15.095 ML Under a Modern Optimization Lens **Fall 2019**
Final project report

Picking Lineups for Daily Fantasy NBA Competitions

Using the power of optimization

Team members

Anis Ben Said
Asher Wright

Contents

Abstract

Online fantasy sports are played by around 60 million North Americans each year [3], and are a great area for analytics. This project targeted a subset of these: daily fantasy NBA competitions. We aimed to more accurately predict players' performances in a contextual way, and create lineups based on the underlying prediction methods, in a way similar to optimal prescription trees [2]. We aimed to create many lineups with an unexceptional average score, but high variance, such that at least one will perform exceptionally well and offset the losses of the others. Unfortunately, we did not meet a key metric value that we felt was required to start testing in the real world.

1 Description

1.1 Inspiration

This project was inspired by "Picking Winners" by David Hunter et. al at the MIT ORC [4]. The authors created an optimization formulation for daily NHL competitions, in which they used purchased predicted player performances alongside hockey expertise to create highly dissimilar lineups with high-variance. Basketball is a much lower variance sport, which makes it harder to apply these techniques. This is one of the differences between our project and the paper. The second, more significant, difference, is that we aimed to use the underlying prediction model to compute pseudo-variances of the players' performances to create high variance lineups, instead of using domain logic.

1.2 Problem Overview

A quick note on terminology. We call a real NBA game a **Game**, a fantasy competition based on NBA games a **Competition**, an NBA player a **Player**, someone playing fantasy sports a **Competitor**, an NBA team a **Team**, and a submission for a competition (list of players) a **Lineup**.

This project targets daily fantasy sports competitions, which are different from the more popular seasonal competitions. Daily competitions occur on a single day alone. The goal for these competitions is to choose a lineup of NBA players under a set of constraints in order to maximize the total "fantasy score". Each player has a position (center, shooting guard, etc.), and a cost, which is roughly based on their historical performance. Each competitor has to choose a lineup of a given number of players to cover a set of positions without exceeding a given budget. After the NBA games are played, each player has earned a fantasy score, based on their performance in the game, and the lineup's score is the sum of the fantasy scores of the players in the lineup. The formula for computed these points is given in Equation 1, where the variables are the abbreviated statistics (can be found online).

$$S = 3(3PT) + 2(FG) + FT + 1.2(RBD) + 1.5(AST) + 3(BLK) + 3(STL) - TVR \tag{1}$$

It is important to note that two competitors can both choose the same player, and even the same lineup. Another note about the targeted competitions is that they are "top-heavy". This means that a disproportionate amount of the winnings is awarded to the top performing competitors. A representative top-heavy competition has an entry fee of \$5 and top-10 winning prices ranging from \$100,000 for the best competitor to \$1000 for the 10th best competitor. Submitting 200 lineups breaks even if one of the lineups places at 10th or better, and all other lineups score no money. Outside of the top 10, the payout reduces significantly as placement decreases, and lineups that score at the 80th percentile are the last to earn a profit.

Figure 1 shows the way a lineup is selected for these competitions. The top bar shows the NBA games that are happening on that day (and thus that are considered for the competition). The left panel shows a list of all of the players, and how much they cost. The right panel shows players that

have been selected per position, and what positions are needed. It is required that lineups have a set number of shooting guards, shooting forwards, power forwards, and centers (real basketball positions). Above that panel, the remaining budget for players is given, which started at $60,000. Note that there is an equivalent CSV upload process that can be used to mass enter lineups.

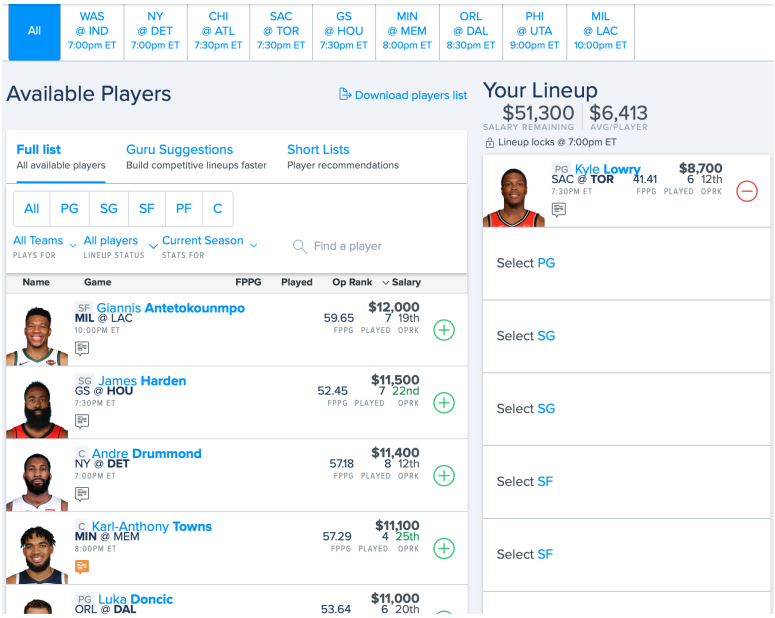


Figure 1: Sample FanDuel competition interface

1.3 Importance

As mentioned earlier, fantasy sports are played by around 60 million North Americans [3]. Succeeding in these competitions is a very hard problem, and may require developing novel analytics methods. These methods can be applicable to other domains, which is one way in which this problem is important. A second way is that portions of our project, if successful, could be used in real-life sports decisions. We propose a way of determining a potentially more accurate prediction of a player's performance in a game, based on the skill levels of the players on his team and the opposing team. If this prediction is accurate, it could be used by coaches to help determine which players to play in a game.

2 Data

Collecting detailed historical data for NBA games is not as straightforward as it may seem. Although there are many online resources for viewing data, none of these have a way to download data in bulk. This difficulty is shown by the success of websites whose sole purposes are to sell bulk data, such as www.nbastuffer.com. We considered this route, as we felt building a scraper would waste too much time. However, we then found an existing web scraper [1] for basketball-reference, a site dedicated to NBA stats [5], which we used instead.

The first data we collected were player seasons (5000 rows), which gave season-level data for each player in the NBA, per year. This included the player's team, position, and average season stats (e.g. foul shots made). The second were player matches (178000 rows), which contained data for every player *per game*. These data include all possible game-level player stats. We collected data from 2010 to present, as we felt that recent data would be better for predicting current stats. We generated a third dataset from the first two - the input to the prediction models we used. This dataset was the `games` table (12700 rows; number of NBA games since 2010). Each row corresponded to a real-life game, and k stats for the n best players from each team.

3 Methodology

To recap, the goal was to generate many highly different and variable lineups, such that at least one was exceptional and thus the system was profitable. A straightforward way to approach this problem was to first predict the performance of each player, then optimize the lineups to submit. In our approach however, we took advantage of the structure of the predictive model alongside its predictions in the optimization part of the problem. This flow can be seen in Figure 2. Moreover, when predicting each player's performance, instead of only using the player's stats, our model tries to capture his interactions with the rest of players on the basketball court.

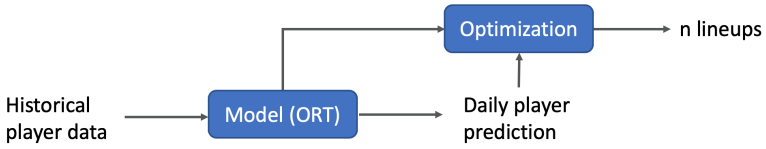


Figure 2: High-level system overview

To obtain highly different lineups, we used the power of optimization to impose that two lineups cannot share more than a certain number of players. To build lineups with high-variance performance, we use trees to predict player performances and we select players that fall into leafs with a high average but also a high variance. In other words, we use impure leafs with high predictions. Our intuition is that, players picked from these leafs will, on average, perform well but can also perform extremely well (or poorly). We hoped this would achieve the desired high variance lineups needed to succeed.

3.1 Prediction

Figure 3 shows the prediction model's inputs and outputs. As mentioned, we aimed to predict each player's performance *in the context of the relevant game*. We thus used the stats of the top n (chosen as 7) players on each team as inputs, when predicting the fantasy points for every player. One way to think about this is that a player with a normal fantasy point average may perform better if his team has players with a high number of assists and a weak all around opposing team.

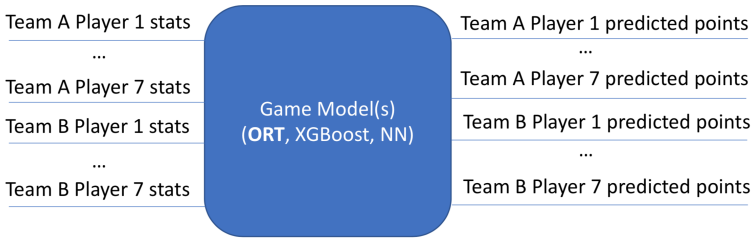


Figure 3: Overview of the predictive "games" model

Ultimately, we used Optimal Regression Trees (ORTs) [2] as the underlying model, for reasons that will become clear. However, as an aside, we were interested in how well these ORTs compared to more common models, and we tested neural networks, random forests, and boosting models to find out. The neural networks were trained with manually selected hyperparameters (hidden layers, nodes per layer, and learning rate). The best gave us an out of sample MAE of 8.35. With random forests, we performed a more proper cross-validation, and achieved an MAE of 8.18. Finally, with a cross-validated boosting model (via XGBoost), we reached an MAE of 8.00.

Keeping those MAEs in mind, we pursued ORTs. Initially, the MAE was around 8.5, which was close enough to the other models that we justified the use of ORTs. However, as we cross-validated

and improved the structure of the incoming data (cutting out unnecessary features), MAE improved to 7.66 - significantly better than all other models! These results are summarized in table 1.

Table 1: Predictive performance of various models

Model	**MAE**
XGBoost	8.00
Neural Network	8.35
Random Forest	8.18
ORT	7.66

The final ORT is given in Figure 4, shown predicting player 1's (p1) performance. In this tree, all of the top splits are on variables related to player 1's performance (indicated by the suffix `_p1`), which is expected! In other words, player 1's historical performance is the most important indicator of future performance. The tree then takes into account other players at the splits at depth 3. Notably, it looks at player 2's and player 8's performance, who are player 1's strongest ally and strongest enemy, respectively, and makes the most sense as choices of the next most important players. As expected, if player 2 is strong, then player 1 will see a boost in their predicted score. If player 8 is strong, then player 1 will be predicted weaker!

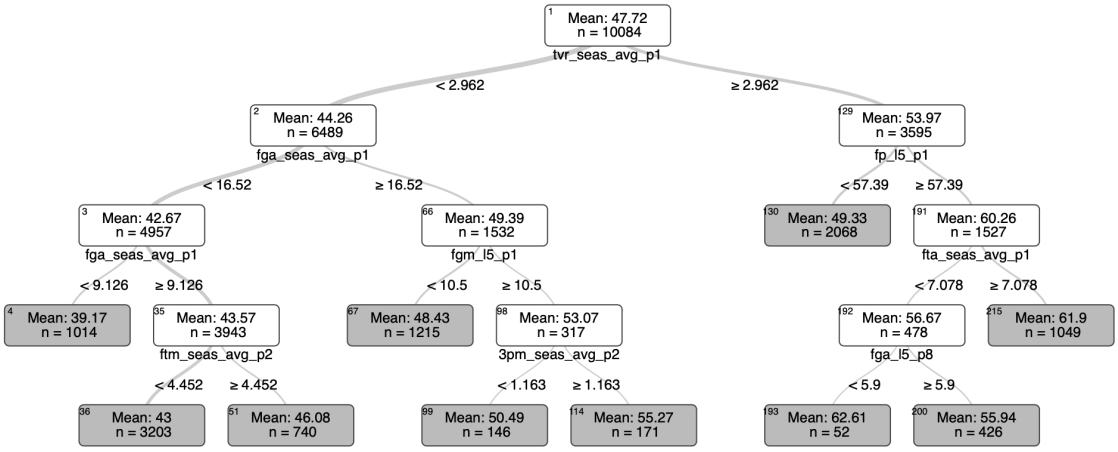


Figure 4: Semi-collapsed ORT for player predictions

3.2 Optimization

Rather than only using the ORT's point-predictions, we added pseudo "variance" to the predictions. To do this, for each new test sample, we examined the training data that was in the same leaf as the new sample. We then looked at the variance of the training data in that same leaf, and added this variance as an input to the optimization model. Figure 5 gives an overview of this process.

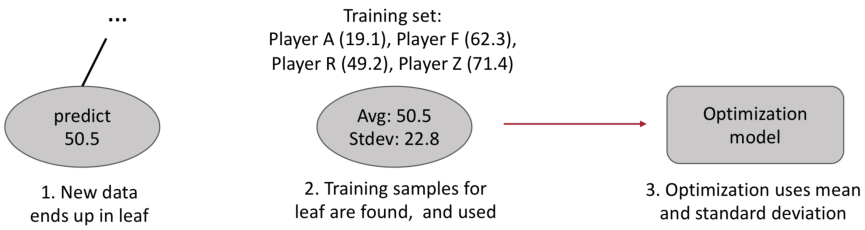


Figure 5: Overview of the pseudo-variance process

Then, to form one lineup we use the following Integer Optimization (IO) formulation :

Sets and indices

$i \in \{1, ..., n\}$: indices of the available indices, n being the number of available players

$j \in P = \{PG, SG, SF, PF, C\}$: set of available positions

Parameters

μ_i : predicted performance of player i

σ_i : standard deviation of the leaf to which player i belongs

ρ : weight of the predicted Fantasy Points and the standard deviation

s_i : salary of player i

$$p_{i,j} := \begin{cases} 1 & \text{if player } i \text{ plays in position } j \\ 0 & \text{otherwise} \end{cases}$$

B : total budget

M_j : number of players required in position j

Decision variables

$$z_i := \begin{cases} 1 & \text{if player } i \text{ is selected in the lineup} \\ 0 & \text{otherwise} \end{cases}$$

Formulation

$$\max_z \ (1-\rho)\sum_{i=1}^{n} \mu_i z_i + \rho \sum_{i=1}^{n} \sigma_i^2 z_i \tag{2}$$

$$\text{s.t.} \ \sum_{i=1}^{n} p_{i,j} z_i = M_j \qquad \forall j \in \{PG, SG, SF, PF, C\} \tag{3}$$

$$\sum_{i=1}^{n} s_i z_i \leq B \tag{4}$$

Explanation of formulation

- The objective function (2) is a weighted average of predicted points and standard deviation.
- Constraint (3) ensures all required positions for a submission are covered by the lineups.
- Constraints (4) ensures that the salaries of the selected players fit in the allowed budget.

To get different lineups, each time we solve the formulation, we store the lineup in matrix x with $x_{i,l}$ equal 1 if player i is selected in lineup l. To get lineup k, $k \in \{2,..,m\}$, we introduce γ, the maximum number of players in common between two lineups, and add the following constraint:

$$\sum_{i=1}^{n} z_i x_{i,l} \leq \gamma \qquad \forall 1 \leq l < k \tag{5}$$

We also created a baseline model that doesn't use the variance in the objective function:

$$\max_z \ \sum_{i=1}^{n} \mu_i z_i \tag{6}$$

And a second model that boosts the Fantasy points prediction by adding the standard deviation:

$$\max_z \ \sum_{i=1}^{n} (\mu_i + \lambda \sigma_i) z_i \tag{7}$$

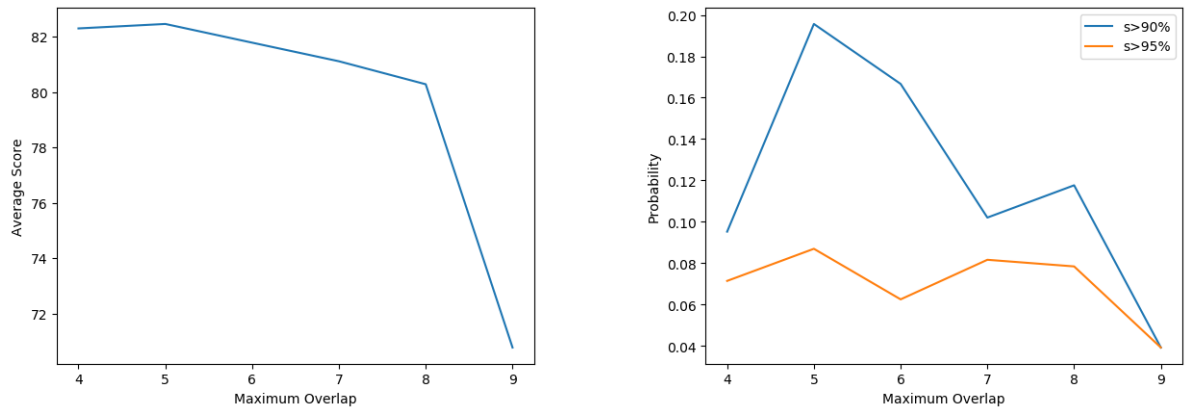
4 Results

Since we couldn't directly compute our system's profitability (no historical competition data), we defined a metric that could be backtested. The metric, called the **score**, is how close our best lineup is to the best possible lineup. I.e. if the best possible total fantasy points is 300, and we have 3 lineups with 200, 250, 275, $score = s = \frac{275}{300} = 0.917 = 91.7\%$.

We generated "fake" historical competitions based on real games and calculated three metrics for each system: the average **score**, the frequency of the score being greater than 90% (s > 90%), and the frequency of the score being greater than 95% (s > 95%). 90% and 95% were chosen because they were found empirically to be the break-even and high-success profitability points.

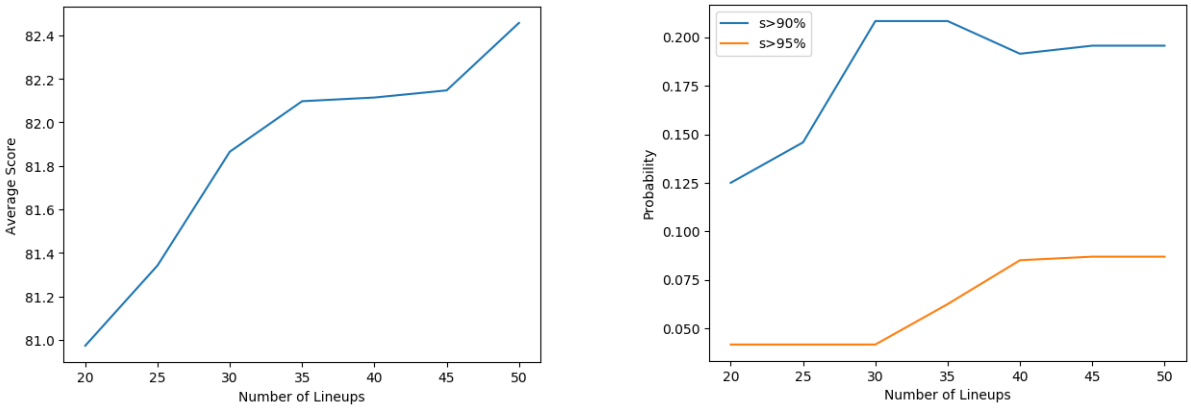
Figure 6 shows these metrics as a function of the max overlap between lineups. Choosing a max overlap lower than 5 is constraining, but allowing for more than 5 common players leads to too similar lineups that do not cover a large enough outcome space. Figure 7 shows the performances as a function of the number of created lineups. We see that, as expected, more lineups means better performance.

Going forward, we chose a maximum overlap of 5 players, and 50 generated lineups. Figure 8 show the evolution of the performance metrics as a function of the weight attributed to the variance in the objective function. Figure 9 does the same, but for a model where we add a percentage of the standard deviation to the objective function (see Equation 7). Concerning ρ, it appears better to either maximize the variance or the score, not both. Concerning λ, we see no obvious trend.



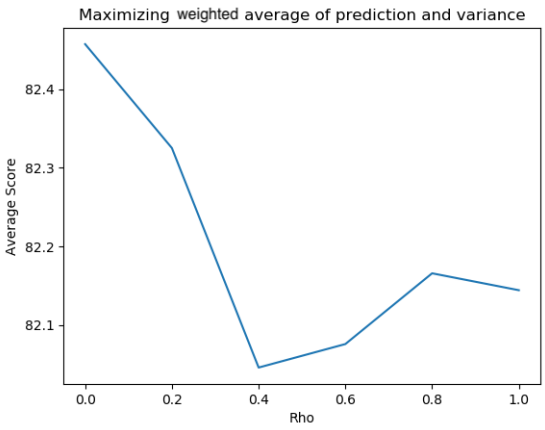
(a) Average performance (b) Probabilities

Figure 6: System performance vs. maximum overlap between lineups

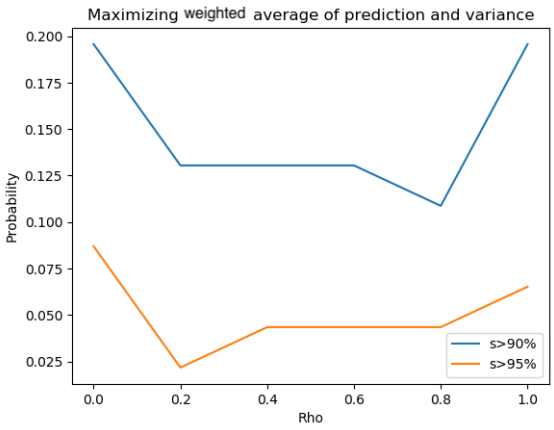


(a) Average performance (b) Probabilities

Figure 7: System performance vs. number of lineups

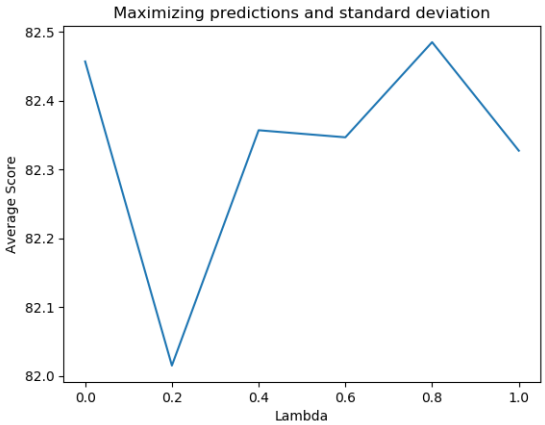


(a) Average performance

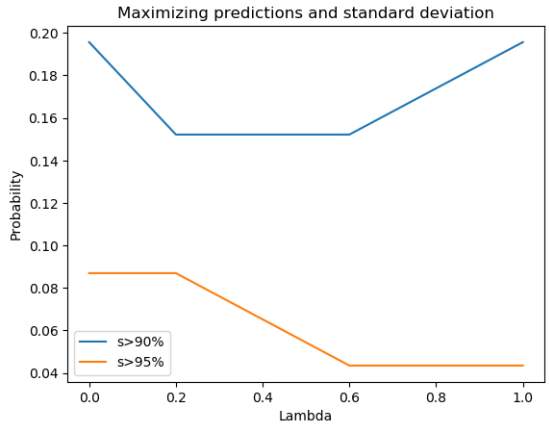


(b) Probabilities

Figure 8: System performance vs ρ (weighted average parameter)



(a) Average performance



(b) Probabilities

Figure 9: System performance vs. λ (percentage of standard deviation to add)

The results of each model are displayed in Table 2. We see that on average, we are unable to reach the 90% profitability mark in any of our three designs. However, some of our lineups hit 95% with a non negligible probability. It is hard to know if any system would be profitable in real competitions, as it would depend on, in the winning lineups (s > 95 %, e.g.), how much money would be won, and how much it would cost to enter all the competitions when we lost. However, by spot-checking competitions manually, it seems that even with a probability of 9% of creating ultra-high-scoring lineups, the overall system would likely not be profitable, or at least some luck would be required.

System	**Avg performance**	**Prob of >90%**	**Prob of >95%**
Point predictions (baseline)	83 %	21 %	8 %
Maximize score and variance	82 %	20 %	8 %
Maximize adjusted score	83 %	22 %	9 %

Table 2: Performance of each of the three tested models

5 Conclusion

Using techniques similar to those in prescriptive analysis, where the underlying prediction model is used in the optimization formulation, may be an effective way to gain an edge in fantasy sports, but the model and assumptions need to be accurate. Our prediction model was not accurate enough to yield a consistently profitable system, and the heuristic for variance was not as helpful as we hoped. The MAE on the predicted player scores of 7.66 was too high, and led to an average score of around 83% of the best possible score, whereas a score of 90% is needed to be profitable. We experimented with different uses of the underlying model, including as a pseudo variance, and saw that our models reached a score of 90% around 20% of competitions. They also reached a "highly-profitable" score of 95% around 10% of competitions, but this was not enough to instill confidence that we should start participating in real competitions. Although it is possible that the system would be profitable in extremely top-heavy competitions, we felt, through manual checks, that this is unlikely. All of that being said, using optimization allowed us to easily model and experiment with different constraints and objective functions. Additionally, many improvements can be made, such as adding professional predictions as input to the model, and adding more data than the just the past decade, or at least looking into the benefit of doing so. Finally, there are other ways to approach these competitions that have been shown to work in certain sports [4], and a strategy could be to instead try to adapt these for the NBA. Finally, we'd like to thank you (Ryan Cory-Wright, Michael Li, and Dimitris Bertsimas) for instilling in us the optimism to try out an idea like this.

6 Teammate Contributions

Asher and Anis worked together on almost all of the project, but both teammates had tasks that they took the lead on. For Asher, that included the data collection, cleaning, and creation of datasets. As well, it included building the non ORT models. For Anis, it included training the ORT models, and building the optimization code. Both Asher and Anis worked on testing; creating fake test competitions, defining a performance metric, and building code to bulk test the fake competitions.

References

[1] Jae Bradley. NBA Stats API via Basketball Reference. `https://github.com/jaebradley/basketball_reference_web_scraper`, 2019. [Online; accessed 2-Dec-2019].

[2] Bertsimas & Dunn. *Machine Learning Under a Modern Optimization Lens*. Dynamic Ideas LLC, 2019.

[3] Eric Allen Hall. Perspective | The Dark Side of Fantasy Football. `www.washingtonpost.com/news/made-by-history/wp/2017/09/10/the-dark-side-of-fantasy-football/`, 2019.

[4] David Scott Hunter, Juan Pablo Vielma, and Tauhid Zaman. Picking winners in daily fantasy sports using integer programming, 2016.

[5] Sports Reference. Basketball Reference. `https://www.basketball-reference.com/`, 2019. [Online; accessed 2-Dec-2019].

7 Appendix A: Code

Please see the GitHub repository at https://github.com/AsherWright/ml-fantasy-basketball for the full code. Note that it isn't super clean. If you have questions about it or want a rundown, please contact Asher or Anis.