NFL Play by Play Prediction- Final Report

Jibran Gilani (jg793), Milan Shah (mrs282)

December 2019

1 Introduction to Problem

During each NFL game, we only see about 11 minutes of action during the 60 minute runtime. Even then, the action and results are only a fraction of the research and work that gets put in by coordinators and coaches to prepare for matchups. As a defensive coordinator, part of that research is being able to guess what the opposing offense will do for each play and provide the proper scheme and design against that play. This process depends heavily on their ability to correctly predict what the offense has planned. The two main types of plays in the NFL are run and pass plays- run plays refer to a quarterback handing the ball off to the running back and pass plays refer to the quarterback throwing the ball to an eligible receiver. Each team has 53 players on their roster and 46 are active on gameday- on the field, there are 11 players on offense and 11 on defense. Those 11 defenders often change- a coach may want to have smaller, quicker players in passing situations and larger, heavier players in running situations. The defensive substitutions are crucial to keep players fresh and rested but mainly to matchup up well with what the offense will execute. Knowing what play they are planning will heavily influence these substitution decisions. Play designs in the NFL are complex and it often takes a while to fully grasp the playbook. However, lack of talent could by mitigated if a proper defensive scheme is used. A well-timed blitz (sending defenders to rush the quarterback) can stifle an offensive play and knowing that a team will run the ball should help stop them at the line of scrimmage. The combination of having the right defensive players on the field with the correct play call is the perfect recipe to stopping an NFL offense. Our problem could potentially help with both of these facets.

Research question: Would it be possible to use machine learning techniques to help NFL defenses combat offenses by predicting the most likely play the offense will run?

2 Data Description

2.1 Dataset

We used a detailed NFL Play-By-Play Dataset that spans from 2009-2018 that has has 356,768 rows and 100 columns. Each row represents the outcome of a single play in a game and the columns provide information including game score, time, play type (run vs pass), game situation (down, distance, and yard line), result, and more.

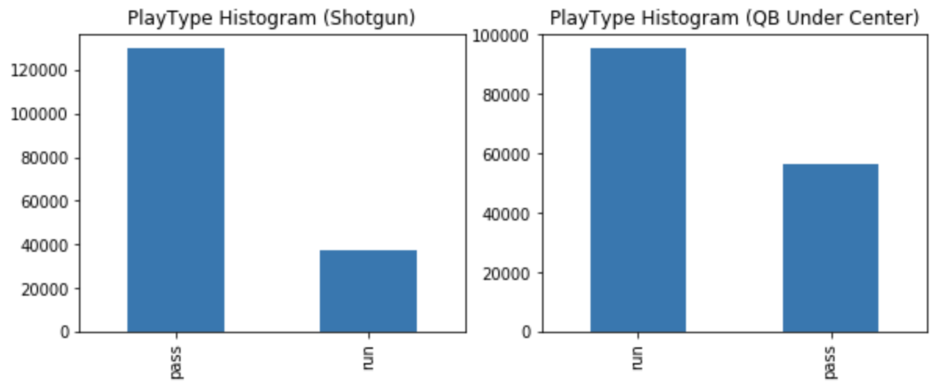
2.2 Data Analysis

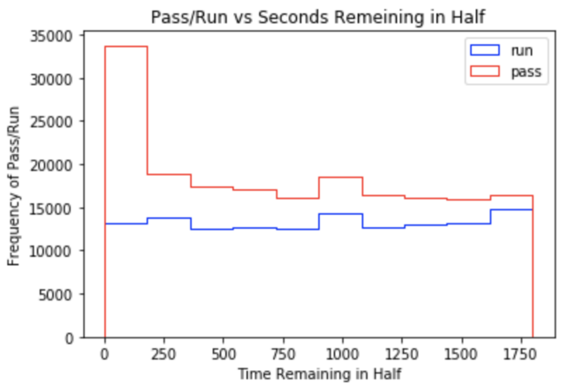
After analyzing the data, there appears to be a division into three distinct categories- game factors, play results, and probabilities. The game factors include information such as time left, score, down and distance etc. and is what we will use for our predictions. We are not as concerned with the result of the play since we only what to predict the type of play beforehand. We are unsure how exactly the probabilities are computed, but perhaps they can be used in later analysis. Examples include touchdown probability, which should be the likelihood that a team scores a touchdown on the given play. We are using 15 columns for our final dataset and created an additional 17 features for a total of 32, and dropping all rows that are not a pass or run play (other plays include special teams plays, such as field goals and punts, and also specialized plays including QB kneel-downs and spikes). In the future, we may want to have multi-class classification, but for now we are focusing only on run and pass plays, by far the two most common plays. We have over 300,000 plays and have narrowed down the features to the most important ones that sufficiently portray the game situation. One of the major benefits for this problem is that the data have will be identical to any future data regarding this problem. Unless the rules of the game drastically change, the data this model will see in any future use will be identical to the train and test data it is being fitted and tested on. We do not have to worry about the distribution of the data it is learning being different from the distribution it will see in applications.

3 Features

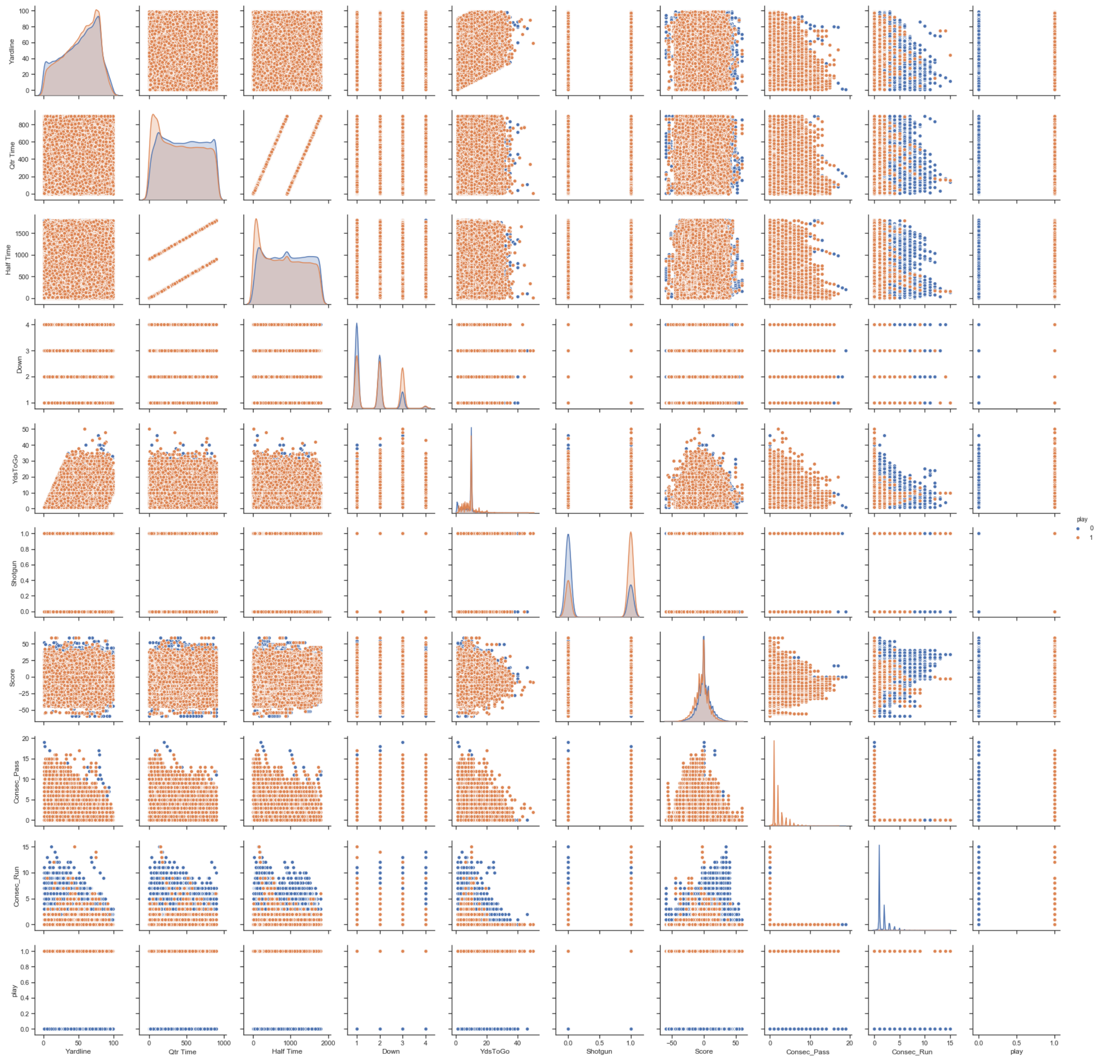
3.1 Game Factors

We began by choosing features that we categorized as game factors. We did not want to use probabilities just yet as they are not as readily available to NFL coaches and staff as game factors are. We began by treating each play as its own entity. This meant that we only considered features if they would be known by simply looking at the field. This included quarterback position, yard line, down, yards to go, time remaining, and scores among a few other features. The feature for quarterback position we used is ”shotgun”, which is a formation in football in which the quarterback stands a few feet behind the lineman who will snap the ball to him- this formation often used for pass plays as it gives the quarterback slightly more time to allow the play to develop and receivers to get down the field. After deciding which features to use, we created a scatter matrix to graph all the features against one another along with their corresponding histograms of values. We plotted the scatter plots using the play label as the color of the data point. This helped us determine which features, which pair of features, and which polynomial order of features show relevant information. We looked for pairs of features where there was a distinct split between red and blue points in order to determine the features that can be best used to predict the best play.





An important feature we encountered was half seconds remaining. As we can see in the graph to the left, the total number of plays run when the half is close to an end is much higher due to teams calling timeouts and quickly moving the ball to get into a scoring position before either half ends. Additionally, the plays that increase are passing plays since they allow teams to conserve more time rather than running the clock out and losing possession of the ball.



3.2 Time Series

We used these game factor features as our base for X, but we were still missing many features. Since football is a game dependent on previous actions, we decided we could view this problem similarly to a time series problem where features can include results from previous states in the game. These can include statistics on previous plays run in the game and how a team had been playing up to the given moment. We decided to implement time series features with two different approaches. First, we calculated a running total of the consecutive pass plays and consecutive running plays previous to the current play being analyzed. Second, we created features for what the previous play was. We took the previous "yard line", "half seconds remaining", "down", "yards to go", "shotgun", "no huddle", "team timeouts remaining", and "score" and appended it to the current play. We choose these specific features since they are indicative of how the previous play went (ie. how many yards were gained, how long the play took, were any timeouts used). For plays that were the first of the drive, we used an default vector to indicate that there was no previous play by choosing values that could never occur.

3.3 Team Specific

The last features we added were offensive, defensive, and current teams. We added these by taking the hash of the team name so that each team had a unique int value. We added these features so that we could separate how certain teams usually play offensively and how certain teams usually play defensively. This helps for situations where certain teams have great run defenses so teams will pass against them, or when teams have a great quarter back so they usually pass the ball.

3.4 Non Linear

After adding all of our features, we decided to see how non linear features would fare in our different models. We created new features of column multiplied together on different order polynomials like down $\times$ yards to go, down$^2\times$ yards to go^2, and several others. We believed that this would better help our models differentiate between how far of a distance the team needs to go relative to how many plays they have remaining.

4 Cleaning

The dataset was mostly clean except several columns had NaN values, which we were able to drop given the size of our dataset. For example, for a pass play the field ”Run side” would be NaN since the ball was not even run. Most of the columns in the data did not have outliers except for “Air Yards” which are the length that the ball travels in air during a pass play. There we found several examples with large negative values, which seem to be typos. However, we did not use this column in our dataset since it is a feature describing the result of the play. After looking at descriptive statistics (min, max, median, mean etc.) for the other features, none appeared to have outliers (downs were from 1 to 4, yard line from 0 to 99 and so on, meaning that the features made sense in football game-play terms).

5 Train, Validate, Test Split

5.1 Time Series

Our data does have time-series elements given that time within the game and season do matter, however we found little evidence of trends across years. We trained on the first 7 years of our data and used the last year for validation. We choose this split initially because we wanted to be able to replicate a production environment, where coaches would have access to previous years data to study.

5.2 K-Fold Cross Validation

Since our time series features only consisted of data within a given game we also implemented 3-fold cross validation to compare our estimates over the entirety of our dataset. The accuracies we received were very similar to each other (numbers are below) reassuring there was no overfitting.

6 Model and Errors in Fitting

6.1 Logistic Regression

6.1.1 Game Factors

The first model we fit was a logistic regression model. We wanted to see how well our features could be linearly separated to provide a clear distinction between pass and run. We used the sklearn logistic regression classifier with the BFGS solver. We used all our game factor features in training and refrained from using time series data until later for a fair comparison. Since logistic regression is a linear classifier, we also refrained from using team specific features as the hash's have no real meaning on a scale.

Training Accuracy: 67.43%
Testing Accuracy: 67.79%

Because the accuracy was not 100%, we are reassured that the dataset is not linearly separable. This is in-line with our rationale as well since football has many random factors that we can not account for giving a small overlap between pass and run plays.

6.1.2 Game Factors and Time Series Features

We then fit the logistic regression classifier with both game factors and time series features. We did expect the model to perform much better as many of the time series features were very indicative of how a team preferred to play.

Training Accuracy: 76.32%
Testing Accuracy: 74.81%

The accuracy increased greatly from before indicating that some of the new time series features we added helped the model understand how a team tended to play within a game.

6.1.3 Game Factors, Time Series Features, Non Linear Features

Next we added in our non linear features to see how they would help a linear classifier. We didn't expect much of an increase as many of our non linear features may not have been linearly separable.

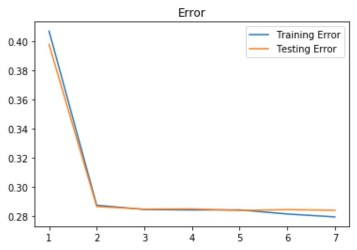
Training Accuracy: 69.72%
Testing Accuracy: 68.62%

The accuracy decreased slightly from before indicating that we were correct in assuming some of the non linear features may have confused the model as they were not linearly separable.

6.2 Random Forest

6.2.1 Game Factors

Since the dataset is not linearly separable, we decided to classify our data with random forests next. We initialized our model using sklearn with 100 random forest estimators averaged with a depth ranging from 1 to 7. We first used the game factor features to see how well looking at the field would be able to predict play type.

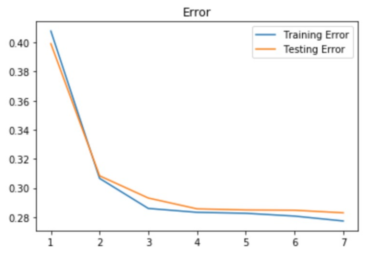


Training Accuracy: 72.10%
Testing Accuracy: 71.60%

We see from above that the error converged at a depth of 2 and higher depths didn't make a difference. The accuracy was higher than the accuracy of the logistic regression classifier possibly due to the fact of most of the features not being linearly separable where a random forest classifier would be able to circumnavigate that (random forest can deal with any type of data also).

6.2.2 Game Factors and Team Specific Features

We then decided to add in team specific features so that the classifier would be able to make specific predictions for certain offensive/defensive teams. This helps in calculating how teams play against specific defenses and how specific offensive teams typically play.

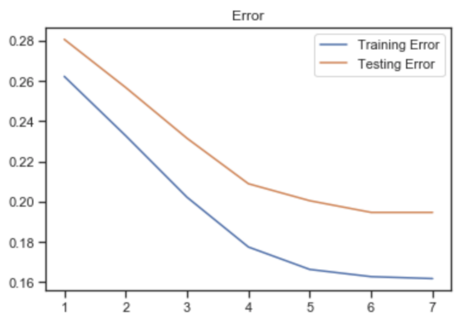


Training Accuracy: 72.30%
Testing Accuracy: 71.70%

We see that the model training with team specific features as well takes a bit longer to converge, but the convergence is a bit smoother indicating that a few of the features have similar importance rather than only one feature making the decision. The accuracy improved slightly from the previous model with no team specific features, but it wasn't anything significant.

6.2.3 Game Factors, Team Specific Features, Time Series Features

We then decided to add in time series data to look at game trends and how they factor into a team making a decision on the next play to run. This helps in determining when a team has being doing especially well passing, so they pass again or when they have just made a large gain, so they decide to run the ball.



test_score	train_score
0.833644	0.832290
0.833952	0.832894
0.830544	0.834038

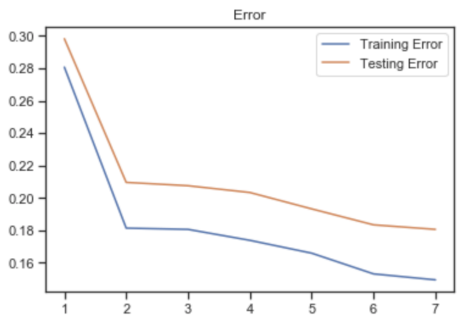
3-Fold Cross Validation

Training Accuracy: 83.82%
Testing Accuracy: 80.53%

We see now that the model converges smoothly seemingly giving many more features equally important weights. The accuracy improved greatly by over 10% and we are now able to predict the correct play with upwards of 80% accuracy. We additionally ran 3-fold cross validation on the model to ensure that our results were not bias and had low variance. The results showed low variance as all accuracies were around 83%.

6.2.4 Game Factors, Team Specific Features, Time Series Features, Non Linear Features

We finally added in non linear features to help our model generalize on some features better (ie. different combinations of down and field placement).



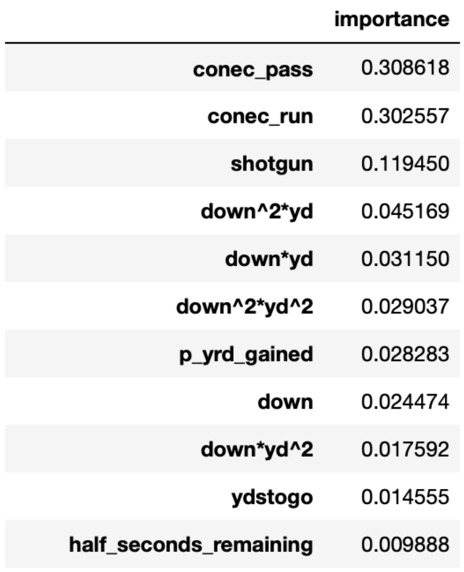
test_score	train_score
0.830377	0.832577
0.829950	0.830846
0.833020	0.831585

3-Fold Cross Validation

Training Accuracy: 85.05%
Testing Accuracy: 81.94%

We were able to receive slightly higher accuracies with non linear features due to the model being able to understand the relation between features like down and yards to go as opposed to just analyzing them individually.

6.3 Feature Importance

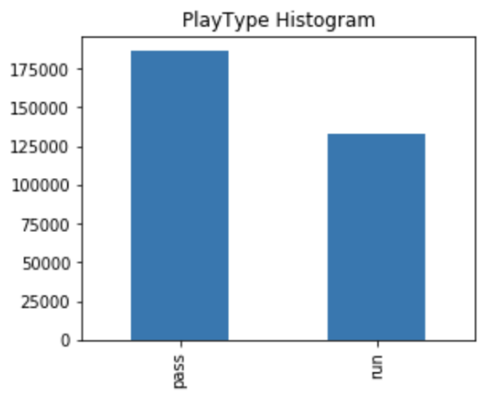


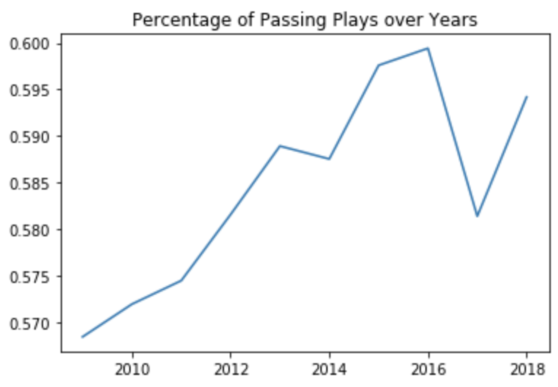
	importance
conec_pass	0.308618
conec_run	0.302557
shotgun	0.119450
down^2*yd	0.045169
down*yd	0.031150
down^2*yd^2	0.029037
p_yrd_gained	0.028283
down	0.024474
down*yd^2	0.017592
ydstogo	0.014555
half_seconds_remaining	0.009888

The feature importance table indicates which features the random forest model found to be most important in its classification. The time series stats for number of consecutive runs and passes were the most indicative of what play was to follow, and in third is shotgun. Now the shotgun is a feature that is difficult to understand in this scenario because it can change even after an offense and defense line up in their formations, but if it were to change the defense would have just enough time to change their formation as well (although they would not be able to make substitutions). The rest of the features that ranked most importance dealt with the game situation. We expect down and distance (ydstogo) to factor heavily because early downs and short distances lead to running plays and longer distances lead to passing plays. Time seems to play a small role as well- half-seconds remaining will only be relevant when it becomes small as that entails the end of both the first and second half. Often times, teams will pass the ball more (running the ball makes it difficult to stop the clock, which is crucial when there isn't a lot of time remaining in the half).

6.4 Model Goal

Over the past several decades, teams have shifted away from a run-first approach into a pass-first approach, but our dataset appears to have consistent ratios of runs to passes (around 58% pass). The baseline accuracy we were aiming for was 58%, so anything better than this would constitute a model that would predict pass vs run better than random. Given a final accuracy of 82%, our model appears to performs 24% better than a random model would. The dataset we were given did not have a major balance issue so we did not have to account for any balance accuracies.





7 Conclusion

7.1 Recap

The problem we are trying to deal with is play prediction in the NFL and attempting to predict whether the next play is a run or a pass. The NFL, like many sports nowadays, are moving to a more analytics-based decision process and there is plenty of data that football generates. The dataset we found consisted of every play from every game from 2009-2018. Before a play, certain features are known, including time left in the game and quarter, the down, distance, and yard line and previous plays.

7.2 Results

	Logistic Regression			Random Forest			
Features	Game Factors	Game Factors Time Series	Game Factors Time Series Non Linear	Game Factors	Game Factors Team Specific	Game Factors Team Specific Time Series	Game Factors Team Specific Time Series Non Linear
Training Accuracy	67.43%	76.32%	69.72%	72.10%	72.30%	83.82%	85.05%
Validation Accuracy	67.79%	74.81%	68.82%	71.60%	71.70%	80.53%	81.94%

Table 1: Model Performance

	Run_Prediction	Pass_Prediction
Run_Actual	9117	2798
Pass_Actual	2527	14975

Table 2: Confusion Matrix

The confusion matrix above is for the final model, which was the random forest model with game, time specific, time series and non-linear features. The precision for run and pass is 0.784 and 0.842 respectively, and the recall is 0.766 and 0.855. This is good in context of the predictions because correctly identifying a pass is arguably more important than a run because an error on a pass play is more likely to be costlier (i.e. give up more yards) than a run. As far as fairness metrics go, we see that our model does not have equalized odds (different rates of true and false positives for both groups).

7.3 Production

The main beneficiaries of a model like this would be defensive coordinators, who are responsible for managing and calling plays for the team's defense. These coordinators analyze the game situation and opposing team's tendency in order to decide what play to call for their defense, which is then communicated to one of the defenders through a headset. If they can optimize their play call to neutralize the offensive play call, then it would put them at an advantage. However, the correct call is only half the battle, with execution being as if not more important, and that is a component that the model cannot help with. An incorrect call could be costly as having smaller defensive backs to defend against the run or larger linebackers to defend against the pass would put the defense at a matchup disadvantage before the play even starts. From the precision and recall stats in the section above, it is preferred to correctly identify passing situations due to the risk of a larger play. In terms of the prediction potentially affecting an outcome, the prediction will not be public knowledge, so unlike something like college rankings, it will not make an event more or less likely to happen. However, what could be destructive is that if the offense is able to have a model of their own, they may know what the defense is expecting and then counteract that and it can go back and forth. At a larger scale, while some money and the joy of fans is on the line, a mistake in the prediction of this model would not have the same dire consequences as mistakes in medical or financial models would.