**Data Sources:**

Currently, projections are being grabbed from, <https://www.fantasysportsco.com/Projections#>.

This was done via javascript with window.open('/DesktopModules/DailyFantasyApi/API/Fantasy/ExportProjectionsCsv?sport=' + sportid + "&site=" + siteId + "&periodId=" + periodId), where the variables are 2, 1, and 459. The period id is the only thing that changes, and increments with the date.

The projections were chosen from this site due to the nicely formatted data. For past projections, there is a column for how much the player actually scored. I signed up for a free trial, and using the above javascript line, I was able to loop through all the dates since October to grab all the csvs. I believe it is because of the free trial, I was able to easily grab all the csvs. Since we have plenty of past data, I do not think a subscription to this site will be needed in the future. We’ll have to create a script to actually scrape this data instead of relying on their api if it comes to it.

With this data, optimize.py formulates solutions with the best optimal lineup for projected values, and as well as the scored values, creating underscores of \_P and \_S respectively. The \_S data should be used as our upper bound, and the ideal lineups.

Other data we have is in the xOldData. This data is currently not being used but could be in the future. An example is <http://rotoguru1.com/cgi-bin/hyday.pl?mon=11&day=7&year=2016&game=ds&order=box>. Since we have the scored value from fantasysportsco, their actual specific stats are not as necessary. For going forward, we might need to rely on this to grab the past scored and salary values if fantasysportsco becomes unfeasible to scrape. We also have 2015-2016.csv data which has all player data from last year.

DVP , defense per position is how much fantasy points teams allow per positions. We should use this to target players against weak defenses. This led to my attempt to “modified” projections to boost non super stars when playing weaker defenses and reduce their projections when playing top tier defenses. This is grabbed from, http://www.rotowire.com/daily/nba/defense-vspos.htm?site=DraftKings&pos=PG

**Optimization Process:**

Currently, the optimization is fairly straightforward. The main constraints are just to satisfy the player requirements, (number of players<=8, number of players per position, etc). The Diversity constraint is to force different lineups. If we claim diversity<=4, then lineup 2 cannot share more than 4 players line up 1. Lineup3 cannot share more than 4 players from lineup2 cannot share 4 players from lineup2. Increasing this constraint relaxes it. Diversity<=8 essentially does nothing as this says lineup2 cannot share more than 9 players from lineup1. Since the number of players total for a lineup is 8…this is saying lineup2 and lineup1 can share 8 players. Likewise on the opposite extreme, saying diversity<=1 would do the exact opposite.

A smaller lineup space should require less diversity, due to fewer number of high value lineups. Likewise a large lineup space where there are many games should have a higher diversity, due to multiple potential lineups at play.

**Additional factors to consider:**

-**Fatigue(**Target “fresh” players)

**-Streaks(**Target hot players)

-**Sensitivity** of non stars(Could filter based off salary, as my current raw attempt)

-**Vegas odd**s: can be grabbed from <http://www.sportsbookreview.com/betting-odds/nba-basketball/?date=20161211>

-Targeting higher pace games means more possessions and more points.

-**Risk:** Younger players have less data, and can explode on any given day. We can calculate risk via std as my current attempt, using the scored data that we grab from fantasysportsco. We could used the entire dataset from 2015-2016 to calculate the std, however this would not be accurate for younger players. Younger players, especially rookies, and sophomores, have higher jumps in improvement so it might be more accurate to use this years data for younger players.

**Defense:** Target weaker defenses

**How to run:**

Here’s an example of how to run the optimize script. Use these lines at the end of the opimitze script and comment out anything else at the bottom. This would generate 1 projected lineup for the date Dec 13. Modified is if we use modified projections, I will replace this later on. The diversity constraint is hardcoded, but for the future, should used as a parameter, depending on the number of games in a given night.

For multiple dates, just make an array of dates and a for loop. The script then builds a csv in the prediction folder. If you set projected\_lineup to False, it uses the scored column to generate the perfect lineups.

projected\_lineup=True #

#date='Dec132016'

iterations=1

modified=False

optimize(projected\_lineup, date,iterations,modified)

**Daily Data grab:**

On a betting data we should

-Grab projections.

-Grab updated DVP positions

-Grab vegas odds for the night(optional)

-Regrab projections right before games start incase of injuries/players resting