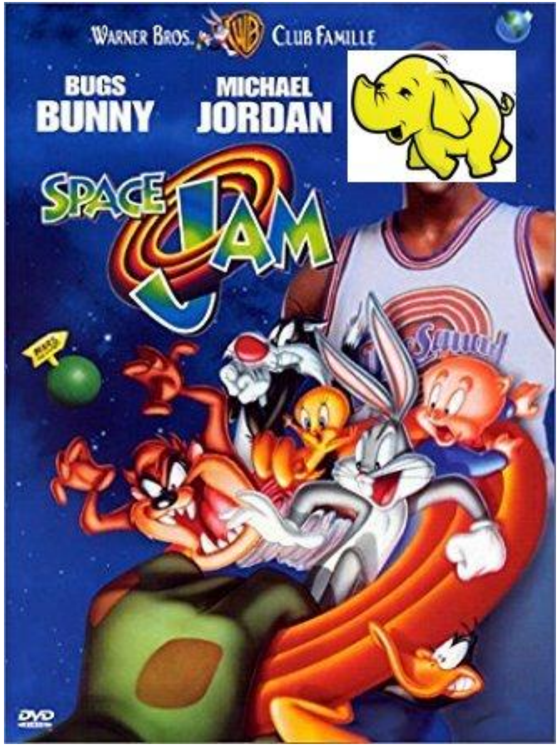
Hadoop Dreams



Evan Smothers

Overview

- I wrote an algorithm to help me win at daily fantasy basketball
- This side project was my intro to data science, and helped me land my first job
- Companies love side projects: they show initiative and can be the only data point for new grads
- We’ll talk about how to choose a project that interests you, then execute and deliver a finished product
- We’ll also go through my daily fantasy project along the way
- (No, you don’t need to know anything about basketball)

About the author

1850 MELIORA UNIVERSITY of ROCHESTER

UC DAVIS AGGIES

here

THEOREM

Uber

1990-2008	2008-2012	2012-2017	2017	2017	2018 -

Fantasy basketball

Choosing the right problem

- What are your interests?
- What are problems from these areas that could benefit from a statistical model?
- Which of these problems have a reasonable scope? (If none, you need to simplify the problem)

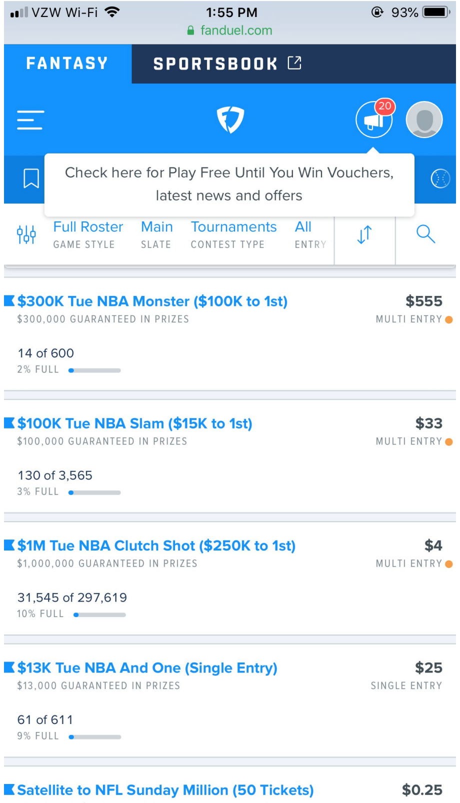
Fantasy basketball primer

- Typical fantasy basketball
  - A **league** with a number of different **teams.** Single **draft** at the beginning of the season where NBA players are selected for teams
  - Each player may only be on one team
  - Teams are **static**
  - League lasts all NBA season
- Daily fantasy basketball
  - **No draft**. Each participant chooses their team **independently** every day
  - A single player may be on **many teams**
  - No permanent teams: participants choose **new teams every day** (or choose not to play)

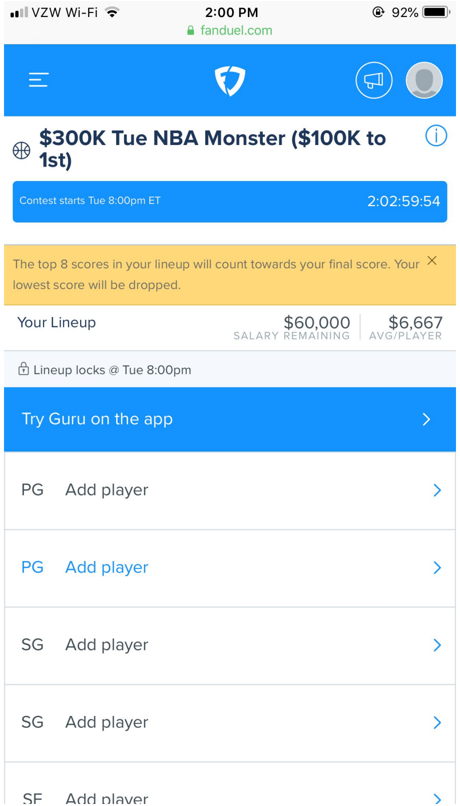
More fantasy basketball details

- Sites like Fanduel and DraftKings allow participants to enter lineups each day
- There are many contests each day with varying
  - Payout structures
  - # of participants
  - Entry fee
  - # of lineups per person
- Not all NBA teams play every day. This means the **available players change each day**

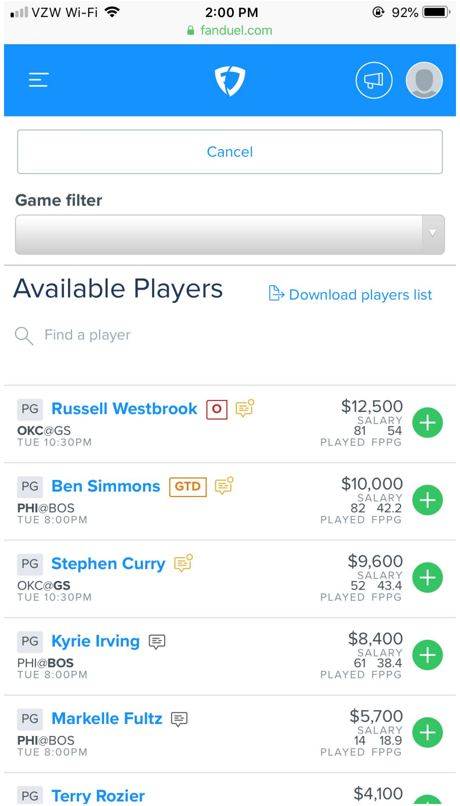
In practice



Contest selection screen



A single contest's page



Player selection screen

Defining the problem

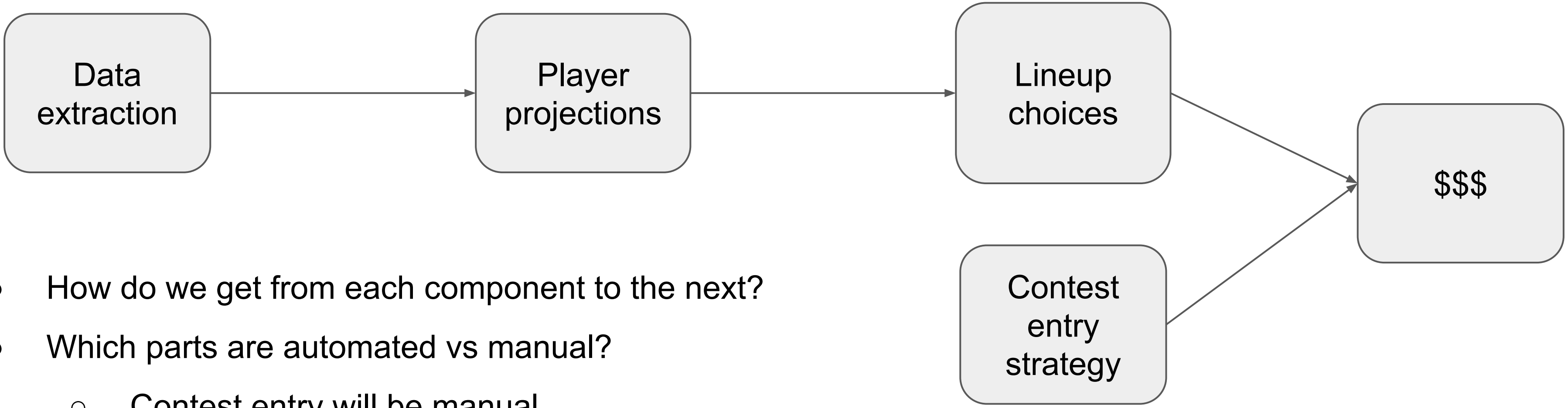
- Version 0: “Build a model to make money playing daily fantasy basketball” (too broad)
- Ask follow-up questions:
  - What are we modeling?
    - Individual player performance, then optimize lineups based on player projections
  - How do we determine if the model is successful?
    - Net profit is one way, but a lot of intermediate variables
    - For the model itself, cross-validate with historical data
  - Is there existing work? If so can we leverage it?
    - Yes, but generally not freely accessible
  - What data will we use? How will we acquire it?
    - basketball-reference.com, ESPN, Vegas lines, depth charts ... Lots of scraping

Now what?

?

Data → $$$

Breaking into subcomponents



- How do we get from each component to the next?
- Which parts are automated vs manual?
  - Contest entry will be manual
  - Lineup optimization will be semi-manual
  - Everything else should be automated
- How do the components interact?
  - Lineup optimization uses player projections as an input
  - Projections require feature extraction from data sources
  - Data extraction should also include performances in previous day’s games (for response variables)

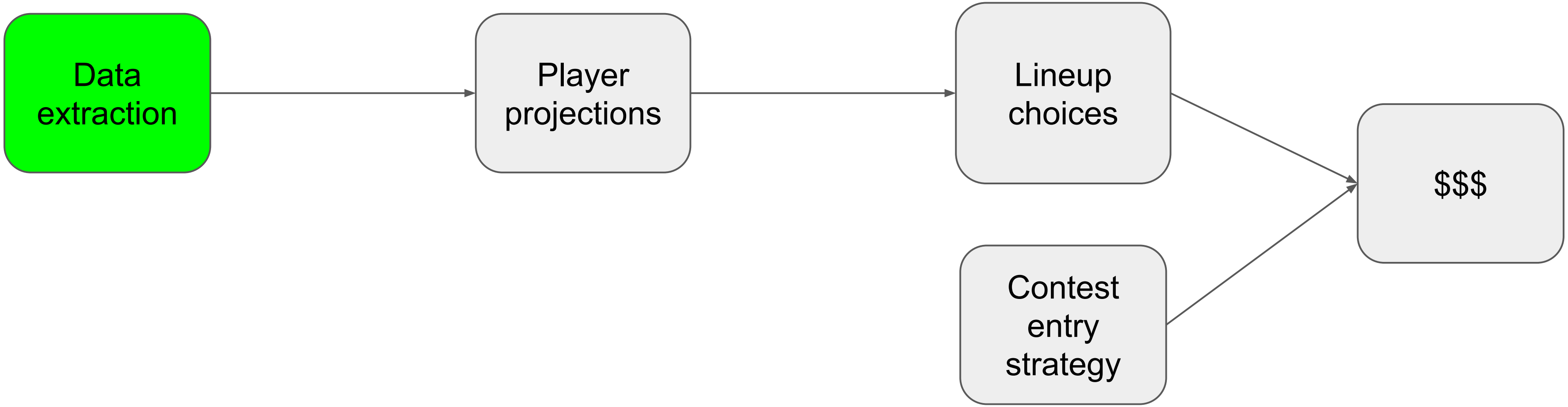
I/O

- What are the inputs and outputs of each component?
  - Data extraction
    - Inputs: CSV of day’s active players
    - Outputs: pandas dataframe containing relevant player features (to be defined) and responses
  - Player projection model
    - Inputs: pandas dataframe of features, historical data with responses for training
    - Outputs: projected player fantasy points for that day’s games
  - Lineup optimization:
    - Inputs: projected player fantasy points, other constraints
    - Outputs: a “winning lineup”

- To simplify our work later, the output of one step should be very similar to the input of the next step

Other considerations

- What is the size of the data?
  - On the order of megabytes, so can be stored locally (otherwise use AWS or Google Cloud)
- What is our compute power? Does this affect our model choice?
  - Running locally on a 2012 Macbook Pro
  - As long as we don't try using deep learning we should be OK
- How should we schedule our jobs?
  - Contests begin at 4PM daily, so the script should be run daily in the afternoon.
- What's an acceptable runtime? Can we design to minimize it?
  - Ideally under an hour end-to-end (in case of last-minute lineup changes it may need a refresh)
  - Scraping takes the longest. Since the data size is small, we can store historical data locally to cut down runtime

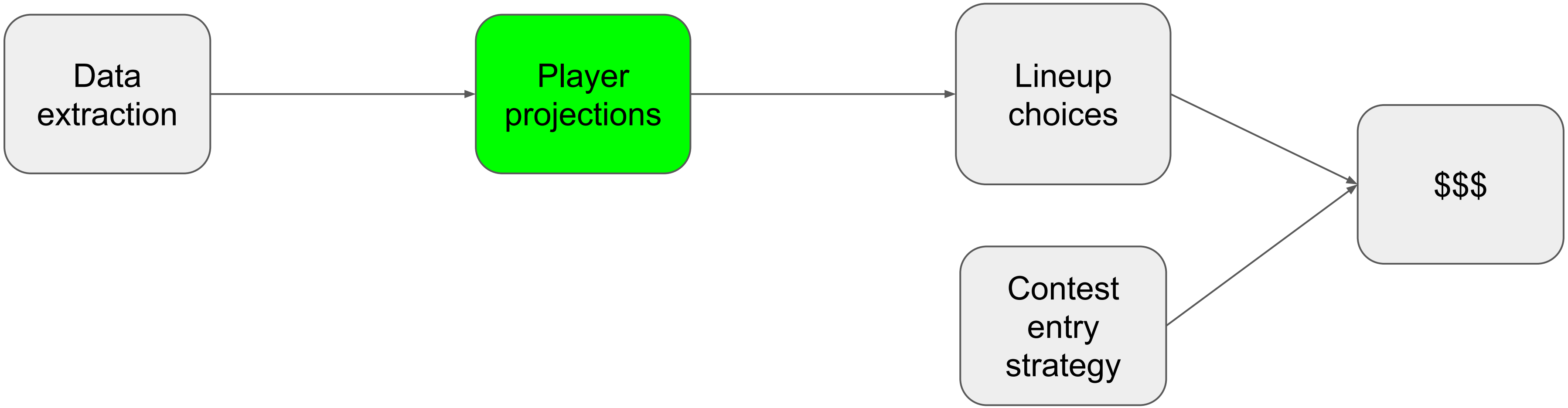

Data extraction
Player projections
Lineup choices
Contest entry strategy
$$$

Data extraction: key questions

- What is the purpose?
  - Obtain all relevant features for predicting a player’s performance in a given game
- What is the approach?
  - Scrape a bunch of different websites
- What are the tools?
  - urllib, BeautifulSoup
- What are potential issues?
  - Extraction for a given day will only contain features, but to train we need to match to responses.
    - Solution: store historical features locally, then match to responses on subsequent runs

Choosing the right features

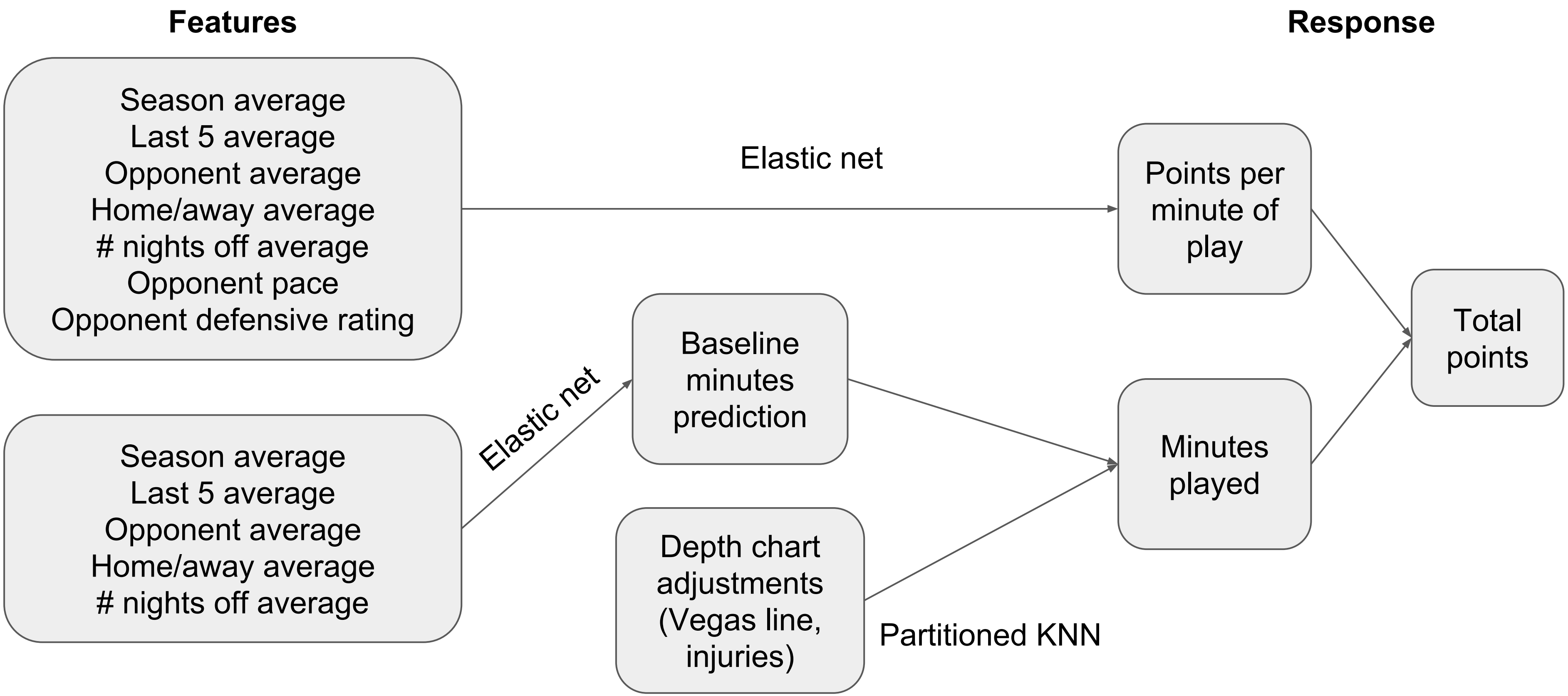
- What have other people done already?
  - Common features include a player's season averages or their averages over the past *n* games
- What is missing in current approaches? (This is where domain knowledge comes in handy)
  - Is the game home or away?
  - # of nights rest
  - Pace of the opposing team (relative to the player's own team)
  - Defensive rating of the opposing team
  - Injuries/depth charts
  - Over/under and Vegas lines (combined with depth charts)

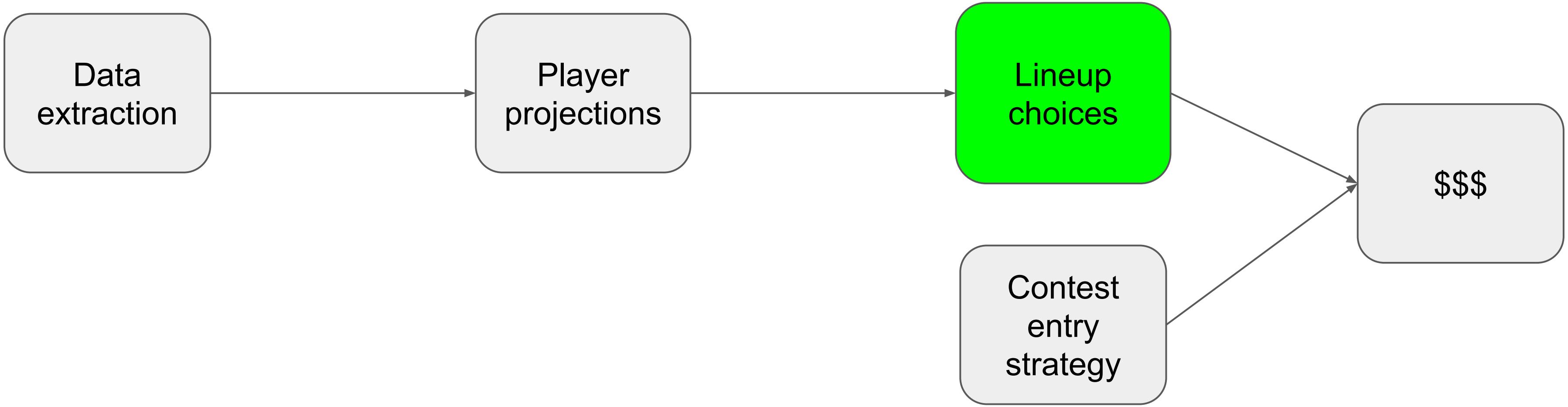

Data extraction
Player projections
Lineup choices
Contest entry strategy
$$$

Player projection model: key questions

- What is the purpose?
    - Predict how many fantasy points a player will score in a given game
- What is the approach?
    - First version: throw all features into a linear model
    - Second version: split the problem into pieces
        - Predict how many minutes a player will be on the floor
        - Predict how many points the player will score each minute
- What are the tools?
    - sklearn: linear model and knn (nothing fancy)
- What are potential issues?
    - A lot of highly correlated features
    - Injuries to teammates are highly predictive, but very difficult to model

Choosing the right model




Data extraction
Player projections
Lineup choices
Contest entry strategy
$$$

Lineup optimization: key questions

- What is the purpose?
  - Given a projection for each player’s fantasy points, find a lineup that optimizes the total expectation while still satisfying all constraints
- What is the approach?
  - Create dummy variables for whether players are in the lineup, cast as an integer programming problem
- What are the tools?
  - Excel solver (an integer programming module)
- What are potential issues?
  - The optimization problem may require additional constraints that we’re unaware of
  - Relies heavily on accuracy of player projections

Optimization in action

Before

Play?	Name	Position	Team	Salary	EFV	EFV/Cost
0	Pau Gasol	C	CHI	9300	38.15605886	4.102802028
0	Damian Lillard	PG	POR	9300	41.22359315	4.432644424
0	Carmelo Anthony	SF	NY	8900	37.39291836	4.201451501
0	Hassan Whitesid	C	MIA	8400	39.86827823	4.746223599
0	Draymond Greer	PF	GS	8400	39.08588843	4.653081956
0	Brook Lopez	C	BKN	8400	44.70223064	5.321694124
0	Kemba Walker	PG	CHA	8200	34.50620984	4.208074371
0	Nikola Vucevic	C	ORL	8100	35.73126175	4.411266883
0	Al Horford	C	ATL	7900	35.21979837	4.458202325
0	Paul Millsap	PF	ATL	7700	33.58379422	4.361531717

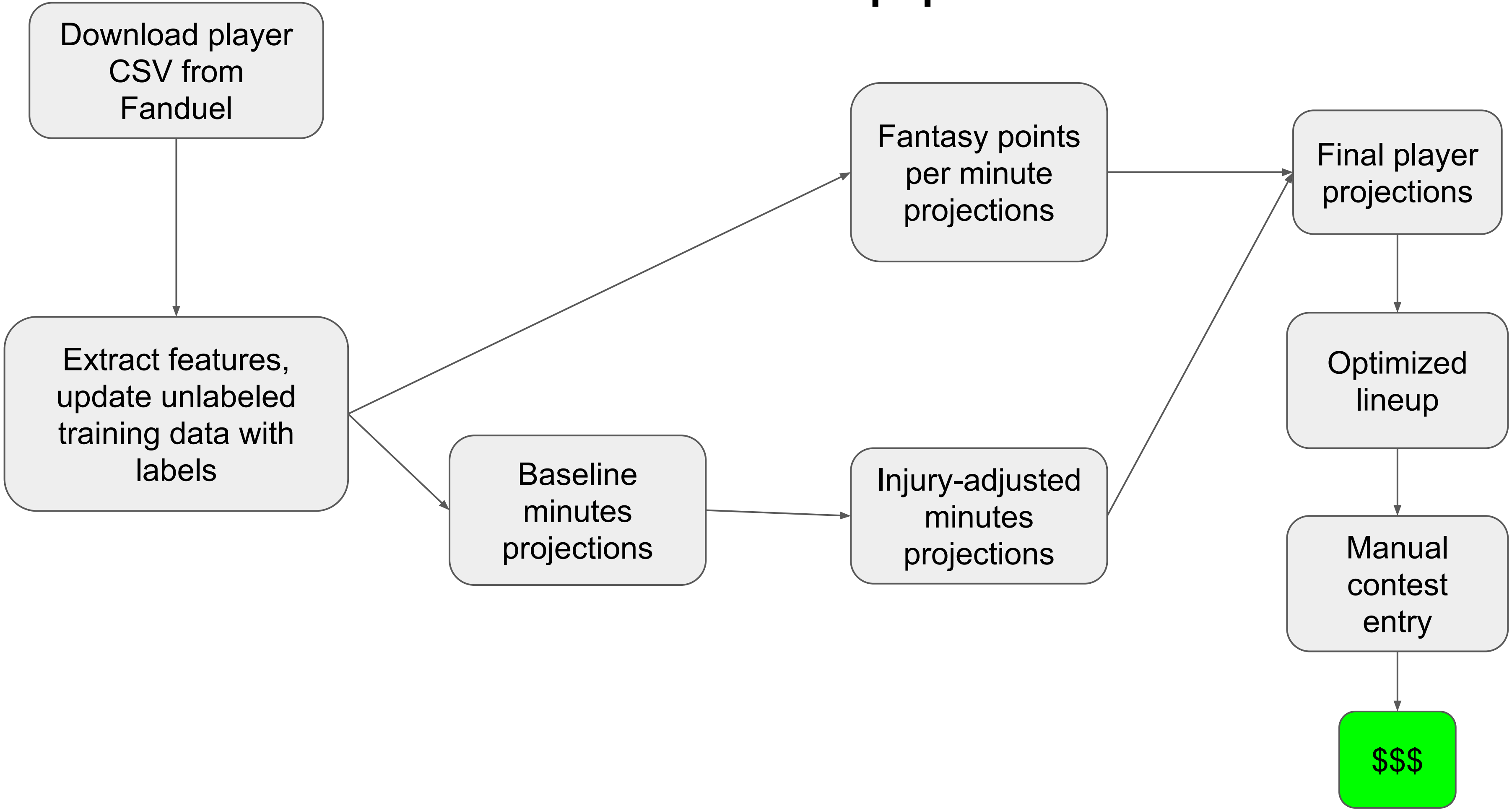
After

Play?	Name	Position	Team	Salary	EFV	EFV/Cost
1	Damian Lillard	PG	POR	9300	41.22359315	4.432644424
1	Carmelo Anthony	SF	NY	8900	37.39291836	4.201451501
1	Brook Lopez	C	BKN	8400	44.70223064	5.321694124
1	Thaddeus Young	PF	BKN	7000	36.07059219	5.152941742
1	Jordan Clarkson	SG	LAL	6000	33.59326208	5.598877013
1	Julius Randle	PF	LAL	6000	32.50687979	5.417813298
1	Deron Williams	PG	DAL	5700	26.90406364	4.720011164
1	Jeremy Lin	SG	CHA	4400	26.78220737	6.086865311
1	Harrison Barnes	SF	GS	4200	22.96604918	5.468106948

	Current	Required
Players	9	9
Salary	59900	60000
EFP	302.1417964	
PG	2	2
SG	2	2
SF	2	2
PF	2	2
C	1	1
Teams		
ATL	0	2
BKN	2	2
BOS	0	2
CHA	1	2
CHI	0	2
CLE	0	2

- Constraints:
  - Cannot exceed maximum salary
  - Must satisfy positional requirements
  - No more than 2-3 players from a given team (heuristic)

The finished pipeline



Wrapping up

- Most worthwhile data science projects require substantial amounts of effort.
- Scoping out and properly defining a problem in advance is very important.
- Real-world problems are ambiguous and open-ended. To deal with this:
  - Break large problems into manageable subcomponents
  - Continuously ask yourself clarifying questions
- And don’t forget, open source libraries are your friend!

Thank you!!!