}}$	$\sum_{i=1}^{N} GP\%_i \cdot DRB\%_i$	The estimate of the percentage of available defensive rebounds a player grabbed while he was on the floor.
Team Offensive Rebonds Percentage	$ORB\%_{team}$	$\frac{ORB_{team}}{ORB_{team}+DRB_{opp}}$	$\sum_{i=1}^{N} GP\%_i \cdot ORB\%_i$	The estimate of the percentage of available offensive rebounds a player grabbed while he was on the floor.
Team Turnover Percentage	$TOV\%_{team}$	$\frac{TSr_{team}}{1+0.44\cdot FTAr_{team}+TSr_{team}}$		The estimate of turnovers per 100 plays.
Team Turover Shot Ratio	TSr_{team}	$\frac{TOV_{team}}{FGA_{team}}$	$\sum_{i=1}^{N} TSr_i \cdot FGA\%_i$	The ratio of turnovers to attempted field goals.

4 Data Visualization and Analysis

4.1 Are certain stats correlated?

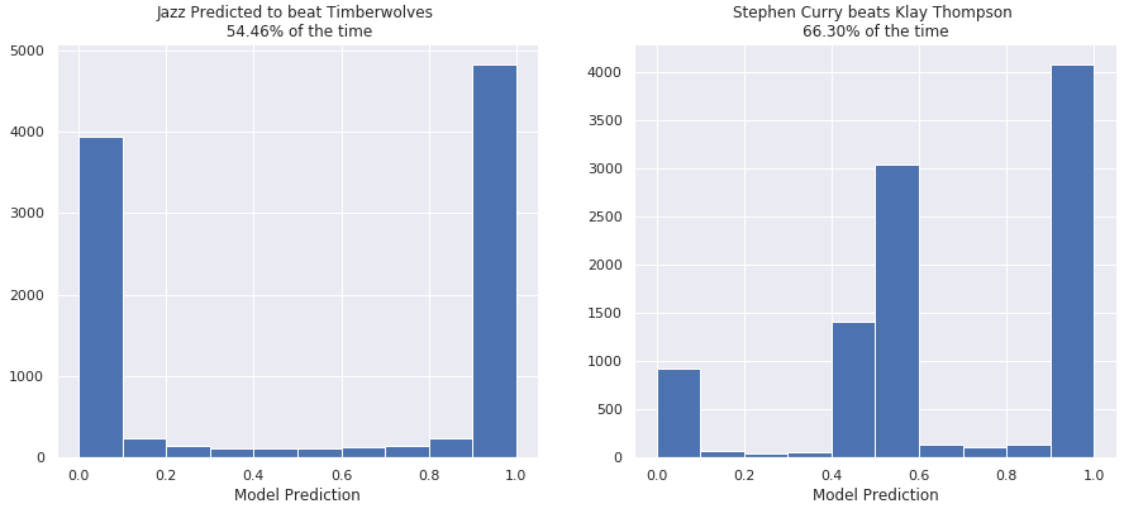
Here we graph player True Shooting Percentage ($TS\%$) against Defensive Rating for the Opposing Team ($DRtg_{opp}$). Both graphs show that TS% has a faint positively correlation with defensive rating. The first graph includes outliers and in the second graph we remove outliers (players that didn't attempt many free throws/field goals or missed all their shots).

4.2 What do the stat distributions look like?



In order to normalize player stats we need to be able to draw from an empirical distribution from their play for the last two seaons. The graph on the left the True Shooting Percentage Distribution for Tobias Harris of the Philadelphia 76ers. The graph on the right shows kernal density estimations (KDEs) for true shooting and free throw percentage for all players in the league.

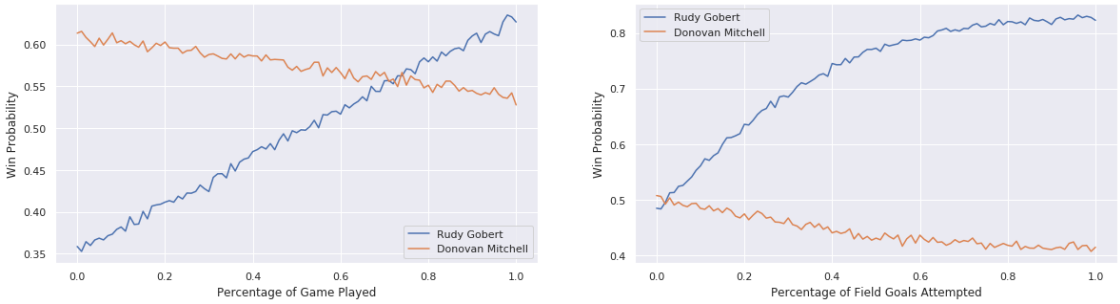
4.3 Who wins in head to head matchups?



Once we pick players and sample their statistics, we can predict head to head matchups. On the right, the prediction was done by making 10,000 draws from each player's stats and combining them into team stats. We then ran the team stats through our model. We did not have 2 of the players' data (they were both rookies), so filled in one of their stats with a player with similar points per game.

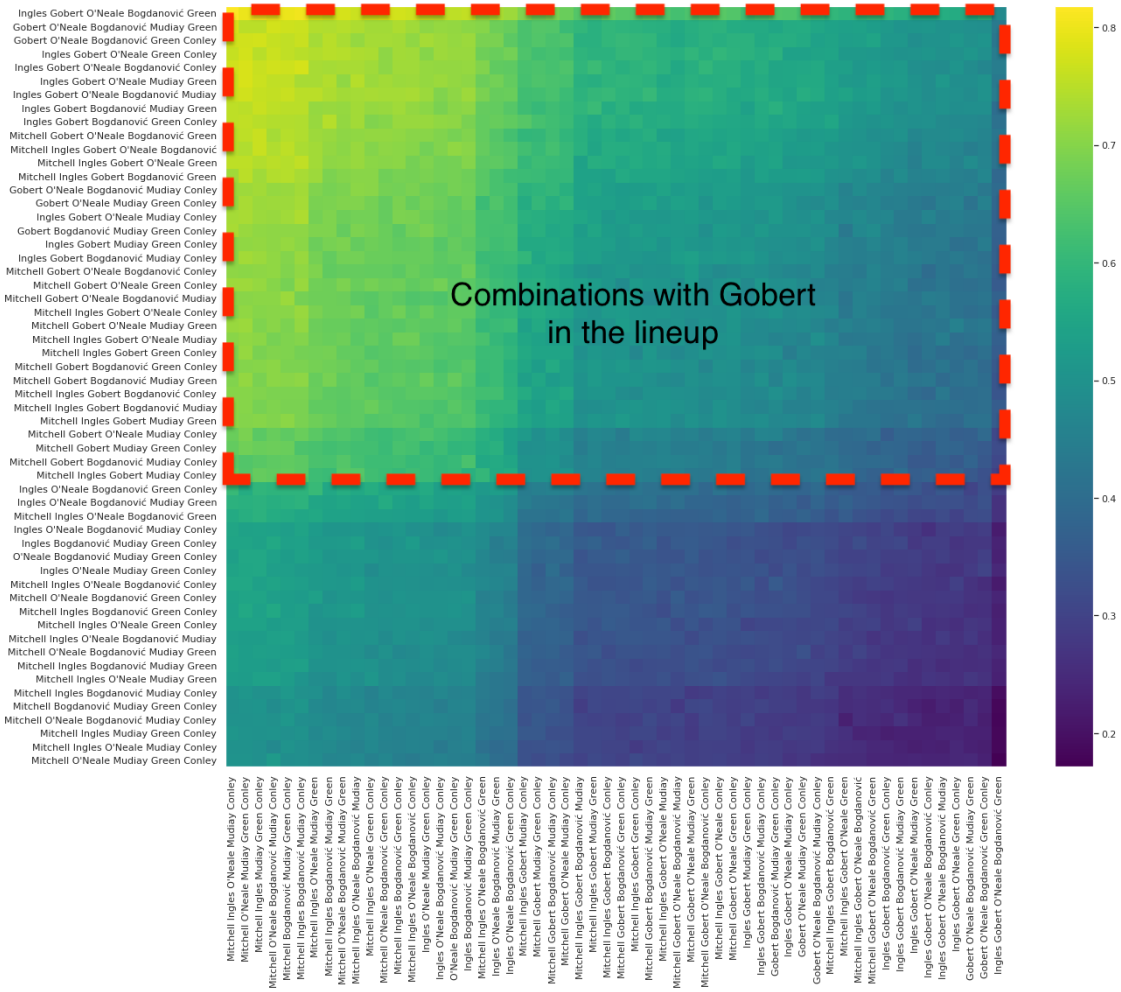
As shown on the left, we found out that we can also predict head to head matchups between players. It might be a stretch to compare players because the model was fit on team data, but the results are reasonable.

4.4 How much should we use certain players?



We ran simulations of the Utah Jazz against the Minnesota Timberwolves (the next team they were facing) to find the optimal playing time and field goals attempts for Donovan Mitchell and Rudy Gobert. On the left we see that as Donovan Mitchell plays more, the win probability of the Jazz goes down slightly. However the more Rudy Gobert plays, the better the team does. On the right, we see that we want Rudy Gobert to shoot the ball more and Donovan to shoot the ball less to help the team. This is important information for their coach.

4.5 What are the best lineups for the Utah Jazz?



We picked the 8 best players on the Jazz and looked at all the possible five-man lineups. Our best lineup hasn't played together on the Jazz before, but it seems like a reasonable lineup. Notice that all the best lineups include Rudy Gobert, which is consistent with our earlier results.

5 Future Directions

We understand that there are many factors we neglected to include in our model. While we did incorperate home-team advantage into our model, it would wise to look at the affect of fatigue, defensive rating, injuries, motivation of players and hot shooting streaks. We look forward to evaluating and refining our model next semester.

6 References

[1] Manner, Hans. "Modeling and forecasting the outcomes of NBA basketball games." Journal of Quantitative Analysis in Sports 12, no. 1 (2016): 31-41.

[2] Loeffelholz, Bernard, Earl Bednar, and Kenneth W. Bauer. "Predicting NBA games using neural networks." Journal of Quantitative Analysis in Sports 5, no. 1 (2009).