ISE 3230 Group 14

Project Report - Austin Trout, Ryan Marshall, Nate Rezell

**Introduction**

Our project goal is to utilize our skills in multiple integer linear programming to create a team of players currently in the NBA that will have the best chance at winning a championship title. We utilized the [RAPTOR](https://fivethirtyeight.com/features/how-our-raptor-metric-works/) score provided by [fivethirtyeight](https://github.com/fivethirtyeight/data/tree/master/nba-raptor) to perform our analysis. The RAPTOR dataset fivethirtyeight provided was cleaned to include only players in the 2020-2021 season that played for at least 500 minutes. The RAPTOR score of a player has three values; offensive, defensive, and total. The acronym stands for Robust Algorithm (using) Player Tracking (and) On/Off Ratings, where its value calculates the player’s impact on the court, using points per 100 possessions compared to the league average. The most important factor of RAPTOR is the on/off factor, where a given player’s team is evaluated while the player is on the court and off the court.Our analysis was done based on the total RAPTOR score of the players to help put together a more well-rounded team (offensive + defensive = total score).

The most impactful aspect to our project comes in our variables and objective function. We chose to utilize a weighted objective function, where each player’s RAPTOR score will be associated with a certain amount of projected playtime (36 min for starters/ 8.5 for bench). This gives priority to the quality of the starters, while still providing a competitive environment for the bench players to earn a spot. This weighted MILP involved maximizing the team’s RAPTOR score by multiplying the playtime of player n by the sum of their RAPTOR score with a decision variable , = {1, 0} based on if they are selected for the starter squad , or bench (

Finally, understanding and applying real-world constraints is paramount to the structure of our model, as the optimal solution to this project without a proper set of limits would yield a significantly different (and probably unrealistic) result. We chose to set our roster size to 12 instead of the usual 15 to allocate more resources into our players with a salary cap of $109,000,000. The cap was chosen because it was the official salary cap from the 2020-2021 season. Using our objective function, decision variables, and constraints, we constructed an MILP to input to CVXPY’s GUROBI solver to obtain our optimal NBA roster.

**Motivation**

With professional sports teams beginning to place more emphasis on analytics when making sports-related decisions, we wondered how we could apply linear optimization to this idea. Our appreciation for basketball made us decide to create an optimal NBA roster given some modern analytics metrics under real-world constraints.

**Decision Variables**

**This is the rendered form of the equation. You can not edit this directly. Right click will give you the option to save the image, and in most browsers you can drag the image onto your desktop or another program.**

**This is the rendered form of the equation. You can not edit this directly. Right click will give you the option to save the image, and in most browsers you can drag the image onto your desktop or another program.**

**This is the rendered form of the equation. You can not edit this directly. Right click will give you the option to save the image, and in most browsers you can drag the image onto your desktop or another program.**

**This is the rendered form of the equation. You can not edit this directly. Right click will give you the option to save the image, and in most browsers you can drag the image onto your desktop or another program.**

**This is the rendered form of the equation. You can not edit this directly. Right click will give you the option to save the image, and in most browsers you can drag the image onto your desktop or another program.**

**This is the rendered form of the equation. You can not edit this directly. Right click will give you the option to save the image, and in most browsers you can drag the image onto your desktop or another program.**

s = cp.Variable(len(data.player\_name), boolean=True)

**This is the rendered form of the equation. You can not edit this directly. Right click will give you the option to save the image, and in most browsers you can drag the image onto your desktop or another program.**

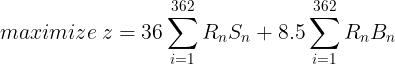
b = cp.Variable(len(data.player\_name), boolean=True)

**This is the rendered form of the equation. You can not edit this directly. Right click will give you the option to save the image, and in most browsers you can drag the image onto your desktop or another program.**

**This is the rendered form of the equation. You can not edit this directly. Right click will give you the option to save the image, and in most browsers you can drag the image onto your desktop or another program.**

**This is the rendered form of the equation. You can not edit this directly. Right click will give you the option to save the image, and in most browsers you can drag the image onto your desktop or another program.**

**Objective Function and Formulation (Code Included)**

****

| # total impact as obj\_func  # assuming 36 minutes for starters and 12 for bench players  obj\_func = 36\*(s.T @ data.raptor\_total) + 12\*(b.T @ data.raptor\_total) |
| --- |

**Constraints**

*KEY:*

*Written Constraint*

*Python Code*

**This is the rendered form of the equation. You can not edit this directly. Right click will give you the option to save the image, and in most browsers you can drag the image onto your desktop or another program.**

| # select 12 players – 5 starters  constraints.append(sum(s) == 5) |
| --- |

**This is the rendered form of the equation. You can not edit this directly. Right click will give you the option to save the image, and in most browsers you can drag the image onto your desktop or another program.**

| # select 12 players – 7 bench  constraints.append(sum(b) == 7) |
| --- |

**This is the rendered form of the equation. You can not edit this directly. Right click will give you the option to save the image, and in most browsers you can drag the image onto your desktop or another program.**

| # total team salary constraint $109 mil  constraints.append((s.T + b.T) @ data.salary <= 109000000) |
| --- |

**This is the rendered form of the equation. You can not edit this directly. Right click will give you the option to save the image, and in most browsers you can drag the image onto your desktop or another program.**

| # combined bench RAPTOR must be nonnegative for offense  constraints.append(b.T @ data.raptor\_offense >= 0) |
| --- |

**This is the rendered form of the equation. You can not edit this directly. Right click will give you the option to save the image, and in most browsers you can drag the image onto your desktop or another program.**

| # combined bench RAPTOR must be nonnegative for defense  constraints.append(b.T @ data.raptor\_defense >= 0) |
| --- |

**This is the rendered form of the equation. You can not edit this directly. Right click will give you the option to save the image, and in most browsers you can drag the image onto your desktop or another program.**

| # starters to include 2 guards  constraints.append(s.T @ data.guard == 2) |
| --- |

**This is the rendered form of the equation. You can not edit this directly. Right click will give you the option to save the image, and in most browsers you can drag the image onto your desktop or another program.**

| # starters to include 2 forwards  constraints.append(s.T @ data.forward == 2) |
| --- |

**This is the rendered form of the equation. You can not edit this directly. Right click will give you the option to save the image, and in most browsers you can drag the image onto your desktop or another program.**

| # starters to include 1 center  constraints.append(s.T @ data.center == 1) |
| --- |

**This is the rendered form of the equation. You can not edit this directly. Right click will give you the option to save the image, and in most browsers you can drag the image onto your desktop or another program.**

| # bench to include at least 2 guards  constraints.append(b.T @ data.guard >= 2) |
| --- |

**This is the rendered form of the equation. You can not edit this directly. Right click will give you the option to save the image, and in most browsers you can drag the image onto your desktop or another program.**

| # bench to include at least 2 forwards  constraints.append(b.T @ data.forward >= 2) |
| --- |

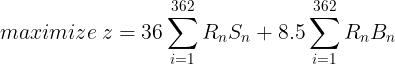
**This is the rendered form of the equation. You can not edit this directly. Right click will give you the option to save the image, and in most browsers you can drag the image onto your desktop or another program.**

| # bench to include exactly 2 centers  constraints.append(b.T @ data.center == 2) |
| --- |

**This is the rendered form of the equation. You can not edit this directly. Right click will give you the option to save the image, and in most browsers you can drag the image onto your desktop or another program.**

| # ensure player can't be a starter and bench player  for i in range(len(data.player\_name)):  constraints.append(s[i] + b[i] <= 1) |
| --- |

**Formulation**

****

**This is the rendered form of the equation. You can not edit this directly. Right click will give you the option to save the image, and in most browsers you can drag the image onto your desktop or another program.**

**This is the rendered form of the equation. You can not edit this directly. Right click will give you the option to save the image, and in most browsers you can drag the image onto your desktop or another program.**

**This is the rendered form of the equation. You can not edit this directly. Right click will give you the option to save the image, and in most browsers you can drag the image onto your desktop or another program.**

**This is the rendered form of the equation. You can not edit this directly. Right click will give you the option to save the image, and in most browsers you can drag the image onto your desktop or another program.**

**This is the rendered form of the equation. You can not edit this directly. Right click will give you the option to save the image, and in most browsers you can drag the image onto your desktop or another program.**

**This is the rendered form of the equation. You can not edit this directly. Right click will give you the option to save the image, and in most browsers you can drag the image onto your desktop or another program.**

**This is the rendered form of the equation. You can not edit this directly. Right click will give you the option to save the image, and in most browsers you can drag the image onto your desktop or another program.**

**This is the rendered form of the equation. You can not edit this directly. Right click will give you the option to save the image, and in most browsers you can drag the image onto your desktop or another program.**

**This is the rendered form of the equation. You can not edit this directly. Right click will give you the option to save the image, and in most browsers you can drag the image onto your desktop or another program.**

**This is the rendered form of the equation. You can not edit this directly. Right click will give you the option to save the image, and in most browsers you can drag the image onto your desktop or another program.**

**This is the rendered form of the equation. You can not edit this directly. Right click will give you the option to save the image, and in most browsers you can drag the image onto your desktop or another program.**

**This is the rendered form of the equation. You can not edit this directly. Right click will give you the option to save the image, and in most browsers you can drag the image onto your desktop or another program.**

**This is the rendered form of the equation. You can not edit this directly. Right click will give you the option to save the image, and in most browsers you can drag the image onto your desktop or another program.**

This is the rendered form of the equation. You can not edit this directly. Right click will give you the option to save the image, and in most browsers you can drag the image onto your desktop or another program.

This is the rendered form of the equation. You can not edit this directly. Right click will give you the option to save the image, and in most browsers you can drag the image onto your desktop or another program.

This is the rendered form of the equation. You can not edit this directly. Right click will give you the option to save the image, and in most browsers you can drag the image onto your desktop or another program.

*Code:*

| Data based on 2020-2021 NBA season Includes player impact (offensive/defensive/total) as estimated by 538's RAPTOR along with position and salary for 20-21 season. Includes only players with at least 500 minutes played in 2020-21. ''' data = pd.read\_csv("raptor.csv")  # convert salaries to type int for i in range(len(data.salary)):  data.loc[i, 'salary'] = int(float(data.salary[i].strip('$').replace(',', '')))  # 362 boolean decision vars s = cp.Variable(len(data.player\_name), boolean=True) b = cp.Variable(len(data.player\_name), boolean=True)  # total impact as obj\_func  # assuming 36 minutes for starters and 12 for bench players obj\_func = 36\*(s.T @ data.raptor\_total) + 12\*(b.T @ data.raptor\_total)  constraints = [] # select 12 players - 5 starters + 7 bench constraints.append(sum(s) == 5) constraints.append(sum(b) == 7)  # total team salary constraint $109 mil # based on  constraints.append((s.T + b.T) @ data.salary <= 109000000)  # combined bench RAPTOR must be nonnegative for offense/defense constraints.append(b.T @ data.raptor\_defense >= 0) constraints.append(b.T @ data.raptor\_offense >= 0)  # starters to include 2 guards, 2 forwards, 1 center  constraints.append(s.T @ data.guard == 2) constraints.append(s.T @ data.forward == 2) constraints.append(s.T @ data.center == 1)  # bench to include at least 2 guards, at least 2 forwards, and exactly 2 centers constraints.append(b.T @ data.guard >= 2) constraints.append(b.T @ data.forward >= 2) constraints.append(b.T @ data.center == 2)  # ensure player can't be a starter and bench player  for i in range(len(data.player\_name)):  constraints.append(s[i] + b[i] <= 1)  # build optimal roster problem = cp.Problem(cp.Maximize(obj\_func), constraints) problem.solve(solver=cp.GUROBI,verbose = False)  #construct table to display optimal team t = PrettyTable(['Player', 'Position', 'Offensive Raptor',   'Defensive Raptor', 'Total Raptor', 'Role']) for i in range(len(data.player\_name)):  if s.value[i] == 1:   t.add\_row([data.player\_name[i], data.position[i], data.raptor\_offense[i],   data.raptor\_defense[i], data.raptor\_total[i], "Starter"])  t.add\_row(['','','','','',''])  t.add\_row(['','','','','',''])  for i in range(len(data.player\_name)):  if b.value[i] == 1:  t.add\_row([data.player\_name[i], data.position[i], data.raptor\_offense[i],   data.raptor\_defense[i], data.raptor\_total[i], "Bench"])  t.add\_row(['','','','','','']) print(t) |
| --- |

**Results**

*KEY: (Position: First Last - Total RAPTOR Score)*

Starters:

* G: Derrick White - 4.620
* G: Cameron Payne - 4.153
* F: Kawhi Leonard - 6.588
* F: Giannis Antetokounmpo - 7.812
* C: Nikola Jokic - 9.249

------------------------------------------

TOTAL RAPTOR SCORE: 32.422

Bench:

* G: Devonte Graham - 2.397
* G: Furkan Korkmaz - 2.170
* G/F: Luguentz Dort - 2.067
* F: David Nwaba - 3.181
* F: John Konchar - 2.326
* C: Daniel Gafford - 3.852
* C: Nicolas Claxton - 5.029

------------------------------------------

TOTAL RAPTOR SCORE: 21.022

**Post Optimality Analysis**

By weighting the objective function to potential playing time, we forced the program to consider players with expensive contracts, who may be worth the cap space. Without the weighted objective value, the optimal solution may have included less all-stars, and more players with small contracts and decent RAPTOR scores.

The salary cap constraint forced the program to fill the bench almost entirely with players under cheap or rookie contracts. A higher salary cap may have allowed the optimal solution to include even more players with higher contracts.

**Video:**

<https://youtu.be/B9hNHt37Xzk>

**GitHub:**

[https://github.com/nrezell/I SE3230FinalProject](https://github.com/nrezell/ISE3230FinalProject)

**Tasks Completed by Team Member:**

Austin: Variables and Obj. Function Creation, Full Presentation Layout

Ryan: Data Collection/Cleaning, Modeling Problem in CVXPY

Nate: Intro/Variables and Objective Presentation, Full Project Report