NFL Data Exploration

Broden Wanner, David Ma, Albert Liu

December 17th, 2020

The National Football League (NFL) is the largest professional sports league in the United States, the wealthiest sports league in the world, and the sports league with the most valuable teams. Yet even despite all of the money moving through the league, data analysis has only recently been incorporated into improving the play of its various football teams. Unlike basketball or baseball, the NFL is significantly behind the big data curve when it comes to applying analytics to the game of football. As a result, the NFL believes that it needs to catch up - a sentiment that we share. We personally believe that applying data analytics techniques to the goldmine of data collected by the NFL can produce meaningful results. To that end, in this project, we seek to analyze NFL passing play data to see what kinds of insights we can glean. Specifically, we attempt to classify the outcome of passing plays, as well as do a bit of analysis on NFL players themselves. By and large, we tried out a number of classifying techniques as well as a bit of clustering, applying rather standard data analysis approaches to our chosen dataset. We find that we are able to classify passing plays to a very minor degree of success, but we encounter greater success classifying NFL players. The GitHub repository for the project can be found at https://github.com/broden-wanner/nfl-data-insights.

Motivations and Objectives

As mentioned prior, the NFL is a gargantuan sports league, where winning and losing can mean a difference of tens, if not hundreds of millions of dollars. And yet, it was only in 2014 that Doug Pederson became the first head coach to publicly mention data analytics in his coaching. While the Eagles are not exactly doing well at the moment, Doug Pederson did manage to win a Super Bowl—which maybe says something about the efficacy of data analytics. In the years since 2014, an increasing number of NFL teams have scrambled to apply data analytics to improve their own performance, building their own analytics departments and hiring data analysts.

However, data analytics in the NFL is still very much in its infancy. Sports and science have not always intermingled intimately, yet the amount of data that can be tracked about football games nowadays is immense. To that end, we believe that applying data analytics to football can reveal significant opportunities to streamline gameplay. Whether to go for it on 4th down, what formations are the most effective against different teams, which routes work well for which cornerbacks, etc. are all insights that might be revealed by digging into the numbers. In a sport as complex as American football, with so many variables to track, modern machine learning principles can discover information not immediately apparent to the naked eye.

In this case, we attempted to discover two distinct insights into football data, each of which will be specifically expounded on in their respective sections. The first is classifying the results of passing plays. The second is deriving insights into NFL players themselves. Overall, we applied various data mining techniques to the raw dataset in the hopes that we would be able to discover interesting results and relationships.

Data Analysis

Dataset Overview

The first dataset used for this project was the play-level dataset from the 2018 NFL season [3]. It contains data on 19,239 plays with 27 attributes relating to various metrics and information about each pass play of the season. This includes attributes such as the game ID, play ID, pass result, pre-snap home and visitor score, offensive personnel, defensive personnel, and penalty information. The attributes themselves are of types continuous ratio, discrete ratio, and discrete nominal. In this project, this dataset will primarily be used for the task of analyzing pass play results and predicting them based on the other attributes in the dataset.

The second dataset used for this project was a NFL Player and Combine dataset [2]. This dataset includes information on 4,419 NFL players with 62 attributes, including attributes such as their team, height, weight, 40-yard dash time, and vertical. The attributes are of types continuous ratio, discrete ratio, and discrete nominal. This dataset will primarily be used for the task of clustering NFL players by position using unsupervised methods.

Dataset Exploration

The pass play dataset contains comprehensive information on every pass play during the 2018 NFL season. Many of the attributes include irrelevant information (such as the game ID and penalty codes), and other attributes are taken after the play has happened (and are thus not useful for the task of prediction). Moreover, a total of 639 entries in the dataset contain null values (disregarding known irrelevant attributes like penalty codes), mostly in the pre-snap scoring attributes and the type of dropback, which could be important predictors of the pass play result. Despite these shortcomings, the data still provides some interesting insights into the pass plays of the NFL.

First, since we are evaluating the ability to predict pass play results, let's examine distribution of the different classes of play results. There are 4 different types: "C" (denoting that a pass was completed), "I" (denoting that a pass fell incomplete), "IN" (denoting that a pass was intercepted), and "S" (denoting that the quarterback was sacked). There was also a fifth class, "R", in 4 entries of the dataset, but this is likely a typo as there are only 4 types of pass play results. The dataset distribution is skewed towards the complete pass plays, which make up approximately 11,000 entries in the 19,000-entry dataset. Further efforts of classification will take into account this class imbalance problem.

There are 7 attributes in the dataset that are taken after the play has completed. This includes the penalty codes, yards gained on the play, and the EPA (expected points added). Notably, the EPA is a metric that estimates the average of every next scoring outcome given the play's down, distance, yardline, and time remaining. For the task of prediction, these attributes will not be used since they are an effect of the play and not a predictor. However, they can be useful for evaluating our own classifier.

To analyze the relationship between the attributes on the data, we apply some statistical techniques. Figure 1 shows a correlation heatmap between the attributes in the pass play dataset. The only strongly correlated attributes are the pass result, play result, and EPA. This is logical because these attributes are metrics taken after the play has happened, and we would therefore expect these to be strongly correlated. The remaining attributes do not show a strong correlation with each other and especially with the target variable of pass result. Table 1 shows the p-values of a chi-squared independence test of the pass result against a subset of the discrete attributes on the dataset. We perform this test because it looks for more than linear relationships (which is what the Pearson correlation coefficient measures). If the p-values falls below 0.05, this indicates that there is some relationship between the pass result and the discrete attribute. The table shows that some of the attributes such as "playType" and "offenseFormation" are related to the pass result, while others such as "under2mins" and "possessionTeamAhead" seem to not have a strong relationship with pass result.

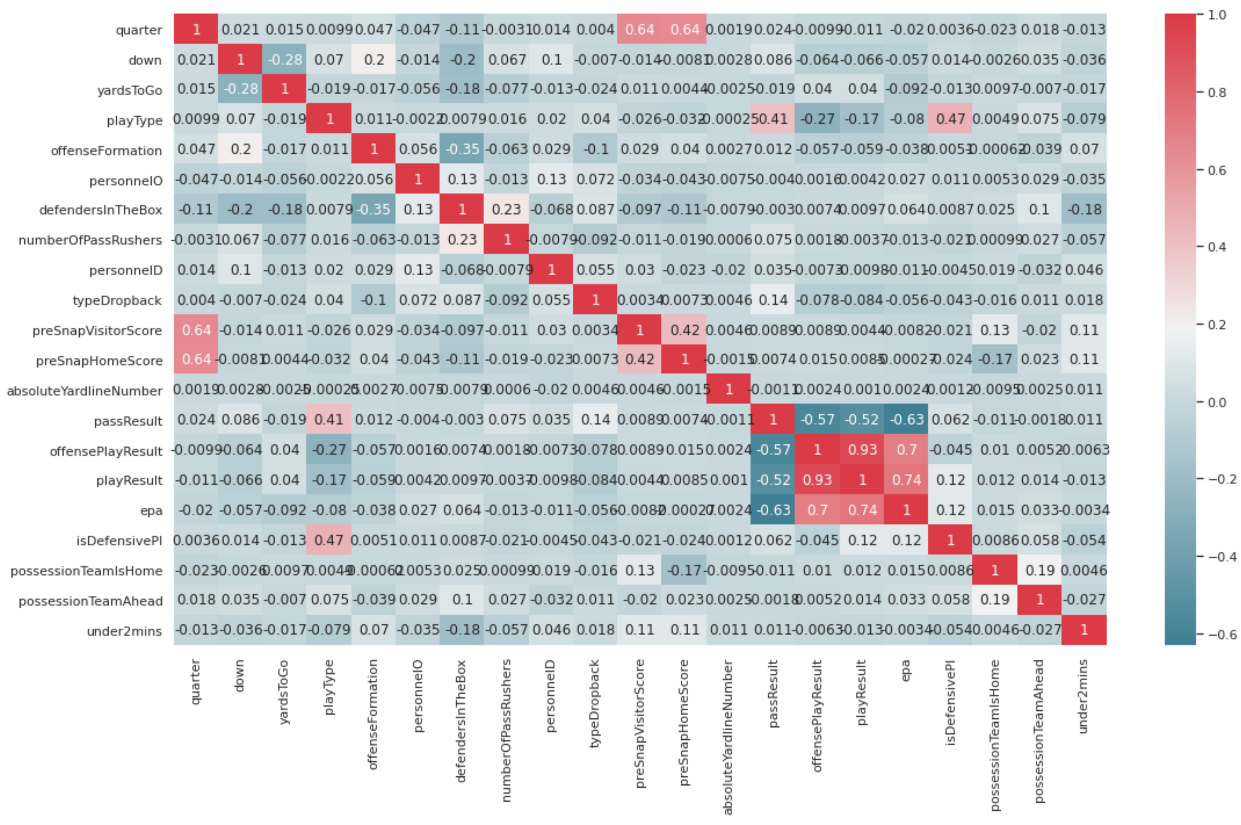


Figure 1: Correlation heatmap for the NFL pass play dataset

Attribute	p-values	Below 0.05?
playType	0.0000	Yes
offenseFormation	0.0024	Yes
personnelO	0.1750	No
personellD	0.0023	Yes
typeDropback	0.000	Yes
under2mins	0.0538	No
possessionTeamAhead	0.6224	No
possessionTeamIsHome	0.4554	No

Table 1: Chi-Squared p-values for Pass Play Dataset attributes vs. pass result

Attribute	p-value	Below 0.05?
numberOfPassRushers	0.0000	Yes
yardsToGo	0.0000	Yes
absoluteYardLineNumber	0.9812	No
preSnapVisitorScore	0.5638	No
preSnapHomeScore	0.0562	No
under2mins	0.0538	No
possessionTeamAhead	0.6224	No
possessionTeamIsHome	0.4554	No

Table 2: One-way Anova p-values for Pass Play Dataset attributes for pass result populations

Table 2 shows the p-values of one-way anova tests for attributes of the complete and incomplete pass result populations. If the p-value is greater than 0.05, then the populations are said to have different means for those attributes. The results show that some attributes should have different population means for complete and incomplete pass plays. Attributes we would expect to be correlated with pass result, such as the number of pass rushers and yards to go have different population means. However, other attributes such as the absolute yard line and pre-snap scores do not seem to have different means for complete and incomplete. This should give us some intuition on what attributes will be best for distinguishing between different pass results.

The NFL Player and Combine dataset is a mix of two datasets we made to aid in the clustering and classification of players. The first dataset contains player information from the 2018 NFL season. However, we found the attributes on this dataset lacking for the task at hand. We then combined this with the NFL Combine dataset, which has numerous attributes of all NFL players from 2002 to 2018. As a bit of background, the NFL combine is a scouting showcase where prospective NFL players perform various mental and physical tasks for coaches, general managers, and scouts. In this case, we're focused solely on those physical tasks; the specific tasks we used were the forty yard dash, vertical jump, bench, broad jump, three-cone, and shuttle run. Every attribute was continuous, so there wasn't too much preprocessing necessary; just standardization. Figure 2 shows a correlation heatmap between the attributes. As can be seen, many of the attributes are strongly correlated with each other.

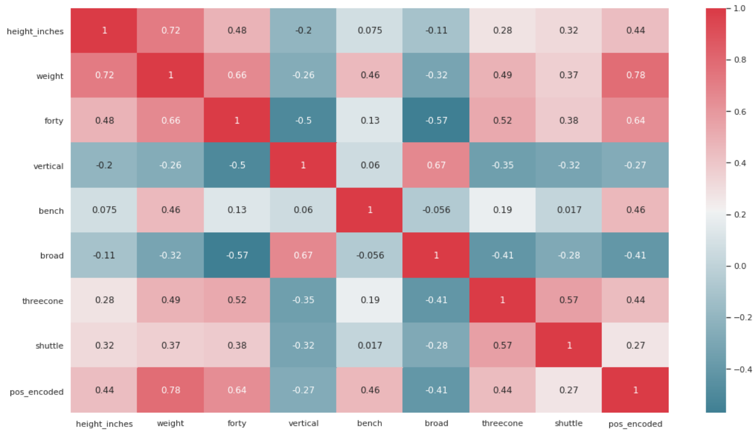


Figure 2: Correlation heatmap for NFL Combine Dataset with selected attributes

Pass Play Insights

Motivation

The primary goals for using data analytics alongside NFL data is to try to derive insights that may lead to winning more games. As such, it makes sense to target attributes that either directly contribute to winning the game, or attributes that are controlled by the teams, to have a better idea what your upcoming opponents may attempt to do in certain situations, such as if they are also trying to maximize their winning opportunities through analytics as well.

Naturally, as this is a pass play dataset, it makes sense to first try and target the "passResult" attribute, which determines whether or not the pass is completed, incomplete, intercepted, or the QB is sacked on a given pass play. If a team is able to maximize the number of completions, as it is the only outcome that results in positive yardage gained (outside of penalties) for the team on offense, they will also effectively maximize their chances of putting points on the board, and therefore, winning. If we are able to find a classifier that can accurately predict the outcome of a pass attempt, NFL teams can use the same attributes the classifier deems relevant, in practice, to improve the outcomes in real games. If it is an easy to visualize classifier, such as decision trees, the specific attributes chosen for splits can provide this information. If it is a classifier that does not lend itself well to this approach, you can also simply use the trained classifier on a series data points representing hypothetical situations, and see how the classifier predicts the team performing on the given play.

A secondary target of using data analytics would be in trying to use the information to predict opponents' decisions in given situations, assuming data analytics backed decision-making sees a proliferation across the league (which seems to be the case). As a result, it makes sense to target some of the attributes that are directly controlled by the teams coordinators, such as the offense personnel and offensive formations.

Methods

Preprocessing

Before running classifiers on the dataset, we first needed to clean and preprocess it to make it better suited for our needs. To start, we completed missing information in the dataset and dropped irrelevant attributes. We dropped any rows that had null values in target attributes, and then used data imputation to fill out the few remaining rows that still had null values. We filled in nominal attributes with the mode and the ratio attributes with the median.

Then, we dropped attributes as follows: the "gameId" and "playId" attributes, since these are arbitrary identifiers for the given play and game; the "playDescription" column is unnecessary, since it is simply description of the details of the play in plain English, and its information is already captured by the other attributes in the dataset; "possessionTeam" was also dropped since we do not care about the specific team involved in the play; "yardlineNumber", and "yardlineSide" are not irrelevant, but redundant attributes, since their information is just a differently formatted version of other attributes in the dataset; the attributes dealing with penalties, like "penaltyJerseyNumbers" and "penaltyCodes", are dropped since when penalties occur, the "play" technically does not happen, and no pass is considered to have been attempted. We also dropped the causal attributes that are metrics which are measured after the play has taken place. These include "epa", "playResult", "offensePlayResult", "playType", and "isDefensivePI". At this point, we end up with 19,233 entries in the dataset, alongside 16 potentially relevant attributes.

Beyond this baseline preprocessing to ensure compatibility with the standard sets of classifiers, we also tried feature engineering on certain attributes we supposed may yield differing results. Precisely, we tried to elucidate some commonly held beliefs about player performance in the NFL, and sports in general. Three features that would seem useful to us are (1) whether or not the team with possession of the ball is the home team (called "possessionTeamIsHome"), (2) whether or not the possession team is ahead or behind (called "possessionTeamAhead"), and (3) whether the game clock is under two minutes (called "under2mins"). Each of these are symmetric binary attributes and were created using a combination of features on the plays dataset and on the games dataset.

We then converted the remaining attributes into types that can be used for classification. First, we one-hot encoded all nominal attributes, which turns them each into a series of asymmetric binary attributes for each possible value of the attribute. We did this instead of simply encoding the attributes as integers because nominal attributes have no sense of order. Encoding them as integers would introduce an ordering that is not necessarily present in the data. The following attributes were one-hot encoded: "playType", "offenseFormation", "personnelO", "personnelD", and "typeDropback". Our target attribute "passResult" was encoded as an integer because the classifiers in our methods section do this to identify different classes instead of using one-hot encoded vectors. After this preprocessing step, we had 139 attributes.

To ensure each continuous and discrete attribute has similar ranges, we scale them using the standard scaler, which centers each attribute by subtracting the mean and scales it to unit variance. We chose this method of scaling because there are virtually no outliers in the data, which would affect the standard scaling if there was a large amount of outliers.

After this preprocessing, we split the dataset in a stratified fashion into a train and test dataset, with 90% of the data in the train dataset and 10% in the testing dataset. The stratified splitting of the dataset will ensure that the same proportion of each of our classes is in the train and test datasets. Using the train dataset, we then performed recursive feature selection using scikit-learn's recursive feature elimination class. Given an external estimator (we chose Linear SVM because of its speed), the recursive feature eliminator will select features by recursively considering smaller and smaller sets of features and choosing the best ones at each step by training the estimator on each set of features and comparing their performance. By using the training dataset to select the features, we will not be affecting the validation score on the test dataset. After some initial testing, we found that selecting 100 attributes gives the best performance on the data. The attributes that were removed were primarily the asymmetric binary attributes created from the one-hot encoding of the personnel attributes. Additionally, the attributes in the data exploration section that had high p-values for the chi-squared and one-way anova tests were removed. So at the end of the preprocessing step, we have 50 attributes to use for classification.

Classification

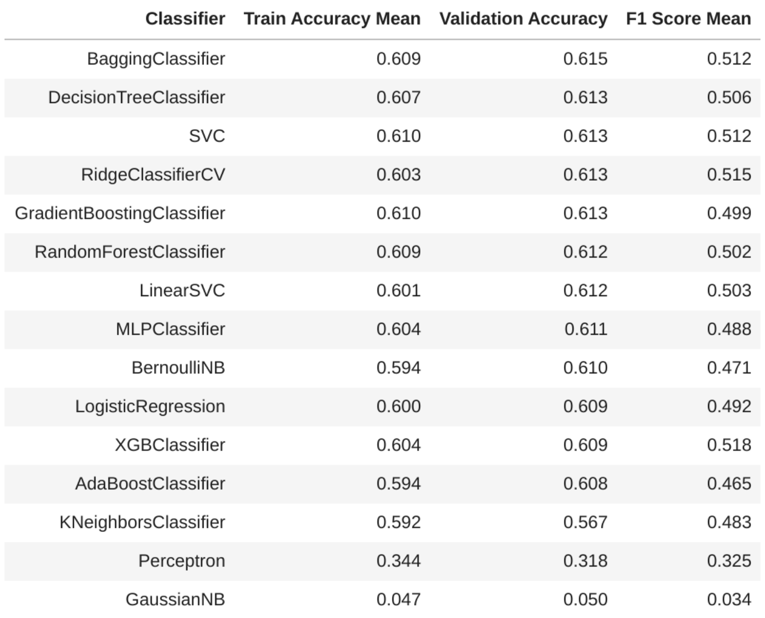
For the task of classifying the pass result of a play based on the other attributes of the play, we use classifiers from the scikit-learn Python Library [4] and one classifier from the XGB Python libary [1]. We tested the following classifiers' ability to predict the pass result given certain attributes of the play: Linear SVM, SVM, KNN, Logistic Regression, Decision Tree, Random Forest, Ridge Classifier, AdaBoost, Bernoulli

Naïve Bayes, Gaussian Naïve Bayes, Gradient Boosting, XGB, Bagging, and Multilayer Perceptron (MLP) classifiers. These were run with parameter values tuned by GridSearchCV. These can be compared against baseline classifiers, including Random Classifier (uniformly random predictions), Stratified Random (generates predictions by respecting the training set's class distribution), and Most Frequent Classifier (always predicts the most frequent class).

The means of the train accuracy, test accuracy, test precision, test recall, and test F1 scores were recorded for each classifier with their best parameters determined by GridSearchCV with 3 folds. Because this is a multi-class classification problem, the overall precision, recall, and F1 scores for each classifier are calculated by taking a weighted average of each metric over all classes. Additionally, the validation accuracy on the test dataset was recorded to get a better sense of how well the classifiers would generalize to new data and check for overfitting.

Results

Overall, when targeting the outcome of the pass ('passResult' attribute), we managed only marginal improvements over the best of the baseline classifiers. The random classifier, stratified random, and most frequent classifiers had accuracies of 25, 44%, and 58% respectively. For the performance of the various classifiers described above, please see Figure 3. The ensemble methods of classification and linear models perform the best with an accuracy over 61%. The worst classifier was the Gaussian Naive Bayes classifier with an accuracy of only 5%. We analyze these results in the discussion section later in the report.



Classifier	Train Accuracy Mean	Validation Accuracy	F1 Score Mean
BaggingClassifier	0.609	0.615	0.512
DecisionTreeClassifier	0.607	0.613	0.506
SVC	0.610	0.613	0.512
RidgeClassifierCV	0.603	0.613	0.515
GradientBoostingClassifier	0.610	0.613	0.499
RandomForestClassifier	0.609	0.612	0.502
LinearSVC	0.601	0.612	0.503
MLPClassifier	0.604	0.611	0.488
BernoulliNB	0.594	0.610	0.471
LogisticRegression	0.600	0.609	0.492
XGBClassifier	0.604	0.609	0.518
AdaBoostClassifier	0.594	0.608	0.465
KNeighborsClassifier	0.592	0.567	0.483
Perceptron	0.344	0.318	0.325
GaussianNB	0.047	0.050	0.034

Figure 3: Classifier performance on pass result attribute ordered by validation accuracy

For the Decision Tree Classifier, we additionally generated visualizations from the selected decision trees. The intent was to track which attributes were chosen as splitting rules, and we found that when targeting the pass result for classification, the decision trees were relatively consistent in rule selection. Typically choosing Pass Rushers, Type of Drop Back, and Down as the splitting attributes. This did not always occur in the same order, but invariably all three will be among the first few attributes selected by the algorithm.

After looking at the pass result as a target for classification, we additionally decided to classify on other attributes, "personnelO" and "offenseFormation" attributes, which refer to the offensive personnel on the field and the selected formation of the players on the field, respectively. The accuracies for these targeted attributes were actually higher than for pass results, averaging about 0.75 across a number of different classifiers. To get better insights into these values, we again visualized the constructed decision trees for these attributes, and found that these two attributes will regularly choose to split on each other, in addition to defenders in the box, indicating that they may have a high degree of impact on each other.

NFL Player Insights

Motivation

Since we didn't explore this dataset with any particular or explicit goal, we sought to try and extract a few different insights. Since the dataset came with information on every single active NFL player as well as a few of their physical characteristics, we were wondering if we could do any data analysis on the players themselves. Specifically, we were curious to see if we could classify or otherwise sort players in various positions by those physical attributes.

It's well known that players in different positions require different physical talents and skills. For example, most wide receivers will need to be fast, agile, and catch well. On the other hand, offensive linemen will need to be exceptionally strong, larger bodied, and generally bulkier. Tight ends are a mix of the two, needing to both catch well and block well. Therefore, given that each position has their own unique traits, we wanted to see if we could actually quantify said differences.

This specific bit of data exploration doesn't have a ton of applications towards the NFL in the real world. It could surely be useful to know the "formula" that makes a complete runningback or a quarterback, but the included dataset doesn't actually indicate which player is considered superior to the others; thus, grouping the data in this way only tells us what attributes make a NFL player and not a good NFL player. And even in the NFL itself, there is significant differentiation within positional groups. For example, Derrick Henry, a runningback, is far taller and larger than the average runningback and yet has proven himself to be one of the best runners in the league. Thus, this bit of data analysis is more motivated by curiosity than the ability to truly benefit an NFL front office.

Methods

Clustering

First, we tried a clustering approach, just to see if we would be able to naturally cluster each of the players into their respective positions by height and weight alone. We started with only height and weight because those were the only attributes provided by the original NFL dataset. A plot of height and weights before clustering is shown in Figure 4.

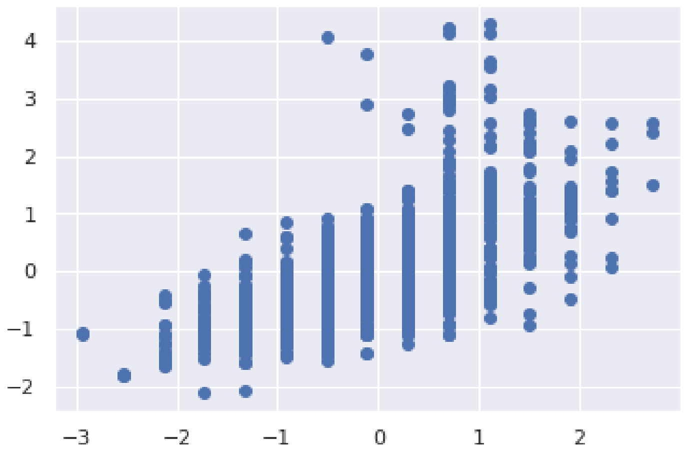


Figure 4: Pre Cluster Plot

Before doing any data analysis, we can already see that the dataset is sub-optimal. There's not enough of a spread on the players' heights to actually be able to get significant differentiation, meaning that it's mostly just arranged in vertical bars by heights.

Even so, we still tried to cluster this data. The first clustering approach that we used was K-means clustering. We played with different numbers of clusters; the perfect number of clusters would be 15, since there were 15 unique positions included in the dataset. Ultimately, it really didn't matter, because the clustering failed to produce any meaningful results regardless of the number of clusters we chose.

We also attempted to cluster with DBSCAN, just to see if a different approach would be able to produce new results. The best eps threshold we found was .4, but even so, we once again failed to produce anything meaningful.

Classification

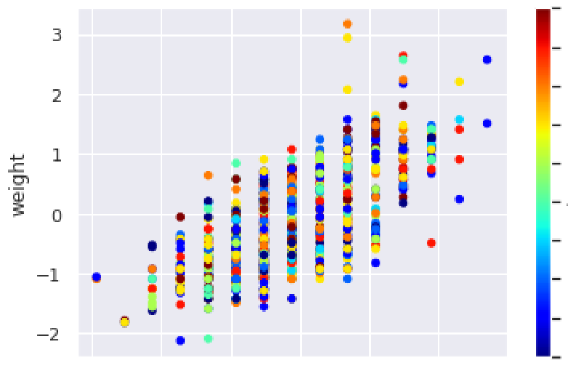
After clustering proved unfruitful, we tried to classify players instead to see if any useful information could be derived therein. Similar to before, we tried out a number of different classifiers, although they all performed relatively similarly. This was probably because of the lower number of attributes that we were using.

As mentioned above, we also tried to append additional information about the physicalities of the athletes in order to see if we could glean more granular distinctions. We included NFL combine data and fed that into the classifiers as well. For each classifier, we also used GridSearchCV to try and optimize the parameters, seeking to find the parameters that would result in the highest accuracy scores.

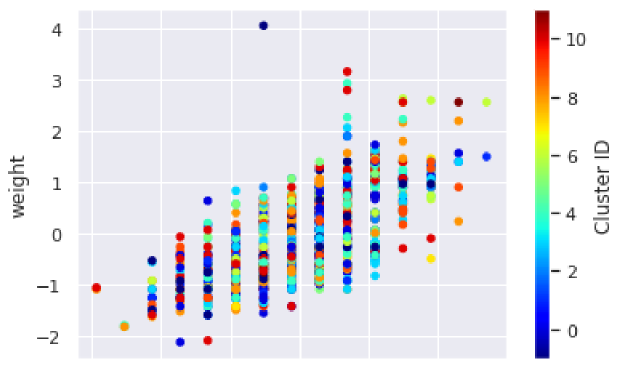
Results

Clustering

As hypothesized, there simply wasn't enough information from heights and weights to cluster in a worthwhile manner. An image of the K-means and DBSCAN clusters is included below.



(a) k means



(b) DBSCAN

The big issue that we ran into was essentially that there weren't actually distinct clusters in the data. Although the hypothesis was true - that different NFL positions require different physical traits - the variance within each position made the clusters sufficiently indistinct to truly cluster.

Classification

We definitely managed to produce better results with classification. Initially, classifying off of only height and weight, we got an accuracy of around .6, which obviously wasn't great. But after appending combine data, we managed to get our accuracy up to .875 on a simple decision tree classifier.

The best parameters for the decision tree was using a max depth of 7 and no cap on max features. The classified classes weren't too imbalanced, except for QB; there were 237 wide receivers, 214 running backs, 170 tight ends, and 10 quarterbacks. There's not much we can do to mitigate quarterback imbalance, though, since each team fields so few of them, and they're not specialized to do anything physically. However, it's relevant to note that we also attained a precision of .876, a recall of .875, and a f-score of .869 - all of which are decent, and definitely significant improvements on the first iteration.

Discussion

Pass Play Insights

As mentioned in the results section, the assorted ensemble methods and some linear models were able to outperform the most frequent baseline classifier by around 3 percent accuracy. This seems to be because these classifiers can handle varying attribute types and are resistant to outliers and irrelevant attributes. The ensemble methods (bagging, boosting, and random forests) are especially good at handling varying attribute types and since their underlying estimators are decision trees. SVM with the radial basis function also performed well and can handle irrelevant attributes and various different types of attributes. Gaussian Naïve Bayes performs notably worse than all the others, which makes sense because it works well with continuous attributes, but we had quite a few asymmetric binary attributes from the one-hot encoding. Bernoulli Naïve Bayes performed much better for this reason. The disparity between the various ensemble methods was not great enough to conclude that any is significantly or concretely better than the others. While this is better than nothing at all, it's obviously not the result that we were hoping for. We think this is probably the result of a few different problems.

First, the dataset itself just wasn't really conducive towards classification tasks. As evidenced by our initial dataset exploration, it's evident that none of the various attributes are especially correlated with the attribute that we were seeking, pass results. Therefore, there was no fundamentally predictive variable that made it easy to predict the results of pass results. While our hypothesis was that perhaps certain defensive formations or offensive lineups would make it easier or more difficult to complete passes, it seems like this wasn't necessarily true.

Second, there was a significant class imbalance problem within the dataset itself. As mentioned before, the number of completions was significantly greater than incompletions, and as a result our biggest error was that many incompletions were classified as completions. We tried a number of things to address the class imbalance problem: stratified sampling, oversampling, undersampling, etc, but none of that seemed very effective. Although the class imbalance was a problem, we think the biggest issues come from the above issue; at its core, it's very difficult to predict the results of NFL plays from pre-snap information.

This brings us to the final issue with our analysis, and it comes down the inherent problem that we are classifying. At its core, too much of the game of football really comes down to execution. A defensive tackle being able to jump off the line of scrimmage a quarter of a second faster than usual can impact a play so much more than whatever formation the offense was using. At the end of the day, the pre-snap setup can only provide so much more information.

Additionally, we found a few more interesting patterns and insights when analyzing these datasets. Specifically, we found that actual incompletions were far more likely to be misclassified/predicted as completed passes, compared to the reverse (actual completions predicted as incompletions). This makes rational sense, since a good initial "setup" for a pass completion can still be derailed by errors in execution, and otherwise impacted by individual player performance. Furthermore, as part of the feature engineering, we found that the pass completion outcome does not seem to be impacted by time remaining in the quarter or game, such as when binning all times that are 2 minutes or less. This provides evidence against the common narrative of players, across all sports, performing notably differently in "clutch" situations.

Ultimately, although our results were far from optimal, we do think that they provide a glimmer of insight into NFL pass plays.

NFL Player Insights

We encountered more success with analyzing NFL players. The clustering of heights and weights alone failed, as is incredibly obvious from the results. This was almost certainly because there just weren't really distinct clusters for the positions, and the heights weren't continuous enough variables to create said distinct clusters.

However, our classification of players into positions given NFL combine data was significantly more successful. If we used a baseline classifier of the most frequent dataset, we would only expect an accuracy of around 40 percent. Our 87 percent accuracy from our simple decision tree far outperforms that.

This result really only affirms the hypothesis that was already largely known: that different NFL players require different physical skill sets. This insight was definitely an easier one to learn than the first one, but it's still interesting to see how the positions can be broken down into statistics.

Conclusion and Future Work

Although we spent a considerable amount of time on this project, it wasn't nearly as large a project as professional statisticians or actual researchers. As a result, the analysis that we did pales in comparison to the ocean of data that is actually available in the NFL and in the game of football. While we did this project mostly as a way to apply the data mining concepts that we learned and to explore data out of curiosity, there is definitely a significant amount of untapped potential in football data analytics.

Even within the dataset that we downloaded from Kaggle, there was a significant amount of data that we didn't use. For example, the NFL includes player tracking data into their publicly available datasets, taking snapshots of each play that tells you where each player is at any given moment in time, as well as their velocity and acceleration. If we were to incorporate post-snap data into our plays, we would surely be able to predict the outcomes of plays far better. Even pre-snap, perhaps we could analyze pre snap motions to see if certain motions throw the defense off guard more so than others. If we wanted to extend this project further, we would probably take the step of doing analysis on more granular data such as the player tracking.

On the NFL player's side, there's also a number of ways that data analytics can be extended. Although not much was shown about players in the given dataset, data analytics can help with a team's drafting needs, filling in positions that are scarce. It can also help players stay healthier and extend their longevity, as well offer players insight into their own play.

If you watch football, you've probably seen at least a couple of the AWS Next Gen Stats advertisements - the most recent ones featuring Christian McCaffrey and Deshaun Watson - that are a clear indication of how machine learning and statistical analysis have made their way into the football mainstream. And it's definitely here to stay. We expect the NFL data analytics market to grow exponentially in the coming years. We've already seen analytics affect football play by increasing the number of times teams go for it on fourth down, and we're excited to see how football improves even further in the future.

Group Contributions

Everyone in the group contributed to the project. Broden did most of the data exploration, which was looking for correlations between attributes in both datasets. All three members worked on the pass play analysis, which included preprocessing the data, playing around with attributes, trying out different classifiers, and tuning parameters. David worked on the NFL player analysis, which included the clustering and classifying attempts.

Similarly, everyone contributed to the non-coding aspects; all three group members worked collectively on the proposal, the interim report, the proposal, and the final report.

References

[1] Tianqi Chen and Carlos Guestrin. "XGBoost: A Scalable Tree Boosting System". In: *CoRR* abs/1603.02754 (2016). arXiv: `1603.02754`. URL: `http://arxiv.org/abs/1603.02754`.

[2] National Football League. *NFL Combine Dataset from 2002-2018*. Data retrieved from Kaggle, `https://www.kaggle.com/savvastj/nfl-combine-data`. 2018.

[3] National Football League. *NFL Pass Play Dataset from the 2018 Season*. Data retrieved from Kaggle, `https://www.kaggle.com/c/nfl-big-data-bowl-2021/data`. 2018.

[4] F. Pedregosa et al. "Scikit-learn: Machine Learning in Python". In: *Journal of Machine Learning Research* 12 (2011), pp. 2825–2830.