Automated DraftKings DFF Roster Optimizer Utilizing Ridge-Regression and Integer Programming

Maharshi Thakker[1], John Gearheart[1], and Sakib Salim[1]

[1]Fu Foundation School of Engineering and Applied Science, Columbia University

Abstract

The capability to model future individual player performance and optimize the overall expected points total of a composed Daily Fantasy Football roster is integral to a DFF Manager's success. While several commercial products and Daily Fantasy Game Websites themselves offer suggested lineups to "optimize" a Manager's roster, it is unknown whether the supplied data is skewed in favor of the gaming host. This paper explores automated roster optimization using Ridge-Regression analysis and Integer Programming Optimization as an alternative to the projections supplied by DraftKings.

1. Introduction

Sports gambling has rapidly grown over the past few years since the Supreme Court overturned the federal ban in 2018. One avenue of growth, in particular, is Daily Fantasy Football (DFF) where the goal is to roster the best lineup of football players for a given week of the NFL season. With close to 300 available players and defenses in any given week, the number of possible roster combinations is around 3.8×10^{17}.

According to a set formula, each player's point total is directly derived from real-life performances[1]:

$$FP = 0.1 * \Sigma(RushYards, ReceivingYds) + \#Receptions + 0.4 * PassingYards + 6 * \Sigma(RushTD, ReceivingTD) + 4 * PassingTD + - 2 * \Sigma(Interceptions, Fumble) + TwoPTConv$$

The catch comes in two types of lineup restraints: positions and salaries. Under the classic format, a valid lineup must have 1QB, 2RB, 3WR, 1TE, 1 DST, and 1 FLEX (an RB, WR, or TE). Additionally, the vendor assigns salaries to each player: better, more consistent players are generally more expensive. The individual is tasked with creating the highest-scoring valid lineup with a total salary of less than $50,000. One can then submit their lineup to any number of different cash competitions. The general idea is that your lineup is pitted against other player's lineups and the best performing lineup wins the cash pot.

1.1. Background and Related Work

Many DFS Hosting and Analysis websites offer their proprietary projection models for NFL players, claiming to be better than the rest. From sites like FantasyPros, which create an amalgamation of projections from "leading experts"[2] in sports analytics, to GridironAI, which boasts a 22% improvement on leading sites such as ESPN and Yahoo! Sports[3], there are many commercially available models to choose from in the marketplace. Academically, teams at Columbia University and Stanford University have applied Machine Learning to project player performances in other sports arenas[4,5], but none have created an approach for Daily Fantasy Football. Although limited sample sizes of player data have been cited as a rationale for avoiding the sport of football[4], an increase in publicly available NFL datasets has created an opportunity for this team to generate a tool for the automated creation of a DFS roster as well as explore the prediction power of Machine Learning algorithms within DFS.

2. Methods

Our overall goal is to create an optimal NFL fantasy football roster that maximizes the expected Fantasy Points our team will earn. Before we can select our players, we need to estimate their individual expected Fantasy Points. We hypothesize that a professional player will generally have consistent game performance. Therefore we believe that we can use past performance data to predict future performance. Additionally, we believe that players exhibit recency bias and that their most recent games indicate their future performance.

2.1. Model

To predict any individual player or defense's next game performance, we decided to use a Ridge Regression model[8]. We chose this model because the penalty term from ridge regression prevents us from prioritizing any one game.

Initially, we chose to use the past four games to predict the next game's fantasy points and a simple model.

$$FP_N = \sum_{i=1}^{4} \alpha_i FP_{N-i}$$

While this was a good starting point, we realize this was too simplistic as Fantasy Points are calculated can vary greatly depending on how well a player does in each game metric. We decided to refine our model to predict the individual game metrics since Fantasy Points are ultimately a formulaic combination of the game metrics. Using this method, we would be able to obtain more accurate predictions.

$$FP_N = \sum_{j=1}^{J} \beta_j \, GM_{N,j}$$

$$GM_{N,j} = \sum_{i=1}^{4} \alpha_i GM_{N-i,j}$$

While accounting for individual metrics did reduce prediction error, we realized there is likely a strong correlation between player position and the type of points they receive. For example, a Wide Receiver is generally much more likely to receive points for a Receiving Touchdown than a Tight End because Wide Receivers are a purely offensive position. In contrast, Tight Ends also often function as linemen and have fewer opportunities to receive the ball.

To account for player position fairly, we use a Onehot Encoding to convert the categorical variable, "positions," to a binary feature vector, which takes a value of 1 when it corresponds to the player's position and zeroes for the rest of the positions.

$$GM_{N,j} = \sum_{i=1}^{4} \alpha_i GM_{N-i,j} + \sum_{k=1}^{K} \gamma_k Pos_k$$

The model was trained on the past five years' worth of weekly game data, using rolling four week periods. We excluded any data that failed to have a complete four-week history for simplicity.

2.2. Data Collection and Processing

The ultimate goal of data collection as an input to our Ridge-Regression model was to compile a list of all available players as well as their corresponding DraftKings Salaries and projected point totals.

This first component collected for this dataset were this week's available players, their field positions ('Pos'), and their individual salaries ('Cost'). FootballDieHards[6] hosts a public dataset of historical and current DFS salaries from DraftKings. Our team wrote Python code to scrape this information from FootballDieHards.com as the following format into a first dataframe:

['Player', 'Position', 'Cost']

A second scraper script was written to field corresponding historical player performance data for the available players from Statheads.com[7]. Statheads Football presents their dataset in the following table structure:

['Player', 'Age', 'Date', 'League', 'Team', 'Opponent', 'Game Result', 'Week', 'Day', '2PointConversionsAttempted', '2PointConversionsCompleted', 'Safeties', 'Points Against', 'Fumbles', 'FumbleRecoveries', 'FumbleRecoveryTouchdowns', 'Interceptions', 'PassYards', 'PassTDs', 'RushYds', 'RushTDs', 'Receptions', 'ReceivingYards', 'ReceivingTDs', 'Sacks', 'InterceptionTDs', 'PuntReturnTDs', 'PointsAgainst']

This information was pulled into a second dataframe where the columns irrelevant to fantasy point totals were dropped, creating a dataframe containing each player's fantasy performances binned by week of the season as in the following example for Rushing Touchdowns:

['Player', 'Position', 'Week', 'RushTD_N', 'RushTD_N-1', 'RushTD_N-2', 'RushTD_N-3', 'RushTD_N-4']

As the ridge-regression model operated on the training data by minimizing the error in predicting the fifth performance based upon the past four performances for each fantasy stat, the player performance data was segmented to create a dataframe for each fantasy point generating stat with a row for each player with five weekly performances in that stat for the season and the last five performances. The projected fantasy points for the upcoming week for each player and defense were appended to the dataframe containing salary information and the result was passed to our roster optimization module.

['Player', 'Position', 'Cost', 'Expected Fantasy Points']

2.3. Roster Optimization

Our model outputs point predictions, which we would like to use to develop our lineup. We can formulate an optimization problem where we seek to simply maximize our lineup's point total given restraints on the salary and the number of players in each position. Given a set of players P, there is a disjoint covering of P where each set in the covering represents the different possible positions: (Q, R, W, T, D).

The decision variables represent the binary inclusion of a player in the lineup. This system is an ILP with only a few constraints:

$$\begin{aligned}
\text{maximize} \quad & \sum f_i x_i \\
\text{subject to} \quad & \sum s_i x_i \leq 50000 \\
& \sum_{p_i \in Q} x_i = 1 \\
& 3 \leq \sum_{p_i \in W} x_i \leq 4 \\
& 2 \leq \sum_{p_i \in R} x_i \leq 3 \\
& 1 \leq \sum_{p_i \in T} x_i \leq 2 \\
& \sum_{p_i \in D} x_i = 1 \\
& x_i \in \{0, 1\}
\end{aligned}$$

Even with a large pool of players, it is tractable, and we expect to achieve an optimal solution using standard branch and bound methods. The output is the optimal lineup given our predictions.

3. Results and Discussion

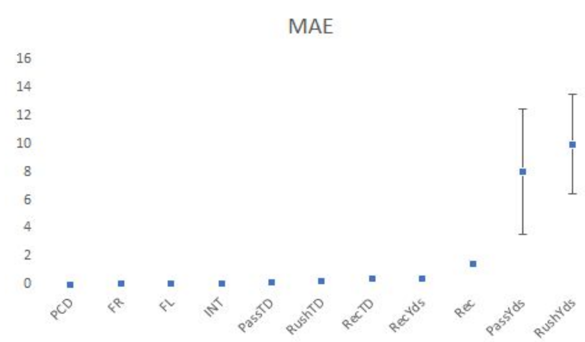

Figure 1 *MAE vs. Standard Deviation for overall game metrics*

On a high level, we can see that overall the Mean Absolute Error (MAE) and standard deviation for the predicted vs. actual game metrics were relatively low (Figure 1), suggesting that our model does a good job predicting future points. However, we indicated that we believe there is a correlation between game metrics and position.

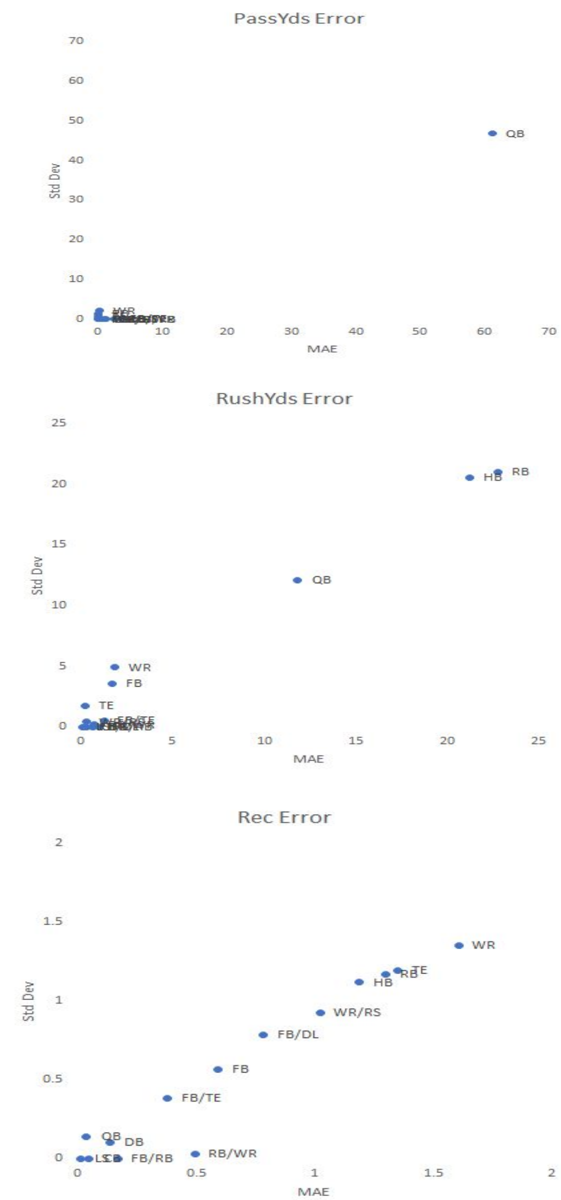

Figure 2 *MAE vs. Standard Deviation for individual game metrics given a player's position*

For some metrics, the model does a fairly good job predicting future values. For example, for PassYds, the model is very accurate for all positions except for QBs. This makes sense because most other players do not score PassYds points and QB game strategy can vary widely given the current circumstances. Therefore we expect most of the error and variability to lie with QBs.

However, the model has a relatively more considerable degree of error and variation amongst all positions for other metrics. For RushYds, this is expected because QBs and RBs are generally the only players that go for RushYds. RushYds are RB's primary source of points. However, there is a much more extensive range in outcomes, so we should expect to see a larger degree of error/variation. On the other hand, most QBs generally attempt very few rushes and have a relatively much smaller range in possible RushYds (say 0-30), so there is less variation. For Receptions, while there isn't an identifiable distribution of predicted player points, the error is small for all positions, so it may not matter.

While our model may not be as precise, if the error ultimately does not significantly impact the total Fantasy Points, it is sufficient.

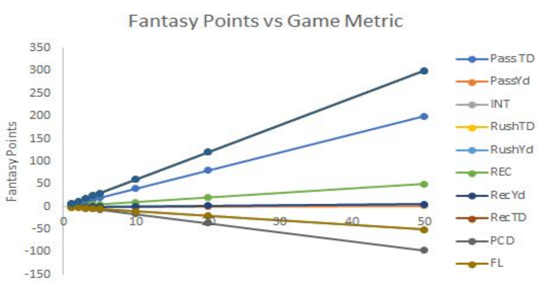

Figure 3 *Game metrics with a larger slope indicate a more significant impact on change in Fantasy Points.*

Based on the game metric contributions, FR, PassTD, and REC affect the overall Fantasy Score the most, emphasizing accuracy on those metrics.

After optimizing our lineup based on our predictions, we would like to evaluate our lineup's quality. The ILP permits an exact solution given a particular set of projections. We note that every week there is at least one globally optimal lineup. The pool of players per position is typically between 30 and 100. So the combinatorics of the problem is such that we are looking at a vast number of possible lineups (on the order of 10^{23}). Finding the globally optimal lineup is not at all feasible.

Method	Week 14, 2020
Ridge-Model	141.22pts
DraftKings	138.66pts
FantasyPros	133.54pts

Table 1 *Our model performance vs. online applications for Week 14 of the NFL 2020 Season*

We instead compare our lineup with the optimal lineups formed using "expert" predictions from different sources such as the vendor, DraftKings, and leading paid fantasy football services such as FantasyPros. Having run our model on Week 14 of the 2020 NFL season, we note that our lineup performs on par with DraftKings and, in fact, outperforms FantasyPros.

After evaluating our methods against commercial Fantasy optimizers during a single week of the NFL season, the team leveraged available player performance data for the rest of the 2020 season to compare our "optimized" rosters against the optimal team composed using DraftKings' predictive performance algorithm and identify if our ridge regression performance window size could be further optimized.

		Ridge-Regression Roster Actual Score									
		Week 5	Week 6	Week 7	Week 8	Week 9	Week 10	Week 11	Week 12	Week 13	Week 14
Ridge-Regression Window Size	7							138.48	160.5	133.42	144.44
	6					99.1	101.58	136.68	157.24	100.72	127.34
	5			86.32	117.74	97.1	147.1	139.68	140.12	93.12	141.22
	4		109.94	127.2	94.82	84.9	96	128.78	173.72	136.12	111.66
	3	106.32	103.6115	104.22	103.12	86.8	149.7	145.56	112	126.72	139.74

Table 2 *Heatmap of our model roster performance across Weeks 5-14 of the 2020 NFL season vs. Ridge-Regression Training Window Length (3-7games)*

	Optimized Roster Actual Score									
	Week 5	Week 6	Week 7	Week 8	Week 9	Week 10	Week 11	Week 12	Week 13	Week 14
DraftKings Model	115.33	102.74	124.56	91.72	150.18	85.32	85.88	107.34	78.66	138.66

Table 3 *Heatmap of DraftKings model roster performance across Weeks 5-14 of the 2020 NFL season*

		Ridge-Regression Actual Score > DraftKings ? (1 = yes, 0 = no)									
		5	6	7	8	9	10	11	12	13	14
Ridge-Regression Window Size	7							1	1	1	1
	6					0	1	1	1	1	0
	5			0	1	0	1	1	1	1	1
	4		1	1	1	0	1	1	1	1	0
	3	0	1	0	1	0	1	1	1	1	1

Table 4 *Win/Lose visualization of effect of choosing Ridge-Regression model over DraftKings model across Weeks 5-14 of the 2020 NFL season (1 = Ridge-Regression Model beats DraftKings, 0 = otherwise)*

Ridge-Regression Window Length	Win % Over DraftKings
7 Games	100%
6 Games	67%
5 Games	75%
4 Games	78%
3 Games	70%

Table 5 *Win rate of utilizing Ridge-Regression model over DraftKings supplied projections across 2020 season.*

As seen in Tables 2-5, while there is variation in the Win % from utilizing a Ridge-Regression model of the projections supplied by DraftKings across the season, one is only able to apply a 7-game training window for later weeks as the model fitting demands the longer sets of input data. When focused on just the most recent four weeks of the season, there is not a significant difference in actual model performance as a function of training window size. The same cannot be said for model accuracy however.

Figure 4, below, shows that the distributions of model error for an optimized Ridge-Regression based roster were more closely distributed around zero than the DraftKings model (green) across weeks 11-14. The relationship holds true for the rest of the 2020 season, but as a 7-window Ridge-Regression model can only be evaluated weeks 11-14, those weeks are the standard for error comparison.

Figure 4 *Total Roster Error (Expected Points - Actual Points Across Weeks 11-14*

4. Further Improvements

While we believe our methods do an acceptable job of predicting the most optimal lineup, there are still opportunities for refinement. We can improve in 3 areas: feature engineering, model selection, and game selection.

As previously mentioned, a player position is strongly linked with point opportunities. Some points, such as fumbles and recoveries, can impact a player's overall Fantasy Points significantly. However, they are so infrequent that it is not worth predicting more accurately, while other points with a greater variance can influence a player's total more. For example, RBs and WRs have the most opportunity to score points in receiving yards. The total passing yards for both positions can vary greatly but are more extensive for WRs than RBs because there is typically only 1 or 2 RB per game while there are three or more WRs, which don't always have equal opportunities. Therefore, we may incorporate additional features such as historical play distribution to capture the bias for individual players.

Furthermore, other features may influence gameplay, such as who the opposing team is for a given week or player-player matchups. A team or player can generally be good offensively; however, if they are matched against a team/player known for having a tight defense, it would make sense to decrease our predictions. Likewise, the same reasoning can be applied to easier matchups.

Our predictions utilize a set number of previous game performances to predict the next performance. The size of this "window" of past games can influence the model's predictive ability. Varying the window size can help determine the optimal size to capture the player's recent performance. A smaller window can capture recent dynamics, such as a player receiving more/fewer opportunities or a team performing better but is more prone to variance. A larger window is less prone to variance; however, performances many weeks prior may not represent the player's current ability/opportunity.

In this particular study, we chose to use ridge regression to model our data. While it is usually sufficient for most applications, it's linear nature assumes features are independent of each other. As mentioned earlier, there may be some interaction between features that influence player performance, such as position and point opportunity or team matchups. A neural network is worth exploring as it can easily capture nonlinearities in the features.

Lastly, this study was limited to the default game type for daily fantasy football. Many different game variants allow for different salary caps, player selection, and Fantasy Point calculation. Due to the nature of DFS competitions, one can incorporate game theoretic methods that consider other rational agents' actions. Some DFS competitions are inherently riskier/more competitive than others (50/50 is generally lower risk than GPP) due to the entry fees, the pool of opponents, and the payout. Thus on average, they require a higher scoring lineup. In such cases, one would prefer a riskier high variance lineup that, in turn, has the potential to score big and win. The optimization procedure can be adjusted to incorporate some of these notions by optimizing over the variance, touchdown potential, or perhaps fantasy point floor/ceilings of the players' lineups.

Ultimately, our objective is to maximize our earnings from fantasy sports, so it stands to benefit from exploring different game types and determine which one we can reliably perform well on. Since we understand the shortcomings of our current methods, we can pick a game type that takes advantage of our model's strengths where we know we predict well and thus increase our likelihood of selecting the optimal lineup.

5. Conclusion

While there is an opportunity to continuously improve upon the derived model through additional feature section as well as develop future functionality for the application interface, the result of this paper demonstrates that roster generation utilizing Ridge-Regression based upon historical player data and integer programming optimization provides a more stable and statistically better outcome than the projections provided by DraftKings.

6. References

[1] DraftKings DFS https://www.draftkings.com/lobby#/featured

[2] FantasyPros: Fantasy Football Rankings, 2020 Projections https://www.fantasypros.com/

[3] GridironAI https://gridironai.com/football/

[4] Raimi, Pugliese. Predicting Optimal Daily Fantasy Basketball Rosters. Columbia University, 2015.

[5] Barry, Canova, Capiz. Beating DraftKings at Daily Fantasy Sports; A statistical approach to estimating the daily fantasy performance of individual players in the National Basketball Association. Stanford University, 2016.

[6] FootballDieHards https://www.footballdiehards.com/fantasyfootball/dailygames/Draftkings-Salary-data.cfm

[7] Statheads Football https://stathead.com/football/tgl_finder.cgi

[8] Bishop. Pattern Recognition and Machine Learning. New York, NY, 2006.