Fantasy Fantasy Football

Willie Boag
MIT CSAIL
wboag@mit.edu

Abstract

Fantasy Football is America's favorite pretend pastime. Friend groups of all varieties[1] *play it, though we all have that asshole friend Jason that never sets his lineup. It is interesting that although most fantasy football platforms allow for computer-aided "auto-draft", there is little-to-no support for "auto-manage my entire team all season." Such a notion – bots playing fantasy football – is a great idea, and I'll tell you why. One can create an entire league of bots playing against one another using different policies. In fact, why stop at one league? Let's create hundreds of leagues of bots, and then draft those bots! It'll feel just like we're playing fantasy football, but without all of the undesirable overhead; we'll be playing* ***fantasy*** *fantasy football!*

Figure 1. The case for thinking bigger.

1. Introduction

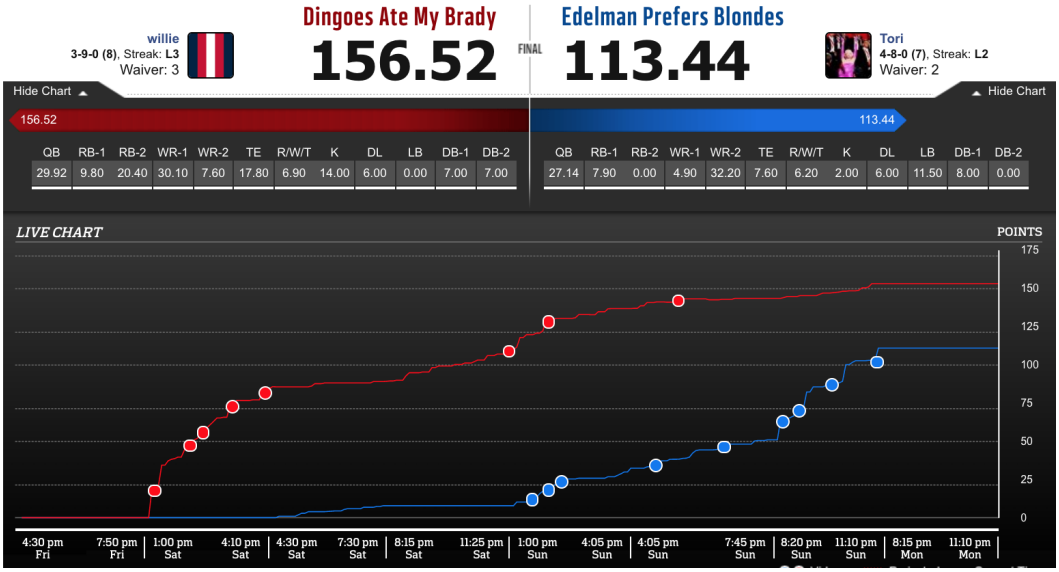
North America Football ("Football" for short in this paper) is a sport popular in the United States of America. It is a contact sport where each team takes turns trying to move the ball across the field to "score" on the other team. They get a couple tries, and if they don't make it all of the way, they can give up and just try kicking it the rest of the way. It sounds silly, but it's not.

Football is very dangerous. Not only do players get "normal" injuries like broken legs, broken backs, broken arms, torn ACLs, etc, it is actually much worse. Repeated hits to the head cause players to develop Chronic Traumatic Encephalopathy (CTE), a neurodegenerative disease that can present itself with behavioral problems, mood problems, and problems with thinking [4]. This is very serious, especially given that the players are overwhelmingly black, resulting in an exploitative practice that people probably shouldn't watch if it isn't part of their cultural and social connections to others [2]. On top of that, the National Football League (NFL) spent years covering up and distorting the science that they are profiting off players destroying their futures for the game [3].

For football fans, however, there are ways of getting more "engaged" beyond watching the games on TV. The online service *Fantasy Football* allows fans to build rosters by "drafting" players from different teams to be on their virtual team. Each player gets fantasy points for doing well in their game (e.g. a 30 yard catch is worth 3 fantasy points, fumbling the ball loses 2 fantasy points, etc). Each week, two virtual teams go head-to-head, where each team's score is the sum of the fantasy points of each individual player on their team. This

[1] just kidding, though. 80% of all team managers are male. 89.8% of them are white and 51.5% are not married. 78% hold a bachelor's degree or higher, making the majority of team managers an upper middle class earner with a median income between 60-100K. It's really not that diverse.

Figure 2. Each week, two teams face off each other.



allows you to feel like you are involved in the league (and gives a good excuse to learn about players on teams other than the one you care about).

Fantasy football offers a way for inexperienced users to draft a team even if they aren't sure what they are doing. However, throughout the season, users must manage their team – performing tasks including: starting/benching given players, adding/dropping players from free agency, trading with other users, and trash talk. Unfortunately, there are no commercial tools available for auto managing a team. This presents a challenge to people who don't care about fantasy football.

In this work, we:

1. Demonstrate a proof of concept for Fantasy Fantasy Football, where bots play fantasy football, and we draft the bots

2. Apply state-of-the-art NLP tools to generate trash talk

2. Methodology

Using data from Pro Football Reference (PFR), we simulate a virtual fantasy football year. In this section, we describe the mechanics of this code and additionally investigate advanced NLP techniques that allow us to auto-generate trash talk.

3. Fantasy Football simulation

To simulate one year of the league, we employ many Player, FantasyTeam, and Agent object.

3.1. Player

We begin by scraping the data from PFR. The data is formatted like this `http://www.pro-football-reference.com/players/B/BreeDr00/gamelog/2009`. The code to scrape this data can be found at `https://github.com/wboag/fantasy/tree/master/data/build`. There are scripts to download league-wide stats, player stats, and bye information for a given year. This assumes the set of team names and cities from the 2002 realignment.

The player object, defined in `https://github.com/wboag/fantasy/blob/master/code/player.py` parses the data downloaded above and applies the fantasy football rules to convert their stats from

Figure 3. Wes Welker is the best New England Patriot Wide Receiver in history. He has more 1st Team All-Pros (2) than all of the other NE WRs combined (1).

each game into weekly fantasy football scores. These weights can be found in the *Settings* section in your fantasy.nfl.com league. Each player has a position (e.g. QB, WR, DL) associated with them.

3.2. FantasyTeam

A fantasy team is a collection of players. There are "active" players[2] and "bench" players on a given week. In the code `https://github.com/wboag/fantasy/blob/master/code/ff_team.py`, we can see that there are interfaces to add/drop players from the team and to activate/deactivate players from the active lineup. This interface makes it easy to see how many fantasy points a given lineup of players scored for a given week.

3.3. Agent

Here is where the magic happens. By creating an Agent object, we allow bots to be able to play fantasy football! Agents are able to decide which players to draft (given their current needs and the remaining players available), which players to add/drop from the weekly pool of available agents (i.e. "waiver wires"), and set their lineup. We define the agent object at `https://github.com/wboag/fantasy/blob/master/code/agent.py`.

In order to decide which players the agent wants, they look to 2 key ingredients:

1. The historical data of all available players from the prior year (e.g. if playing the 2016 season, then using the 2015 data as a basis)

2. Agent-specific "parameters" which allow them to make different decisions from each other.

The agent's parameters are randomly generated. There are 2 kind of parameters: stat-based (i.e. how much that agent cares about passing, rushing, fumbles, etc) as well as history-based (i.e. how much the agent cares about the last month of performance from their players). As you might imagine, better agents are more likely to weight important stats high and (probably) put more weight on recent performance than old (stale) performance. Figure xx shows how one can think of an agent as parametrized by a set of these values, where agents with similar weights would behave similarly and would be nearby in a high-dimensional vector space.

3.4. League

A simulation of the league creates 10 agents, executes a draft, and simulates 17 weeks of football games. Agents are able to add/drop players and update their rosters. The code for this can be found at `https://github.com/wboag/fantasy/blob/master/code/league.py`.

[2]QB, WR1, WR2, RB1, RB2, TE, FLEX, K, DB1, DB2, LB1, LB2, DL1, DL2

Figure 4. Similarly parametrized agents would be nearby in a vector space.

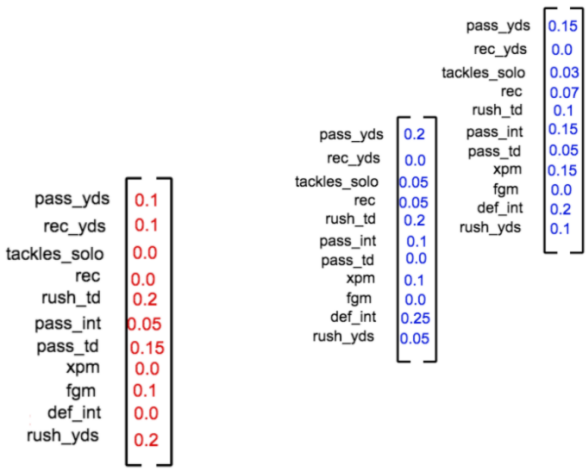


Table 1. History weights for the league shown in Figure 7.

Agent	1 wk ago	2 wks ago	3 wks ago	4 wks ago
Tara	.07	.32	.11	.50
Willow	.26	.21	.34	.18
Spike	.10	.35	.40	.15
Dawn	.43	.12	.06	.39
Xander	.11	.39	.29	.20
Oz	.37	.35	.22	.06
Buffy	.06	.21	.54	.18
Angel	.07	.36	.23	.33
Anya	.39	.29	.31	.02
Giles	.07	.23	.36	.35

Because each agent is randomly intialized, every run of the league script produces different results. Figure 7 shows a snapshot of weeks 15-17 from one run. We can see that some teams like Tara and Willow performed very well, whereas Giles and Anya won less than a quarter of their games. Table 1 shows the history-based weights for each agent, though it ignores the stats-based weights (71 vals). There does not seem to be an obvious trend between successful bots and history-based feature weights. This could be because stats-based weights are more valuable (i.e. players are mostly consistent, so its more important to care about QB passing as opposed to knowing to look at last week vs 4 weeks ago) though it could also be because of randomness in the league pairing. On a given week, the agent doesn't need to score a lot, they only need to score more than the agent they were randomly matched against; sometimes agents get lucky.

In order to assess what factors make a successful agent, we would need to run hundreds of leagues, and try to identify which kinds of agents perform the best.

4. Auto Trash Talk

In this section, we use GPT-2 [1] to generate trash talk. The demo for this tool can be found at `https://transformer.huggingface.co/doc/gpt2-large`. We can see the trash talk shown in Figure 6.

Figure 5. GPT-2 Trash 2 Furious.

Your team is so bad that the world doesn't even notice you. You've always been good at hiding your weaknesses, but the world is aware of them and they don 't take them seriously. You're such a pain in the ass that even your own friends don ' t take you seriously.

All things considered, this trash talk is pretty good.

5. Future Work

There are some areas of work that we would like to improve in the future. For one, the current agent drafting algorithm does not know how to evaluate rookies for that year (because it looks at the previous year's performance to assess player quality). Additionally, we need to get around to adding James O'Shaughnessy to the league, per the open challenge defined in Figure 6.

6. Acknowledgments

I'd like to thank Tom Brady, Wes Welker, Danny Amendola, Devin McCourty, and James White.

References

[1] Better language models and their implications.
[2] W. Boag. Revisiting past decisions: Im going to try watching football again.
[3] M. VASILOGAMBROS. The nfls concussion cover-up.
[4] Wikipedia. Cte wikipedia.

Figure 6. Parsing is difficult.

```
def read_name(text):
    #regex = "<title>([\.a-zA-Z'\s]+) \d+ Game Log | Pro-Football-Reference.com</title>"
    regex = "<title>([^|]+) \d+ Game Log | Pro-Football-Reference.com</title>"
    match = re.search(regex, text)

    # NOTE - fuck James O'Shaughnessy. we'll throw an exception and exclude him.
    '''
    if not match.groups()[0]:
        regex = "<title>([^|]+) Stats | Pro-Football-Reference.com</title>"
        match = re.search(regex, text)
    '''

    return match.groups()[0].strip()
```

Figure 7. Results from weeks 15-17 of one league.

```
week 15
         1. Tara     (12-3-0)
         2. Willow   (11-4-0)
         3. Spike    (9-6-0)
         4. Dawn     (9-6-0)
         5. Xander   (8-7-0)
         6. Oz       (8-7-0)
         7. Buffy    (6-9-0)
         8. Angel    (5-10-0)
         9. Anya     (4-11-0)
        10. Giles    (3-12-0)



week 16
         1. Tara     (13-3-0)
         2. Willow   (11-5-0)
         3. Dawn     (10-6-0)
         4. Xander   (9-7-0)
         5. Spike    (9-7-0)
         6. Oz       (9-7-0)
         7. Buffy    (7-9-0)
         8. Angel    (5-11-0)
         9. Anya     (4-12-0)
        10. Giles    (3-13-0)



week 17
         1. Tara     (14-3-0)
         2. Willow   (12-5-0)
         3. Xander   (10-7-0)
         4. Dawn     (10-7-0)
         5. Spike    (9-8-0)
         6. Oz       (9-8-0)
         7. Buffy    (8-9-0)
         8. Angel    (5-12-0)
         9. Giles    (4-13-0)
        10. Anya     (4-13-0)
```