Machine Learning Engineer Nanodegree

Capstone Project

Andrew Lazzeri
January 12th, 2021

I. Definition

Project Overview

Sports analytics are ubiquitous in Major League Baseball, the National Basketball Association and more recently the National Football League. Sharp staffs and general managers are trying to extract every advantage and possible edge using teams of data crunchers to optimize personnel decisions and evaluate players. However, these analytical techniques are strangely lacking in the realm of fantasy sports despite the ever-increasing payouts and growth in the billion-dollar industry. Nowhere is metric-based fantasy analysis lacking more than the NFL and particularly dynasty fantasy football in which players keep the majority or all of their players year to year and try and identify promising rookies.

The ensuing analysis helps remedy this by applying frequently used machine learning algorithms to assist dynasty fantasy football owners. Specifically, the analysis estimates the likelihood rookie wide receivers become useful starters (for fantasy football) within their first five years entering the NFL. Currently, most dynasty player evaluation is based on qualitative inputs such as watching tape with minimal statistical analysis. Even when there are some statistical measures present, they are often fairly basic and do not have the same predictive power modern machine learning tools can provide. My hope is the ensuing analysis will serve as a useful complement to traditional player evaluation providing rewards to analytically minded owners and to help further the community's understanding of what metrics augur NFL WR success.

Problem Statement

Identifying productive young wide receivers in dynasty fantasy football is difficult given all the intricacies of modern NFL and NCAA passing offenses yet remains absolutely crucial in the context of dynasty fantasy football. Wide receivers have lengthy careers and with the increasing prevalence of points per reception scoring – which gives wide receivers higher relative value – selecting the right wide receivers in rookie draft can have lasting positive effects when constructing dynasty rosters.

Criteria for NFL wide receiver "success" abound but in this context the analysis will target dynasty wide receivers (WR) notching at least one half point per reception top 24 WR finish within their first five years in the NFL as a "hit". Not only do these WRs tend to provide multiple years of production, but they also tend to be the type of reliable producers week in week out which lead to more competitive teams in a game with highly volatile outcomes.

This study will use college production, NFL combine measurables, draft position as well as some grading and statistics from Pro Football Focus to construct a model which identifies productive fantasy starters. The analytical techniques in the beginning will be fairly basic but evolve into successively more complicated machine learning techniques and approaches to increase the usefulness of the model.

Metrics & Benchmark

The whole purpose of this analysis is to improve upon the existing techniques for player evaluation. There is no unanimous approach to player evaluation in NFL front offices and dynasty football players and sometimes participants don't even have standardized approaches to scouting. However, there are two data sources to get an idea of the consensus view on incoming rookies – the NFL draft and average draft position in dynasty football leagues available via the My Fantasy League website. A player can derive significant information from using these two sources and basic regression techniques but there are further ways to increase their predictive capabilities using additional data and more sophisticated approaches.

The model will focus on two evaluation metrics compared to the benchmark to assess success:

1) Accuracy: Measures the extent to which the model accurately designates success and failures scaled by the total number of predictions

2) Area Under the Curve: An estimate of the probability that a classifier will rank a randomly chosen positive instance higher than a randomly chosen negative instance. This metric speaks to how well the model distinguishes between likely successes and failures and is superior in scrutinizing data where class imbalances exist

The initial benchmark for the study is a simple logistic regression and scaled NFL and fantasy draft positioning. Selecting a simple model as a benchmark was intentional to illustrate how adding features and using more sophisticated modelling techniques can improve the model's predictive capabilities. It also helps represent the potential gains in information from supplementing fundamental player data with data science.

II. Analysis

Data Exploration

Datasets for the problem are difficult to compile and require various source material:

- College Football Reference Receiving Statistics
    - Contains receiving data for college WRs detailing college production
- Dynasty League Football (DLF) Dynasty Avg. Draft Position Data
    - Represents the average rank dynasty football participants assigned to each player. As an example, the player most consistently being drafted at the #1 overall spot is considered the player with the most promising career

- ESPN NFL Receiving Statistics
    - This is for the dependent. This will measure their success after selection and is the measure the dynasty fantasy football community most cares about
- Pro Football Focus Elite College Receiving Metrics
    - Advanced statistical metrics on performance which help supplement the college statistical data
- Pro Football Reference Combine Data
    - The NFL combine brings all incoming NFL players together to create standardized athletic tests measuring player speed, strength, agility and other measurables. In addition, this data also contains the player's placement in the NFL draft which provides information on what professional evaluators think of the player (and differs from fantasy draft placement)

All of this data is available for free via their respective websites, with the exception of the Pro Football Focus data that requires a subscription.

Compiling the data in of itself requires effort, however there will be some significant pre-processing and manipulation of the data itself before using machine learning techniques:

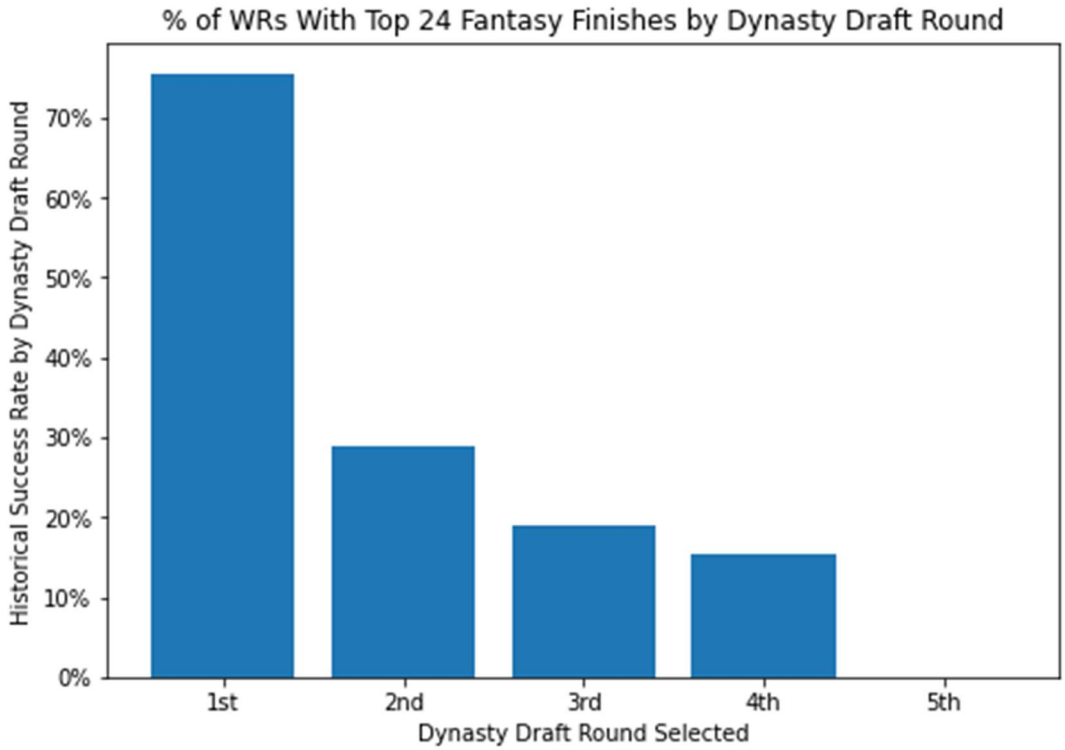
1) **Missing Data:** Some of the data points are missing for legitimate reasons. As an example, a player might not post combine measurables and Pro Football Focus doesn't have full grading coverage and statistics for all eligible players. Data imputation – filling in missing data using cues from existing data - can help alleviate some of the gaps in the data. As an example, if a player is missing 40 yard dash times at the combine but has stronger than normal yards after catch, avg yards per reception and a fast shittle time, it would suggest they also have a fast 40 yard dash time.

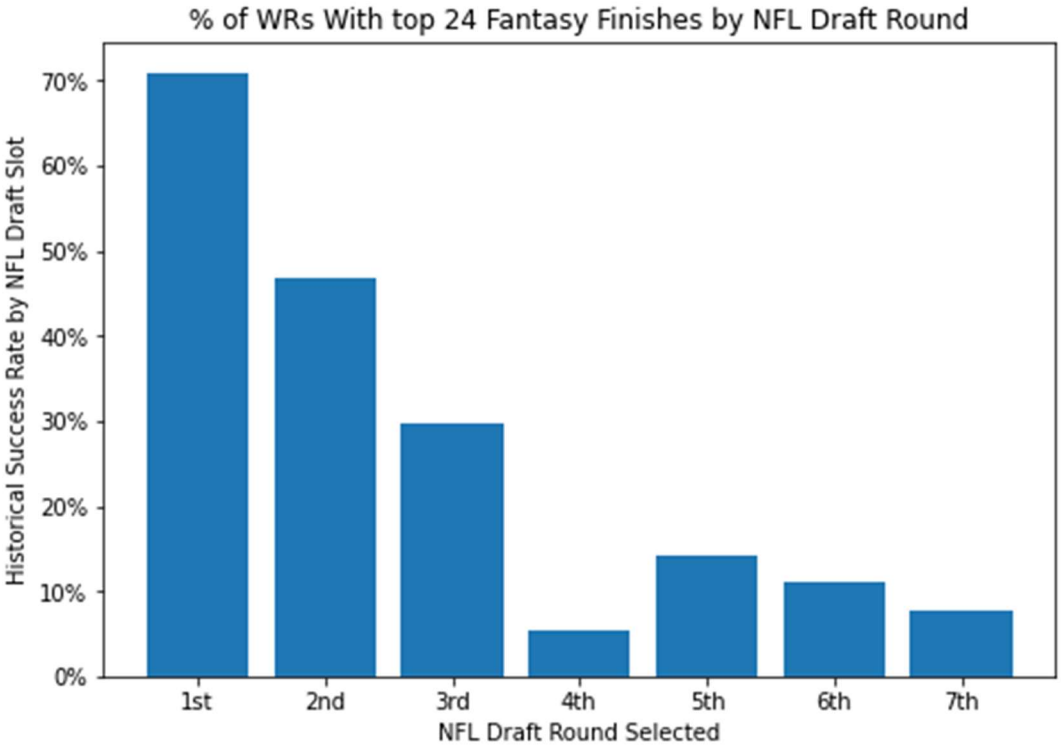
2) **Standardizing Data:** It's important to standardize data prior to using machine learning techniques, otherwise, difference the units of measurement might distort the results. Additionally, it's important to scale data appropriately so as to reduce outliers. Most of this is done by using features which are fairly normally distributed (such as population combine data) or focusing on player production per game

3) **Class Imbalances:** A class imbalance is when the target outcome is far more/ less common than the alternative (in this case the likelihood a top 24 fantasy WR). Class imbalances can lead to issues training classifier algorithms because at certain thresholds, the algorithm might find it most accurate to simply classify all observations in one category or another without discriminating. Two ways to circumvent this are to adjust the dataset to "balance" the two classes (upsampling/downsampling) or to use another criteria such as area under the curve when training models. This study will use both techniques with more detail in the data pre-processing and evaluation section.

Visualization

Player draft position can be fairly predictive itself and while everyone has their own views on players most dynasty players take some cues from the NFL and dynasty community consensus. But how accurate is it? Pretty accurate as it turns out. If you were to take a random cue on player selection it does appear that NFL and the dynasty fantasy football hive mind do a fairly good job of ordering prospects, particularly the top prospects. It may make sense to weight the consensus in the model albeit as one of several factors that drive player selection.





Algorithms and Techniques

This analysis will use a variety of algorithms to evaluate the data at hand starting from very basic linear models to more sophisticated bagging and boosting classifiers in an effort to increase the model's predictive capabilities. The rationales for each of the different algorithms is as follows:

Logistic Regression (Stats Model & Scikit Learn Logistic Regression)

Pros: This is the only model with coefficients which can be used for analysis as well as providing information on the statistical significance of the different variables. It is also the easiest algorithm for many people who are less familiar with machine learning to understand.
Cons: Despite these appealing characteristics for presentation, the underlying assumption of linearity and the presence of multi-collinearity result in lower accuracy than models which allow for more variance.

Decision Tree Classifier (Scikit Learning Decision Tree Classifier)

Pros: Decision tree classifiers allow for more non-linear interpretations of the data which are unavailable in a logistic regression model. Further, the decision tree nodes focus on how to partition the data in a manner with the highest node purity, giving insights into crucial rules to follow when evaluating WR success or failure.
Cons: Decision tree classifiers can over-fit the data so it's crucial to take steps to ensure that the model performs as well in testing as well as training. Further, there are several inputs which will require tuning to determine the optimal way to setup the classifier itself.

Decision Tree Bagging Classifier (Sklearn Bagging Classifier)

Pros: Bagging stands for bootstrap aggregation and is akin to create several different smaller decision trees and using their inputs to come to a consensus on the data. They are useful to avoid overfitting because they lower variance and provide a plurality of outputs.
Cons: The lower variance of using a bagging classifier might lead to a less accurate model than an approach with more variance. Additionally, the model might be difficult to interpret using visualization tools with the same ease as an individual decision tree classifier given its using so many model inputs.

Decision Tree Random Forest Classifier (Scikit Learn Random Forest Classifier)

Pros: Random forest classifiers are a subset of bagging classifiers which also take random samples of the data and construct models but also take an extra step where they randomize the input features. By doing so, they reduce variance even more than a bagging classifier due to the exclusion of specific features and can help avoid overfitting or having a few dominant features drive all the decisions for the model.
Cons: The lower variance of using a random forest classifier might lead to a less accurate model than an approach with more variance. The omission of specific features might result in a large portion of the input models missing key features which assist in segmenting the data. Additionally, the model might be difficult to interpret using visualization tools with the same ease as an individual decision tree classifier given its using so many model inputs.

Decision Tree Boosting Classifier (Gradient Boosting Classifier)
Pros: Gradient boosting classifiers are a group of machine learning algorithms which combine multiple decision tree classifiers together to create a stronger predictive model via iteratively incorporating new decision trees which reduce a loss function. The iterative boosting process typically results in very accurate models especially because gradient boosting trains on hard to classify residuals compared to adaptive boosting which will train on a newly sampled distribution.
Cons: Gradient boosting can be prone to overfitting as it can fixate on outliers too much. Additionally, the computation time and tuning can be quite intensive in nature and the network of decision trees can be difficult to interpret.

III. Methodology

Data Preprocessing

Preprocessing the data involves several steps to build features from the statistics, some minimal one hot encoding and finally imputation for missing values. The data starts with 5 csv files which are processed and merged together to create the underlying data (same data addressed in the data exploration section).

Adjustments to the data include the following:

1. Creating the dependent for WRs recording a top 24 finish in ESPN half point per reception scoring
    - Step 1: Take receptions, yards, touchdowns and lost fumbles and convert them into a season-long score according to standard half PPR rules
    - Step 2: Take the season long score (from the above) for all WRs in ESPN's scoring and then rank them grouped by season in descending fashion (highest score =1 and so on)

    - Step 3: Create a filtering mechanism to flag WRs with less than 5 years of eligible experience in the NFL
    - Step 4: Create a boolean column flagging if a WR notched a top 24 season for each season they were in the NFL
    - Step 5: Take the maximum of the boolean flagged column for each WR (AKA if they had any season where they were a top 24 WR, return a 1)
2. Preparing the college statistical data
    - Take the data on the college conference, replace references to conferences which no longer exist (such as replacing Pac 10 with Pac 12) and then create a dictionary mapping the most prestigious conference vs. lower levels of competition. Finally assign conference designations based on the player's most recent season prior to the draft (if they player switched conferences)
    - Create some efficiency metrics based on college statistics:
        - Yards per reception
        - Touchdown per reception
        - Yards per game
        - Touchdowns per game
        - Receptions per game
3. Prepare the combine data
    - Parse the data indicating draft position which is a string and extract the specific slot a player was selected in the NFL draft
    - Take players who were undrafted and assign them the 224^{th} draft slot instead of a null value (this treats undrafted players as if they were the last pick in the NFL draft)
    - All other combine metrics are already floats or integers and do not require further conversion
4. Prepare the Pro Football Focus data
    - Limit the PFF data to player seasons with at least 25 targets in a season to avoid providing stats on players with severely limited opportunities
    - Aggregate all of the college player's statistics for eligible seasons of 25 targets or more to create a dataset which reflects the players total college production
    - Create a series of career efficiency metrics
        - Yards per target
        - Catches per target
        - First downs per target
        - Turnovers (fumbles and credited receptions) per catch
        - Missed tackles per reception
        - Yards after catch per reception
        - Drops per target
        - Longest catch
    - Create average grades for players over their full college career with a group by mean of all PFF season grades
5. Merge all of the inputs into a final dataframe for the final cleanup, scaling and imputation
    - Filter the data to include any WR who is already a success (even if they were drafted less than 5 years ago). This is because we already know their classification in this case
    - Remove players who were drafted in the last five years who have yet to post a top 24 WR season (because they could be a future success with development)
    - Count the total number of nulls in the data and remove observations with a critical mass of nulls which simply don't have enough data
    - Use scikit learn's min-max scalar to scale to normalize the data for comparison

    - Create a function to help with inputs to the k nearest neighbors' imputation. This function will produce a loss function (RMSE) when imputing the dataset an using a random forest classifier on the predicted information
    - The loss function was minimized with 26 clusters in k nearest neighbor so that was what was used for imputation. This is enough clusters to start filling in data but not too much to overfit the imputation of the dataframe
6. The last step is dealing with the class imbalances present in the data. At the beginning roughly 1/3 of the WRs drafted ended up with top 24 WR seasons thus the data will require resampling. In response the study utilizes Sklearn's utils resample feature to resample the data for a 50/50 split between successes and failures (there were 110 failures to match).

Implementation

This study is designed to model several machine learning techniques and then settle into the model which has the highest area under the curve on the testing set as the final solution to the problem. The study will cover the following test scenarios to illustrate progressively more effective models for the problem:

Test 1: Logistic Regression with NFL draft & fantasy football draft position,
- Purpose: This model is designed to serve as the initial benchmark for success. While this may seem naive, recall NFL and dynasty draft position represent the consensus views of many practitioners with different modalities for player selection.

Test 2: Logistic Regression with all possible feature data
- Purpose: This model is intended to illustrate the predictiveness using all available features in a logistic regression. While this will improve the AUC because of the increase in features, it will likely be less predictive than other more flexible approaches.

Test 3: Decision Tree Classifier with default inputs
- Purpose: A decision tree classifier serves three purposes 1) to break the assumption of linearity 2) to establish a baseline for the predictiveness of a naïve decision tree classifier prior to tuning and 3) to provide illustrations for decision tree nodes.

Test 4: Bagging Classifier with optimized inputs
- Purpose: Bagging models are for exploring if several optimized decision tree classifiers provide a more effective solution than just one tree (and alternatives).

Test 5: Random Forest Classifier with optimized inputs
- Purpose: The random forest model is in the analysis to explore if an optimized random forest classifier provides a more effective solution to the final problem.

Test 6: Gradient Boosting Classifier with optimized inputs
- Purpose: Finally, the gradient boosting classifier is for investigating if optimized gradient boosting classifier provides a more effective solution to the final problem.

Refinement

The initial accuracy and area under the curve results were fairly strong with the default settings but can be improved with further tuning. The tuning / refinement process will take place in a few steps. To do so, I created a function which can iteratively feed the models different inputs, run the model and produce a new AUC for each to pinpoint areas which improve accuracy.

Base Decision Tree Classifier: Tuning some of the initial decision tree classifier's major inputs such as max depth, minimum sample leafs and max features indicated changing the max depth of the tree to 6 and

max features to 15 were the only parameters adding value. This resulted in an improvement to the first model from an AUC of 84.2% to an AUC of 87.6%.

Bagging Classifier: Tuning some of the bagging classifier's major inputs such as the max features and number of estimators indicated manipulating the max features to 10 helped tune the model. This resulted in an improvement to the first model from an AUC of 84.2% to an AUC of 94.4%.

Random Forest Classifier: Tuning some of the random forest classifier's major inputs such as the number of estimators and max depth indicated potential improvements to the random forest classifier. Altering the max depth to resulted in an improvement to the first model from an AUC of 86.4% to an AUC of 92.2%.

Gradient Boosting Classifier: Tuning some of the gradient boosting classifier's major inputs such as the learning rate, number of estimators and max depth indicated potential improvements to the gradient boosting classifier. Altering the max depth to 2, the number of estimators to 500 and a learning rate of .55 resulted in an improvement to the first model from an AUC of 86.4% to an AUC of 88.7%.

The bagging classifier had the highest AUC of all the models. This appears to be the leading candidate for the final model, however, further validation and sensitivity tests will come into consideration as well as it's important for the model to generalize well to the dataset.

IV. Results

Model Evaluation and Validation

Model Validation and Sensitivity Analysis

It's important the final model generalize well across multiple samples and thus the models require some more validation. Additionally, the NFL landscape is in flux and observations from a longer time ago might not generalize as well into the future if there are fundamental changes in the way colleges and the NFL utilize WRs. Therefore, validation will involve testing the AUC over a series of fresh randomly generated samples as well as an exercise which trains the models using data pre-2014 and post-2014 to see if the selected model is as effective now as it was in the past.

Thus far the test has centered on one random seed to provide the ability to replicate results. However, it's important to know if the model generalizes well to other samples. One way to test this is to generate a variety of samples with random seeds and then test the AUC for a series of trained models given different sample inputs. Since the Gradient Boosting, Random Forest and Bagging Classifiers showed the most promise, I will train and test each one using 200 random seeds and evaluate the area under the curve for out of sample data.

The Gradient Boosting Classifier was the most successful model when using AUC as a benchmark over the 200 random samples. It achieved a mean AUC of 93.1% compared to an AUC of 92.5% and 91.5% for the Random Forest and Bagging Classifier respectively. Further, while it was less successful than the Bagging Classifier post 2014, it's AUC (92.9%) was similar to the other classifiers (93.1% and 92.9%) during that period.

Justification

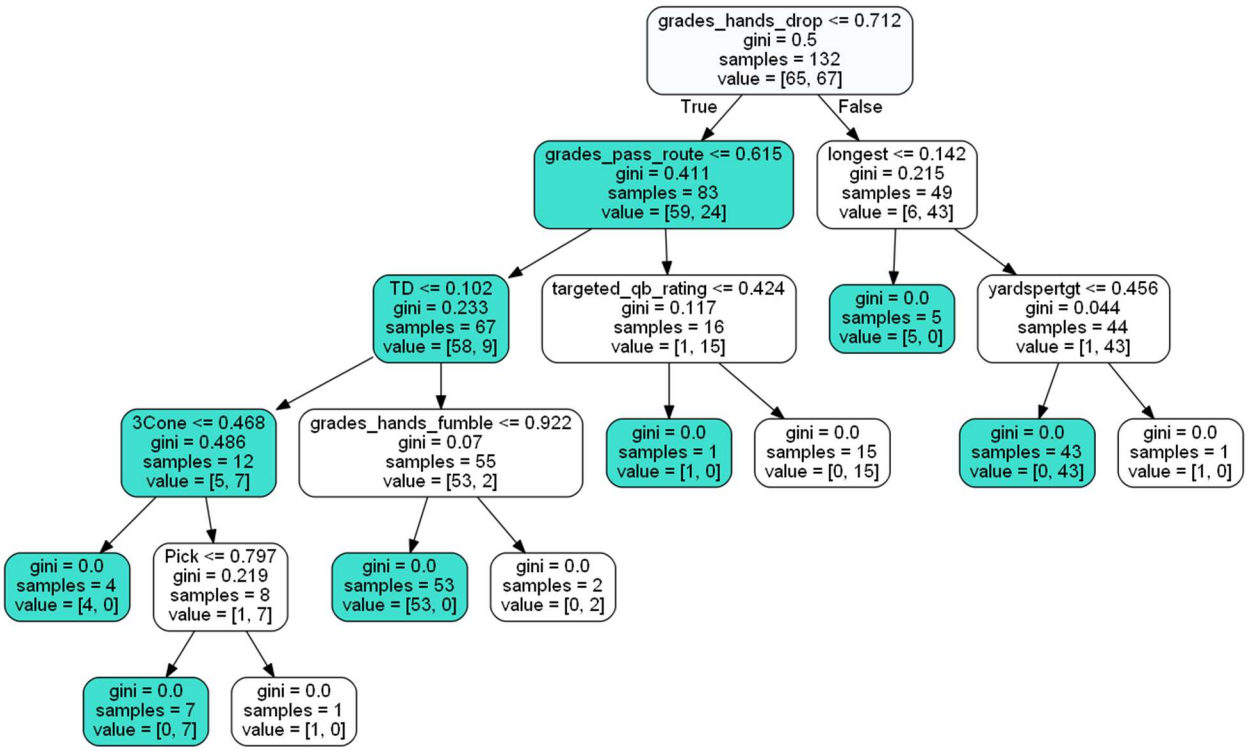
The final solution to the problem is to use the trained Gradient Boosting Classifier to model success in the future. It was the most successful model looking at a variety of samples and will likely benefit time progresses and there are more samples for the gradient boosting to work with.

The Gradient Boosting Classifier had a demonstrably larger AUC relative to a model which proxies the NFL and dynasty community consensus (the logistic regression model with an AUC of 68.4%) which is one of two justifications for using it as a solution to the problem. The other justification is that it can be used as a supplement to existing scouting processes. An "ensemble" approach of using models such as this and fundamental scouting might lead to even more accurate player evaluation.

V. Conclusion

Free-Form Visualization

One of the appealing aspects about decision trees generally are the simple rules they create for application in the real world – which are even useful to those who are uninformed about machine learning. While we cannot visualize all the trees from a Gradient Boosting Classifier, it is possible to show the results for the original tuned decision tree classifier from earlier as an indication of what the Gradient Boosting Classifier might look like. We can generate a visual using graphviz.



The decision tree above helps to visualize the major factors driving WR success. The turquoise bubbles represent nodes where the prior condition was true and white nodes represent when the prior segmentation was false. The brackets will have two slots, the left indicates the number of WR failures whereas the right indicates receivers which ultimately "hit" and produce at least one top 24 season. Nodes are formed by how much they reduce "impurity" or in more simplistic terms what splits the data in ways which provide the biggest information gain.

First Level – PFF Hands Drop Grade: The first way to sort WRs is by the PFF hands drop grade which measures the extent to which WRs drop catchable passes. In this case, the min-maxed scaled bar which divides WRs is at <= .712 which is similar in interpretation to a percentile. The PFF hands drop grade is on a scale from 1 to 100 with 100 being receivers who do the best in securing catchable passes thus a higher number equates to better hands. Any WR who has hands >.712 relative to all the other eligible WRs (good hands) is more likely to have success and vice versa. This makes sense on an intuitive level given that catching the football is an integral part of playing WR in the NFL.

Second Level – PFF Pass Routes Grade & Longest Receiving Gain: The next level has two dividers based on how WRs were routed along the first criteria.

> **PFF Hands Drop Grade <= .712 -> PFF Pass Route Grade**
> WRs with sub-optimal hands are then evaluated based on their PFF pass route grades which measures a WRs' ability to run routes during college. Again, this separates receivers based on <= 61.5% scaled grade and again the higher this number the higher PFF grades a WR on their route running ability. Naturally, this is important in WR selection given they have three primary goals – to get open, catch the football and then generate yardage in the open field after the catch. It's fairly intuitive receivers with poor hands and route running ability are unlikely to do well.
>
> **PFF Hands Drop Grade > .712 -> Longest Reception During the Season**
> After considering a WR's hands, the decision tree next evaluates the player's longest receiving play during the season. A player's longest receiving play during the season is indicative of their ability to threaten deep and/or generate yards after the catch from a production perspective. Additionally, it might also infer a heightened level of speed, acceleration, vision and agility in the open field. Players with good hands who have <= .142 scaled longest play might simply be serviceable possession/slot receiving options in college that are too one dimensional for fantasy production in the NFL. Increasingly, NFL WRs are expected to challenge all levels of the field and generate breakaway plays. Without this their future as valuable fantasy assets is shaky.

Third Level – Touchdowns Per Reception, QB Rating When Targeted, Yards Per Target
My qualitative analysis will end at the third level. The third level has three decision nodes and the end of a node for WRs who have good hands but are severely limited generating breakaway plays.

PFF Hands Drop Grade<= .712 -> PFF Pass Route Grade<=.615 -> TDs / Reception
By now the population of WRs in the far left node have poor hands and run poor routes. This node evaluates the number of TDs per reception and splits WRs into receivers when the scaled TD rate <= .102 – essentially determining if a WR was in the bottom 10% of scoring touchdowns per opportunity. At this point, receivers don't execute their primary responsibilities well and don't have a knack for scoring either. While there are some successes further down the node with some additional caveats, this is very infertile territory for success.

PFF Hands Drop Grade<= .712 -> PFF Pass Route Grade>.615 -> QB Rating When Targeted
Receivers in this node have sub-optimal hands but do have the redeeming value that they are good route runners. However, one failure remains which the model is trying to weed out. The decision tree uses a PFF metric measuring the QB rating when that WR is targeted (a measure of the passer's efficiency on the QB/WR connection). It's difficult to say the true reason why a QB rating would serve as the deciding factor but one theory is that it may speak to a lack of connection or miscommunication with the QB and a lack of preparation or understanding passing concepts (or noise).

PFF Hands Drop Grade > .712 -> Longest Reception > .142 -> Yards Per Target
Receivers in this node have good hands, have some breakaway ability but there is still one more measure needed to weed out the lone failure on the node. Yards per target here appears to be another supplemental measure of playmaking ability / the extent to which a receiver is a deep threat versus some sort of slot/possession WR. It's possible the lone WR who passed the prior node had a fluky long play during the season which wasn't indicative of their true game breaking talent and the decision tree is cleaning this up.

Reflection

The original problem was the need to supplement existing fundamental scouting of dynasty WRs with a more empirically driven way to analyze them. This had several interesting and challenging components such as organizing and cleaning all the data, determining all of the machine learning algorithms to use, validating the proposed models and then visualizing and interpreting the data from the tuned tree model. Cleaning and prepping the data into a consolidated dataframe was probably the most difficult part because it involved so much manipulation, merging, scaling and imputation with each step requiring analysis on the pros and cons of different approaches. The model's predictive capability (AUC in this case) ended up exceeding my expectations and the relevant features were also surprising. I would have thought draft position would have played a more pivotal role in the final model but perhaps there is some correlation with the PFF data which took center stage. Now the model can be deployed to future classes of WRs and an advantage in dynasty fantasy football.

Improvement

In my perspective, there are three areas for potential improvements; 1) add some features in the data which speak to the landing spot for WRs and schematic fits 2) test if other uninvestigated algorithms are capable of increasing the AUC using the prior data 3) spend more time considering different methods for imputation and backfilling missing data to increase the number of observations.