Predicting FanDuel Fantasy Football Production

Alexei Bastidas (alexeib@stanford.edu)
Ingerid Fosli (ifosli@stanford.edu)

Problem

We sought to predict weekly fantasy football production for six different position groups. Given a player and a week of the NFL season, we featurize the player and generate two outputs: with regression, we predict the actual points and with classification we predict the range (i.e. 10-15 points).
We built models for all six fantasy positions (QB, RB, WR, TE, PK, Def), tested three different learning algorithms(Random Forest, Gradient Boosted Trees, and Logistic/Linear Regression) for both regression and classification. Ultimately, we leveraged these predictions for our CS221 project to create optimal Line-ups to effectively bet on the FanDuel website.

Features

- **All:**
  - Difference between team and opponent rank and DVOA statistics for last X games and home/away boolean.
- **QB:**
  - Passing and rushing yds, passing and rushing TDs, pass completion percentage, sacks, INTs, QB rating, fumbles lost, pass attempts, FP scored, and salary for last 13 games.
- **WR:**
  - Receiving yds, receptions, receiving TDs, fumbles lost, FP scored, and salary for last 6 games.
- **RB:**
  - Rushing yards, rush attempts, receiving yards, receptions, receiving TDs, avg reception, avg rush, FP scored, and salary for last 3 games.
- **TE:**
  - Receiving yds, receptions, receiving TDs, fumbles lost, FP scored, and salary for last 3 games.
- **PK:**
  - FG made, FG attempted, PAT made, PAT attempted, points scored, FP scored, and salary for last 3 games
- **Def:**
  - FP scored and salary for last 6 games.

Automated Feature Selection:

For each position group, we leveraged SKLearn's SelectPercentile automated feature extractor to optimize the features for classification.
We selected the following top percentiles for each position:
QB - 90% ; WR - 85% ; RB - 85% ; TE - 85% ; PK - 85% ; Def - 75%

Data

We handbuilt our dataset by scraping: box-score statistics for receiving, rushing, kicking, and passing from NFL.com; offensive line, defensive line, team efficiency, team defense, and team offense rankings and DVOA analysis from FootballOutsiders.com; historical fantasy production data for FanDuel from RotoGuru.com; and we downloaded the weekly available player list directly from FanDuel.com.

Dataset Sizes		
Pos	**Train**	**Test**
QB	1949	421
WR	9520	1681
RB	8548	1268
TE	6052	905
PK	2386	354
Def	2368	354

For training, we used data from the 2011-2015 NFL seasons.
For testing, we used data from the ongoing 2016 NFL season.

Model: Random Forest

Random Forest utilizes a multitude of decision trees, where each tree is built top-down to ensure all leaves point to one class using the Gini impurity:

$$I_G(f) = \sum_{i=1}^{J} f_i(1-f_i) = \sum_{i=1}^{J}(f_i - f_i^2) = \sum_{i=1}^{J} f_i - \sum_{i=1}^{J} f_i^2 = 1 - \sum_{i=1}^{J} f_i^2 = \sum_{i \neq k} f_i f_k$$

Each tree gives different probabilities for each class; to get the decision function, we average the them and pick the class with the highest probability.

We opted to use 500+ trees of max height 20, with most position groups only using trees of height 10. This allows us to generalize better by reducing the likelihood of overfitting.

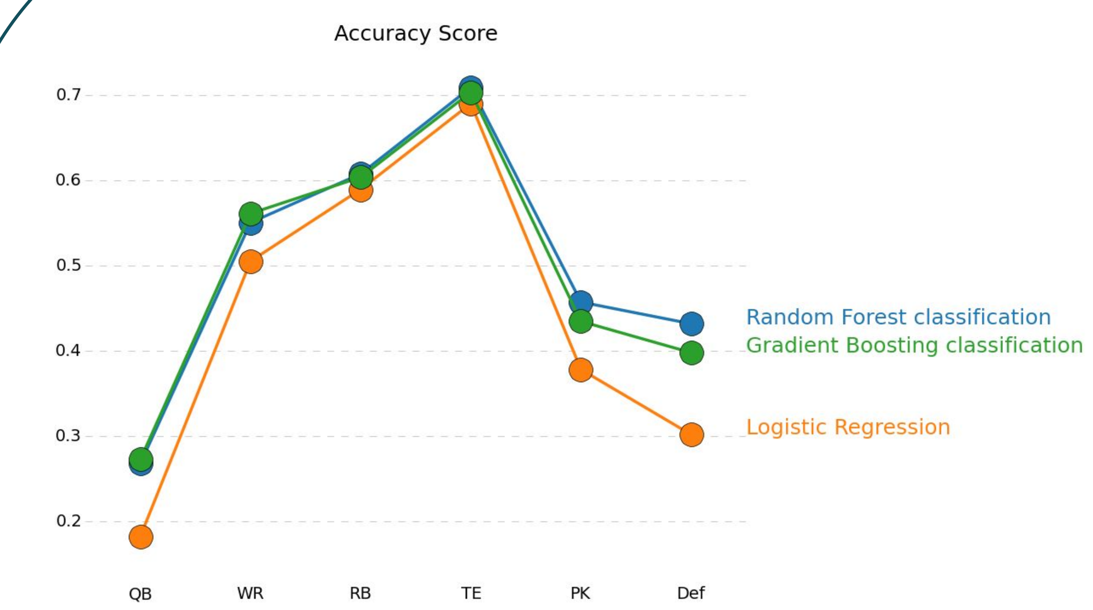
Discussion

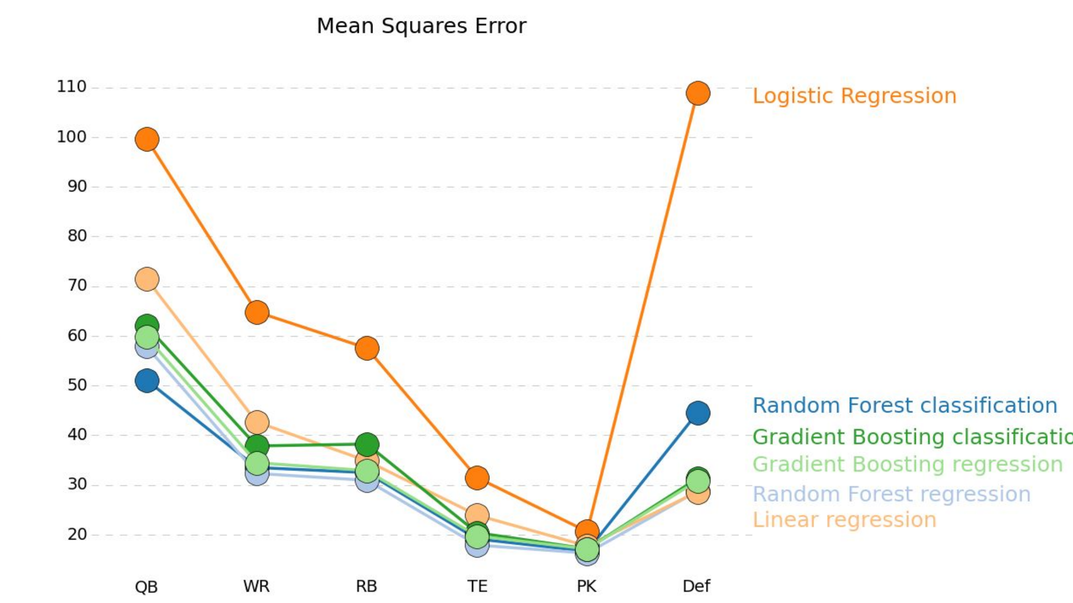
We define 7 potential output classes: (0 : 0-5 pts), (1: 5-10 pts), (2: 10-15 pts), (3: 15-20 pts) , (4: 20-25 pts), (5: 25-30 pts), (6: 30+ pts). Our focus is less on the class with the highest probability, but rather on the entire probability distribution for a given player.

Our goal is to generate predictions for betting. We found that accurate player predictions are extremely difficult -- the team based nature of Football implies a lot of inter-dependence between players on both sides of the ball, which, as evidenced by our accuracy and MSE scores, we did not capture.

However, while we could not reliably predict the exact number of points a player will score, by looking at the whole probability distribution across classes, and sorting by expectation, we found that the players with the highest expectation tend to be the players with the highest fantasy point production. So while we are not accurately predicting points, we do a surprisingly good job of selecting high performers on which to bet. With that being said, we have yet to turn a profit -- in fact, we've so far lost $60 :(

Results





To get the the mean squares error on the classification algs, we use an expectation generated from the classification buckets.

The table uses mean squares error for regression, and the accuracy score for classification.

Based on our evaluations, we note that Gradient Boosting and Random Forests perform very similarly, but we give the edge to Random Forest on account of a higher accuracy score and lower MSE on ⅘ position groups.

Regression - Mean Squared Error						
	Random Forest		**Gradient Boosting**		**Linear Regression**	
Pos	**Train**	**Test**	**Train**	**Test**	**Train**	**Test**
QB	47.121437	57.868713	50.386858	59.792819	48.305206	71.501647
WR	35.396886	32.303592	32.882062	34.459238	31.116383	42.638593
RB	26.223301	30.917668	28.022192	32.871084	25.405439	34.840013
TE	19.903092	17.883016	21.753863	19.709517	20.950363	23.902736
PK	19.006974	16.256255	20.551494	17.115988	19.229433	17.686407
Def	35.643216	28.738592	38.934565	30.805453	33.635457	28.604993

Classification - Accuracy						
	Random Forest		**Gradient Boosting**		**Logistic Regression**	
Pos	**Train**	**Test**	**Train**	**Test**	**Train**	**Test**
QB	0.312821	0.268409	0.328205	0.273159	0.264103	0.182898
WR	0.53729	0.550863	0.532356	0.561570	0.542973	0.505651
RB	0.611696	0.608044	0.612865	0.604101	0.592982	0.589905
TE	0.694467	0.709392	0.681055	0.703867	0.696945	0.690608
PK	0.41841	0.457627	0.393305	0.435028	0.393305	0.378531
Def	0.341772	0.432203	0.316456	0.398305	0.324435	0.302260

Future

There are some additional features we'd like to add, including snap-counts information and modeling likelihood of injuries.

More interestingly, we'd look at the interdependence between players. We could use covariance matrices to model them, and use convex optimization to generate the lineups.

References

1. Breiman, "Random Forests", Machine Learning, 45(1), 5-32, 2001.
2. J. Friedman, Greedy Function Approximation: A Gradient Boosting Machine, The Annals of Statistics, Vol. 29, No. 5, 2001.
3. J. Friedman, Stochastic Gradient Boosting, 1999
4. T. Hastie, R. Tibshirani and J. Friedman. Elements of Statistical Learning Ed. 2, Springer, 2009.
5. "Decision Tree Learning", Wikipedia, https://en.wikipedia.org/wiki/Decision_tree_learning