# Leveraging Machine Learning to Predict Playcalling Tendencies in the NFL

by

Udgam Goyal

Submitted to the Department of Electrical Engineering and Computer Science
in partial fulfillment of the requirements for the degree of

Master's of Engineering in Computer Science and Engineering

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

February 2020



Author .....
Department of Electrical Engineering and Computer Science
December 19, 2019

Certified by .....
John V. Guttag
Dugald C. Jackson Professor of Computer Science
Thesis Supervisor

Accepted by .....
Katrina LaCurts
Chair, Master of Engineering Thesis Committee

# Leveraging Machine Learning to Predict Playcalling Tendencies in the NFL

by

Udgam Goyal

Submitted to the Department of Electrical Engineering and Computer Science
on December 19, 2019, in partial fulfillment of the
requirements for the degree of
Master's of Engineering in Computer Science and Engineering

## Abstract

In this thesis, we apply four machine learning models to NFL play-by-play data from 2009-2018 to predict whether a team will run or pass the ball on a given play. We tested our models using league-wide and team-specific data in five different situations on the field. Our best league-wide models achieved a test accuracy of 80% and our best team-specific models achieved a test accuracy of 86%. Relative to the baseline of the run-to-pass ratio, the best league-wide models achieved an increase in accuracy of 25% and the best team-specific models achieved an increase of 27%.

Our models showed that the Tennessee Titans, the New York Jets, and the Cincinnati Bengals have been the most predictable offenses in the NFL over 10 years. We found that a team's in-game run-to-pass ratio and their win and score probabilities are the driving factors for offensive play-calling. Additionally, our results show that teams are more predictable later in games, and that less predictable teams tend to experience greater success offensively.

Thesis Supervisor: John V. Guttag
Title: Dugald C. Jackson Professor of Computer Science

# Acknowledgments

First, I would like to thank my thesis advisor, John Guttag, for his advice and insights. This thesis would not have been possible without his unfaltering support and enthusiasm. Professor Guttag's patience and guidance were pivotal to this thesis and my growth throughout these past two years, and for that, I am very grateful.

Additionally, I would like to thank my family, specifically my mom, dad, and brother. It's hard to put into words how thankful I am for them and everything they've done for me. They support me in each and every step of my life and they keep me motivated and inspired through thick and thin. They have given me their everything, and they are the reason I am who I am today.

# Contents

# List of Figures

# List of Tables

# Chapter 1

# Problem Statement

Football is often characterized as a sport of physicality. To the untrained eye, football is seen as a brutish sport, with large men hitting and tackling each other. However, football is as much a mental game as it is physical. The cerebral nature of the sport arises from its strategy. The strategy of football occurs in various aspects of the game, from game management to coaching to personnel decisions. One of the key strategies in the game of football is play-calling. In this thesis, we use machine learning models to predict play-calling tendencies in the National Football League.

## 1.1 Value Proposition

Coaches who are able to predict an opposing team's future play calls have an inherent advantage, for this provides the coaches with an ability to attack the weaknesses of the opposing team's expected play call. However, being able to understand a team's own predictability is arguably more important. Insights gained from understanding a team's own patterns of play-calling can reduce future predictability, providing another competitive advantage over opponents.

Given the abundance of effective machine learning methods today, it has become increasingly feasible to leverage machine learning to predict future outcomes based on past data. In sports, we have seen an increased use of machine learning methods through data analytics and computer vision in baseball [14], basketball [12], and

soccer [15]. Multiple organizations, from the Houston Rockets to FC Barcelona, have invested in machine learning and have experienced tangible results.

## 1.2 Why Football?

Despite the success of machine learning in other sports, football organizations have been slow to adopt these methods. In a sport as strategic and pre-determined as football, machine learning's ability to predict outcomes accurately based on tangible data should be extremely useful. However, three main factors cause machine learning's paucity in the sport of football: a lack of quality data, the ability to audible, and the restrictions placed by the National Football League upon the use of technology during games.

### 1.2.1 Lack of Quality Data

Machine learning requires large quantities of well-labeled, diverse data for accurate model creation, but data for football is often inaccessible or low quality. The most useful data in football for machine learning is play-by-play data, which provides information about the situation of a given play (e.g down and distance, yard-line) and the results of the play (e.g yards gained, name of ball carrier). However, existing play-by-play data lacks features regarding important intricacies of the game.

For example, data around the movement of specific players is difficult to extract since players are often extremely close to each other during a given play. This makes it difficult for computer vision and even human methods to extract movement information. Specifically, it becomes difficult to track the movements of offensive and defensive lineman, who are within inches of each other throughout a play. Though there are recent developments of player-tracking through sensors in helmets and other equipment used in-game by players, there is currently not enough movement data for use in machine learning methods. As a whole, data for machine learning in the National Football League is limited.

### 1.2.2 Ability to Audible

The ability to change a play at the line of scrimmage results in missing information within play-by-play data. On offense and defense, a team may change their formation, play call, or personnel at various times before the snap. Each of these changes play a role in the final play call, but are not accounted for in public play-by-play data. For example, let's say an offensive team initially calls a run play down the middle of the field. However, once both teams line up, the offensive team notices that the defensive team will be blitzing players in the middle, so the offense changes its play call to a pass to the outside to counteract the defense. Though the situation may call for a run play based on other factors such as down or distance, the defensive formation and coverage may result in a last-second change in play call. Audibles like this are often difficult to discern in current play-by-play data.

In fact, it may be impossible to extract this information. The only way to obtain this information would be through getting data from a specific team which tracked its initial play calls, its audibles, and final play calls. However, most teams will not have tracked play calls to this extent.

### 1.2.3 National Football League Rule

Existing National Football League rules limit the use of technology on the sidelines and the coaches' boxes to Microsoft Surfaces with only one function: displaying photographs. This ensures that no real-time modeling can be used by coaches and players during a game. The idea behind this, according to the National Football League, is to keep the focus of the game around a team's coaching skills rather than a team's technological prowess. This hinders the development of technological arms of football organizations, since teams have less incentive to invest in machine learning methods for their organization.

## 1.3 Related Work

Much of the previous work done in football through machine learning is for game prediction and commercial use. For game prediction, previous work has been focused on figuring out which team will win a specific game. Bosch utilizes machine learning methods to predict winners of NFL games [6], while Klein and Frowein leverage a logistic regression model with player ratings and weather factors as inputs for game prediction [13]. Lock uses random forest models to evaluate win probabilities of NFL teams in-game [19], and Miceli, Balreira and Tegtmeyer utilize Markovian methods to predict NFL game results [21]. Outside of machine learning, others have leveraged statistical models to predict games as well, such as Lee and Danileiko, who use a statistical Elo method based on performance histories of NFL teams for game prediction [16].

Similarly, researchers have explored predicting the margin of victory in a given game. Warner uses machine learning to predict the score margin of NFL games [28], while Glickman and Stern use Hidden Markov models to predict game scores [10]. In both cases, their results were compared to the expected score line for each game.

Data analytics and machine learning have often been used in the entertainment aspect of the NFL to showcase interesting in-game statistics and to evaluate players, as shown in figure 1-1. Mallepalle and Pelechrinis utilize computer vision methods to evaluate NFL passers and track players given image data [20], and Ajmeri and Shah use computer vision to classify game film and track players [3].

Existing research on play prediction in the NFL has been focused on leveraging simple methods to predict run-pass or predicting the result of a specific play. Lee, Chen, and Lakshman explore play prediction based on play type, but do not use any deep learning or personnel information in their models [17]. Teich, Lutz, and Kassarning use machine learning to predict the yards gained in a specific play and whether the play results in a touchdown or a first down with limited accuracy (around 60%) [27].

This thesis is novel in that it attempts to leverage a wide variety of machine

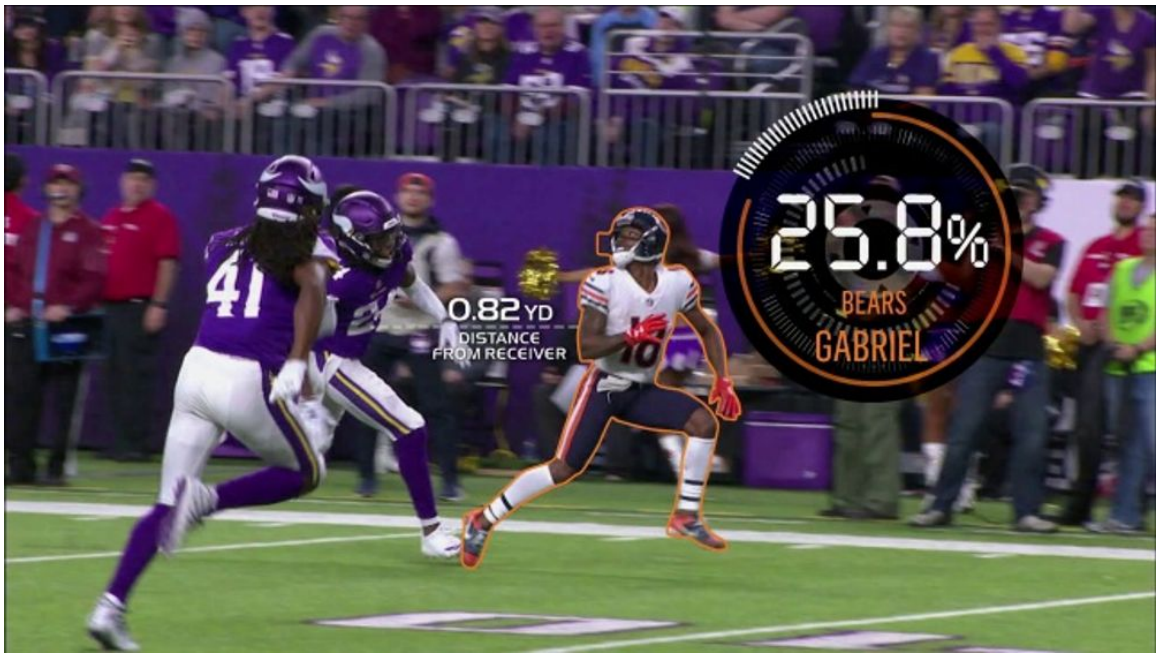



Figure 1-1: Amazon's Next Gen Stats uses Computer Vision Methods to Show Statistics During Games

learning methods, from artificial neural networks to random forests, to create highly accurate models for predicting run or pass relative to the probability of run or pass on a given play. We focus on predicting run and pass in specific game situations because it provides the highest value proposition for an NFL organization. After speaking with the Director of Data Science for the New England Patriots, he mentioned that the focus of situational play prediction should be high accuracy of prediction. For coaches to actually implement such a model in their play-calling, they need data-driven insights which are practically guaranteed to work. Therefore, through focusing only on binary classification, we intend to create value through highly accurate models.

## 1.4 Thesis Roadmap

In this thesis, section 2 provides an in-depth description on the data used in the machine learning models, section 3 breaks down the four different types of models created, and section 4 elaborates upon the results and important insights from our models.

# Chapter 2

# Data

One of the key aspects of machine learning is the quality of the input data. Without high quality data, even the most advanced models are unable to extract meaningful insights and develop predictive capabilities for a problem space. For the NFL use case, in which data is sparse and limited, data conditioning methods and new data sources are necessary to provide high quality data for machine learning.

For football, the most granular, publically available data that can be used is play-by-play data. Play-by-play data for the NFL from years as early as 2009 can be found on the NFL's official website [2]. The historical play-by-play data used in this project, which ranges from the 2009 to the 2018 seasons, was scraped using the *nflscrapR* R package [29]. Though there are around 100 features for each play in this dataset, the data provides limited feature dimensionality for this use case. Many of the features provided cannot be directly used for prediction because the information is not available to coaches before a play occurs. Therefore, our classification problem reduces the scope of the limited data even further.

To combat these limitations, we leveraged data conditioning and data augmentation methods to increase the diversity of the data. Specifically, we filtered the play-by play data to only include pre-snap features and supplemented this data with offensive and defensive rankings per team, win and score probability features, and player grades based on Pro Football Focus analysis [8]. This allows for a wider variety of features and an increased potential for accuracy through machine learning.

## 2.1 Feature List

For each model, we used the variables shown in table 2.1. Based on the model being created, each of these data values may be normalized to be a value between 0 and 1. This ensures that the model is not placing a bias on a specific feature's weight in prediction because of the feature's magnitude relative to other features. Of the four types of models created, the logistic regression and neural network models require normalized data, since each of these methods assigns weights to the input features and updates the weights through the training process. Normalizing the features ensures that features with higher ranges do not influence prediction results more than features with smaller ranges.

Each feature can be placed into one of four categories, based on its source. Each category is explained in further detail below.

## 2.2 Pre-Snap and Post-Snap Data

In the play-by-play data from the NFL, each play includes various features, such as play type, down, and distance. Each feature can be defined as one of two main feature types: pre-snap or post-snap. Pre-snap features are defined as information that is available before the play begins, such as down, distance, and the current offensive and defensive teams. Post-snap features refer to information available after a play is completed, such as yards gained, direction of play, and the result of the play. To ensure that our models only utilize information available before a given snap, the play-by-play data is filtered to only include pre-snap features for each play. This ensures that our models are not predicting outcomes based on future information.

### 2.2.1 One-Hot Encoding Offensive and Defensive Teams

Among the pre-snap features, some of the data is categorical rather than quantitative. For example, the offensive and defensive teams of a given play are designated through a three-letter abbreviation (e.g. DAL for the Dallas Cowboys). However,

| Variable | Description | Range |
|---|---|---|
| Drive | Drive Number in Game | $1-33$ |
| Quarter | Current Quarter | $1-4$ |
| Down | Current Down | $1-4$ |
| Time Under | Time Remaining in Quarter in Minutes | $1-15$ |
| Yard Line | Yards until the End Zone | $1-99$ |
| Yards To Go | Yards Needed for First Down | $1-49$ |
| Score Difference | Difference in Score between Offense and Defense | $-59-59$ |
| No Score Prob | Probability of Not Scoring On Current Drive | $0-1$ |
| Field Goal Prob | Probability of Scoring Field Goal On Current Drive | $0-1$ |
| Safety Prob | Probability of Getting a Safety On Current Drive | $0-1$ |
| Touchdown Prob | Probability of Scoring Touchdown On Current Drive | $0-1$ |
| Win Prob | Probability of Winning the Game | $0-1$ |
| Game RTP | Run-To-Pass Ratio in Current Game | $0-1$ |
| Season RTP | Run-to-Pass Ratio in Current Season | $0-1$ |
| Month | Current Month | $8-12$ |
| Offense Ranking | Offensive Ranking in Past Season | $1-32$ |
| Defense Ranking | Defensive Ranking in Past Season | $1-32$ |
| Run Rank Offensive | Offensive Run Ranking in the NFL in Past Season | $1-32$ |
| Run Rank Defensive | Defensive Run Ranking in the NFL in Past Season | $1-32$ |
| Pass Rank Offensive | Offensive Pass Ranking in the NFL in Past Season | $1-32$ |
| Pass Rank Defensive | Defensive Pass Ranking in the NFL in Past Season | $1-32$ |
| PFF Grades | Pro Football Focus Grades per position | $0-100$ |
| Offensive Team | Hot-Encoded Binary Variable for Offensive Team | $0-1$ |
| Defensive Team | Hot-Encoded Binary Variable for Defensive Team | $0-1$ |

Table 2.1: Table of Features Used in All Models

many models do not handle categorical data. Therefore, through one-hot encoding, categorical variables such as team name can be converted into a quantitative values. We created 64 features for the current offensive team and defensive team (2 for each of the 32 NFL teams) to replace the 2 qualitative features of offensive and defensive team. Examples of these features are shown in tables 2.2 and 2.3. One-hot encoding is more effective than simply providing a numerical value (e.g. giving an identification number from 1-32) to each team since the latter could result in the model misinterpreting the values of each feature based on its magnitude relative to other teams.

| **Variable** | **Description** | **Range** |
|---|---|---|
| OffTeam_ARI | 1 if Offensive Team is Arizona Cardinals, 0 otherwise | $0-1$ |
| OffTeam_ATL | 1 if Offensive Team is Atlanta Falcons, 0 otherwise | $0-1$ |
| ... | ... | $0-1$ |

Table 2.2: One-Hot Encoded Features for Teams as Offensive Team of Play

| **Variable** | **Description** | **Range** |
|---|---|---|
| DefTeam_ARI | 1 if Defensive Team is Arizona Cardinals, 0 otherwise | $0-1$ |
| DefTeam_ATL | 1 if Defensive Team is Atlanta Falcons, 0 otherwise | $0-1$ |
| ... | ... | $0-1$ |

Table 2.3: One-Hot Encoded Features for Teams as Defensive Team of Play

## 2.3 Feature Creation

There are two key factors of offensive play-calling that are not accounted for in the *nflscrapR* play-by-play data: coach preferences and personnel. First, certain offensive coordinators may be more run-first then others. Secondly, teams will call plays based on their own personnel. A team with a Pro Bowl quarterback and a below-average running back will be more likely to throw the ball on any given play than a team

with a below-average quarterback and a Pro Bowl running back. Additionally, before games, teams will often scout their opponent to figure out the opponent's strengths and weaknesses on both sides of the ball, which plays an important role in the offensive play-calling during the game. Therefore, based on these considerations, the data should include information regarding coaching tendencies and personnel of the offensive and defensive team on a given play.

### 2.3.1 Run To Pass Ratios

A coach's general preference towards run or pass plays a key role in deciding what play he may call. Often times, many offensive coordinators maintain a relatively consistent run-to-pass ratio throughout various games. Therefore, we added a season RTP feature, which is the run-to-pass ratio of the offensive team in the current season, to the data. This feature is created through calculating the run-to-pass ratio the plays run in the season so far. It is important to look at the current season because this ensures that the offensive coordinator remains consistent through each play call in consideration.

Though an offensive coordinator's tendency to run or pass the ball as a whole may remain consistent, a team's run-pass tendencies may additionally vary based on their run-pass ratio within a game. For example, if a team has run the ball more than usual in a game, they may be more inclined to run the ball on the next play if they have had success running the ball in the game so far. Thus, we augmented the data with an in-game RTP feature, which is the run-to-pass ratio of the offensive team in the current game so far. This is calculated by finding the current ratio between run to pass for each play run in the game so far.

### 2.3.2 Offensive and Defensive Rankings

When an offensive coordinator calls a specific play, he must take into consideration his own team's strengths and weaknesses. Additionally, he must look at the opposing team's ability to defend specific types of plays. Therefore, our models take into con-

sideration not only the offense's ability to run or pass the ball, but also the defense's ability to defend against the run or pass. As such, we added six features to the data that quantify a team's offensive and defensive strengths and weaknesses.

For the offensive team, the data includes three key features: the offense's rank relative to other NFL teams' offenses in running the ball, passing the ball, and overall. Similarly, for the defensive team, the same three features are used, but in respect to the defensive rankings relative to other NFL team's defenses. This provides the model with information regarding a team's strengths and weaknesses on each side of the ball. The past season's rankings, scraped from the NFL website [1], are used for each play since, if the model used the current season's data, then it would be using future information for prediction, which should be avoided. Though not always the case, we assume a certain state of consistency of offensive and defensive success in consecutive seasons through usage of the past season's rankings.

## 2.4 Probabilities

Despite lacking access to technology in games, NFL teams have been able to leverage mathematical analysis to provide information upon a certain play's expected outcome based on a situation (e.g. down, distance, and other pre-snap features). Since teams have access to this information to make play calls, the models' use of win and scoring probabilities is warranted.

There are two types of probabilities used in our models: score probabilities and the win probability. The score probabilities are the likelihood of scoring on the current drive and the win probability is the likelihood of a winning the game at a certain point of the game. All of the probability data is included in the play-by-play data from the *nflscrapR* package [29].

### 2.4.1 Score Probability Calculation

To calculate the probability of scoring, the data splits up probabilities based on the type of score. For this use case, the data includes four score probabilities: not

scoring, scoring a field goal, scoring a touchdown, and getting a safety. These score probabilities are calculated based on six pre-snap variables of a given play, as shown in table 2.4.

| **Variable** | **Description** | **Range** |
|---|---|---|
| Down | Current Down | $1-4$ |
| Seconds | Number of Seconds Remaining in Half | $1-1800$ |
| Yard Line | Yards from Endzone | $1-99$ |
| log(YTG) | Log of Yards to Go for a First Down | $0-1.5$ |
| GTG | Indicator of if Team is in the Redzone | $0-1$ |
| UTM | Indicator of if there are 2 minutes or less remaining in the half | $0-100$ |

Table 2.4: Pre-snap Variables Used to Calculate Score Probabilities

The score probabilities are calculated through a multinomial logistic regression model [9]. The model not only considers these six factors directly, but it also takes into account the role that score difference may play into a team's offensive mindset. For example, in certain situations, when a team is leading by a large number of points at the end of a game, it will sacrifice scoring points for letting time run off the clock. This suggests that plays with high score differentials can demonstrate a different kind of relationship with the expected points scored relative to plays with lower score differentials. The model takes this into consideration by assigning a weight based off the score differential scaled between 0 and 1.

### 2.4.2 Win Probability Calculation

To calculate win probability, the data does not take the players on the field, each team's relative strengths and weaknesses, or home-field advantage into account. At the start of each game, each team starts with a 50% probability of winning the game. The win probability is calculated through a Generalized Additive Model (GAM) [11]. One of the key benefits of GAMs that makes them ideal for modeling win probability is that they allow the relationship between the explanatory and response variables to vary according to smooth, non-linear functions [29]. The model considers eight

different variables, as listed in table 2.5, and calculates win probability through the following equation, in which $s$ is a smoothing function:

$$WP = \frac{P(Win)}{P(Loss)} = s(E[s]) + s(s_h) * h + s(E[\frac{s}{s_g + 1}]) + h * u * t_o + h * u * t_d$$

| Variable | Description |
|---|---|
| $E[s]$ | Expected Score Differential |
| $s_g$ | Seconds Remaining in Game |
| $s_h$ | Seconds Remaining in Half |
| $h$ | Current Half of the Game (1st, 2nd, or overtime) |
| $u$ | Binary Indicator of whether Time remaining in Half is under two minutes |
| $t_o$ | Timeouts Remaining for the Offense |
| $t_d$ | Timeouts Remaining for the Defense |
| $E[\frac{s}{s_g+1}]$ | Expected score time ratio |

Table 2.5: Pre-snap Variables Used to Calculate Win Probabilities

As we can see, the win probability model takes into consideration the score probabilities mentioned earlier and other easily accessible pre-snap variables. Figure 2-1 shows an example of how the win probability changes over the course of a game.

## 2.5 Pro Football Focus Player Grades

One of the key factors in play-calling is each team's personnel. Specifically, an offense must consider the caliber of its own players and the defensive players before deciding on which play to run. However, in the play-by-play data, information on the players who are on the field on a given play is limited. For each play, the NFL does not publicly provide information about which players are on the field, a team's offensive or defensive formation, or the pre-snap and post-snap locations of players on the field. This is especially true for positions other than QB, RB, WR, and TE, since NFL data only contains information upon which players have touched the ball on a given play (e.g. the passer, rusher, or receiver). For offensive linemen, very little information is

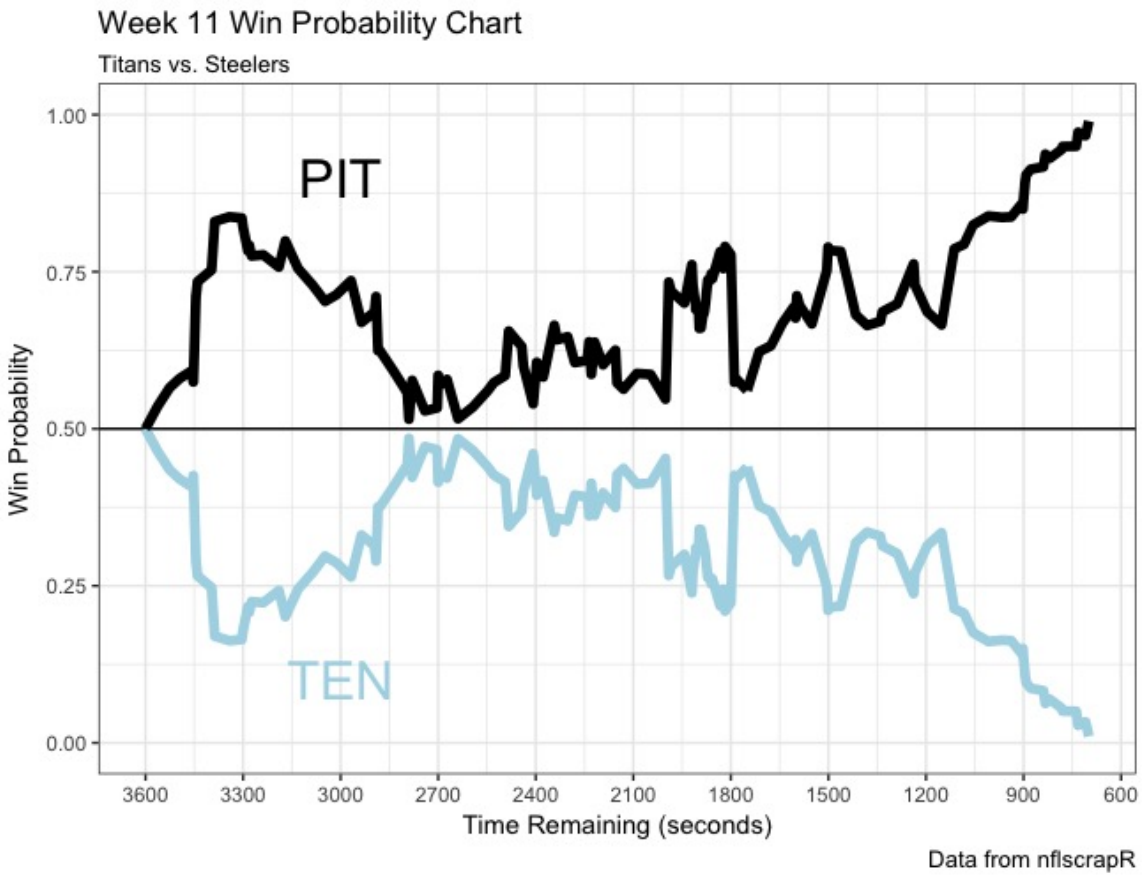
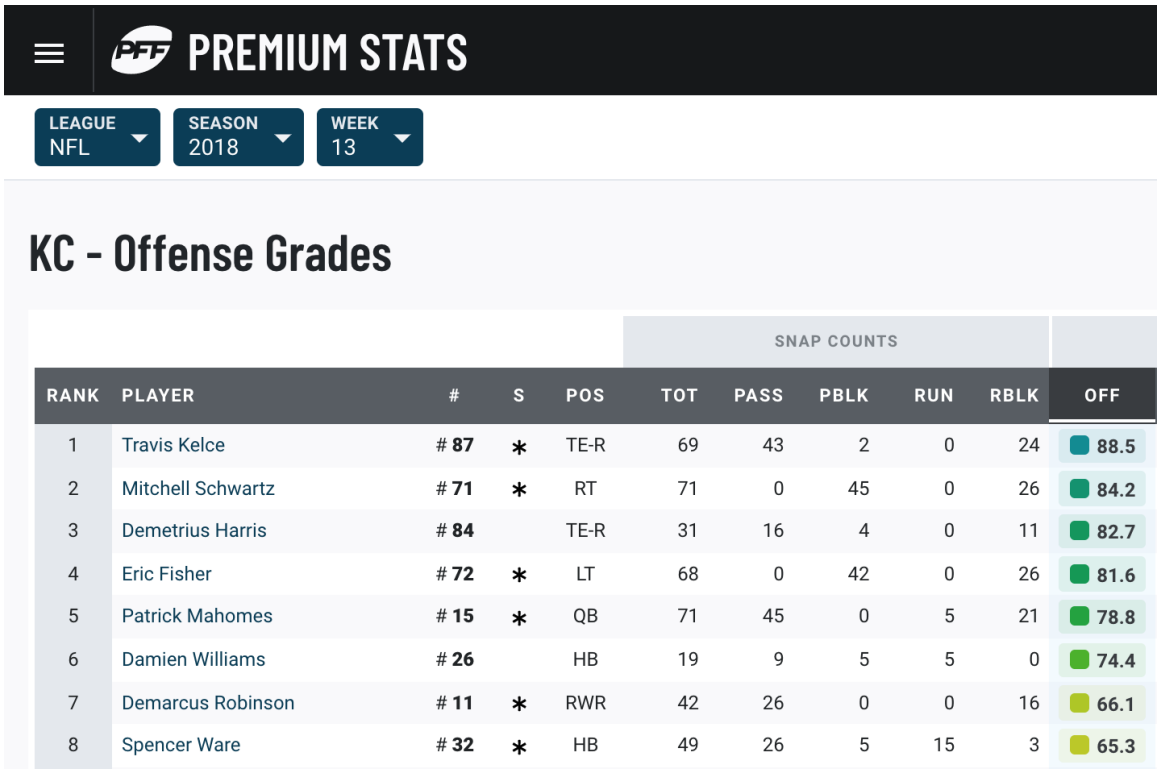



Figure 2-1: Example of Win Probability over the course of a Game

available to statistically compare players, as offensive linemen typically only touch the football on broken plays. For defensive players, the NFL only provides information about which players were directly involved in a play (such as the defensive player who was guarding the receiver targeted on a given play). This makes it difficult to obtain adequate comparisons of player values for each team on a given play, let alone across-position comparisons.

Therefore, for information on personnel, we utilized Pro Football Focus grades of a player's performance in past games [8]. The assumption is that, given a player's success in recent games, he is more likely to remain consistent in his performance in the current game. This provides a metric to quantify player performance. Pro Football Focus (PFF) uses a proprietary grading algorithm to rate players on a scale from $0-100$. Therefore, through purchasing access to these player grades and scraping




| | | | | | SNAP COUNTS | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| RANK | PLAYER | # | S | POS | TOT | PASS | PBLK | RUN | RBLK | OFF |
| 1 | Travis Kelce | # 87 | * | TE-R | 69 | 43 | 2 | 0 | 24 | 88.5 |
| 2 | Mitchell Schwartz | # 71 | * | RT | 71 | 0 | 45 | 0 | 26 | 84.2 |
| 3 | Demetrius Harris | # 84 | | TE-R | 31 | 16 | 4 | 0 | 11 | 82.7 |
| 4 | Eric Fisher | # 72 | * | LT | 68 | 0 | 42 | 0 | 26 | 81.6 |
| 5 | Patrick Mahomes | # 15 | * | QB | 71 | 45 | 0 | 5 | 21 | 78.8 |
| 6 | Damien Williams | # 26 | | HB | 19 | 9 | 5 | 5 | 0 | 74.4 |
| 7 | Demarcus Robinson | # 11 | * | RWR | 42 | 26 | 0 | 0 | 16 | 66.1 |
| 8 | Spencer Ware | # 32 | * | HB | 49 | 26 | 5 | 15 | 3 | 65.3 |

Figure 2-2: Example of PFF Offensive Grades Database

roster information from each game, we created features to quantify the quality of a team's current personnel for various positions. This feature takes into account key players on the team, not the current players on the field, since the latter is not publicly available data. For each position, we set the feature value to a weighted average of the overall PFF grades from the last game of the top players per position for each team. We used the average number of players on the field for each position to calculate this value. For example, for wide receivers, since many teams use four main wide receivers during a game, we set the WR Grade feature value to a weighted average of the grades of the top four wide receivers on the offensive team based on their respective number of targets. These PFF grades provide the model with a metric to evaluate the offensive personnel in a given game.

# Chapter 3

# Models

Machine learning encompasses a wide variety of algorithms and model structures. Each method approaches its inputs differently and can provide varying insights based on the purpose of the model. The broad scope and flexibility of machine learning methods allow these methods to be useful in a wide variety of problem spaces and applications.

When looking at the problem of predicting run vs. pass on a given play, various types of methods can be leveraged. However, since each method is optimized for specific types of problems and outputs different types of insights, it is essential to consider a variety of model architectures to be able to extract diverse insights. Given the binary classification task at hand and the data at our disposal, we experimented with four main model types: logistic regression, neural network, decision tree, and random forest models. Each model comes with a different value proposition in the quest for understanding play-calling tendencies.

Before deciding which methods to use for our classification task, we first set forth two main priorities regarding the model outputs: accuracy of the model and the weights of each variable in the model. First, we want our models to be as accurate as possible. A key aspect of the value proposition of machine learning models is high prediction accuracy. It is also important that we understand which factors play the biggest roles in predicting future play types in each model. Therefore, we want to create models that will be able to provide information upon the value of a specific

variable in a given model. We keep these two priorities in mind when choosing and evaluating our models.

In this section, we will look at four types of models. We will examine the creation, the value proposition, and the potential drawbacks of each model. This will serve as the foundation for the insights we glean from the results of each model in the following chapter.

## 3.1 Logistic Regression

Logistic regression is one of the most commonly used models for two-class classification. Given its simplicity and applicability, it will serve as starting point for our approach to the binary classification problem as a whole.

### 3.1.1 Model Description

Logistic regression is a statistical method used to predict multiple classes. In this case, we will use binary logistic regression to predict run vs. pass given our data set. Logistic regression methods output a probability to classify a test sample [5]. We attempt to predict run plays with a classification value of 0 and pass plays with a classification value of 1.

$$x' = \frac{x - \mu}{\sigma} \tag{3.1}$$

First, the logistic regression model centers the data to have zero mean and unit standard error. This is achieved through equation 3.1, where $\mu$ is the mean of the data and $\sigma$ is the standard deviation of the data. We find the values for $\mu$ and $\sigma$ for the training data, and then transform the training data and the test data using these values. This ensures that our training and test data set are centered.

Our next step is finding the linear separability of the data space. In binary logistic regression, the assumption is that the input space can be split up into two regions, one for each classification. Therefore, the logistic regression model's purpose is to find the boundary, or the linear discriminant, which separates the space.

In a linear regression model, the two classes of data points are treated as numbers (0 and 1) and the model fits the best hyperplane that minimizes the distances between the points and the hyperplane. For example, if we only had a single feature, the hyperplane created would be a line. In linear regression, the function used to create the hyperplane is shown in equation 3.2, where $y$ is the classification probability, $x_i$ is an input feature of a sample in the data set, and $\beta_i$ is the weight placed on on the input feature $x_i$ by the logistic regression model.

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + ... + \beta_n x_n \tag{3.2}$$

However, since the predicted outcome is not a probability, but a linear interpolation between points, there is no meaningful threshold at which you can distinguish one class from the other. The logistic regression model solves this through using the logistic function shown in equation 3.3 to squeeze the output of the linear equation 3.2 between 0 and 1.

$$logReg(n) = \frac{1}{1 + e^n} = \frac{1}{1 + e^{\beta_0 + \beta_1 x_1 + \beta_2 x_2 + ... + \beta_n x_n}} \tag{3.3}$$

This equation, once formulated by the logistic regression model based upon the training data points, is applied to each test data sample for classification [7].

### Parameter Choices

As in any machine learning model, we must tune parameters based on the desired output and the classification task at hand. For logistic regression, the key parameter to consider is the type of regularization method to use. Regularization decreases complexity within a model to ensure that the model does not overfit. Regularization works on the assumption that smaller weights generate simpler models and thus help avoid overfitting. To ensure we take into account the input variables, we penalize all the weights by making them small. This also makes the model simpler and less prone to overfitting. There are two main types: L1 and L2 regularization. Each regularization method adds to the loss function of the model, which is optimized to

be as low as possible during the training of the model. This optimization extracts the optimal weights per variable for accurate prediction. In L1 regularization, also known as Lasso regression, the absolute value of the magnitude of each variable's weight is added to the loss as a "penalty term." This is shown in equation 3.4. In L2 regularization, also known as Ridge regression, the squared magnitude of each variable's weight is added to the loss as a "penalty term." This is shown in equation 3.5. The key difference between L1 and L2 regularization is that L1 regularization shrinks the weights of the less important features to 0. Therefore, given the wide variety of features in our dataset and the desire for feature selection, our logistic regression model makes use of L1 regularization [22].

$$\text{L1 Loss+} = \lambda \sum_{i=1}^{n} |\beta_i| \tag{3.4}$$

$$\text{L2 Loss+} = \lambda \sum_{i=1}^{n} \beta_i^2 \tag{3.5}$$

As shown, each sum is multiplied by a $\lambda$ value, which dictates how much the penalty term impacts the loss value. If the $\lambda$ value is too high, then too much weight will be added to the penalty term, resulting in under-fitting. However, if the $\lambda$ value is too small, then over-fitting may occur.

### 3.1.2 Advantages: Simple and Insightful

Logistic regression models provide two key advantages: simplicity of use and feature insights. With the use of pre-existing libraries such as *sklearn* [24], logistic regression can be easily implemented with a given training and test data split. Secondly, creating a logistic regression model involves finding weights for specific input features. Therefore, given a logistic regression model, it becomes easy to extract information about the importance of features in the prediction model. This serves as an important advantage in our use case, since understanding why a model predicts a run-pass play in a given situation is arguably more important than being able to build a model with high accuracy. These insights on play-calling tendencies might be utilized by NFL

teams to understand how to make their own offenses less predictable and how to gain a defensive advantage upon their opponents in the league.

### 3.1.3 Disdvantages: Lack of Complexity and Overfitting

One of the key disadvantages of a logistic regression model is that it necessitates linear separability of the data for best results. Considering that a logistic regression models aims to split a data region linearly, it tends to be inaccurate with data that requires more complex separability. Therefore, if there are nonlinear correlations between features in the dataset, the logistic regression method would have difficulty finding accurate weight parameters. Another disadvantage of logistic regression is its reliance upon a proper presentation of our data. This means that logistic regression is not a useful tool unless we have already identified all the important independent variables. Lastly, logistic regression is a model known to be vulnerable to over-fitting with a large number of features as in our case.

## 3.2 Neural Network

Since one of the main priorities of our model creation is high accuracy given variables with potentially non-linear correlations, we experimented with the use of neural network models, which provide high potential for accuracy.

### 3.2.1 Model Description

Through the use of multiple interconnected layers of neurons, neural networks are able to consume large amounts of data and propagate these sample points forward for classification. Through forward and back propagation, neural networks can update the weights of neurons to provide high accuracy for a test set of data points. We use a Fully-Connected Convolutional Neural Network model, which contains an input layer, multiple hidden layers, and an output layer. Each neuron is connected to every neuron in the previous layer and the following layer [26].

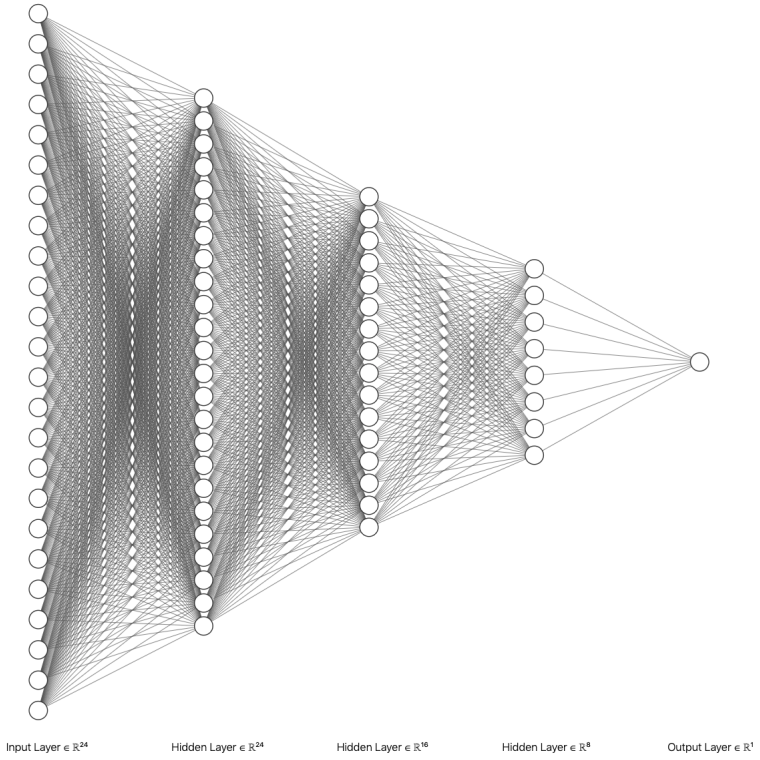



Figure 3-1: Neural Network Model Architecture

An example of one of our convolutional neural network models is shown in figure 3-1. This figure demonstrates a neural network with 5 layers: an input layer, three hidden layers, and an output layer. Each edge, which connects two neurons as shown by the lines, is given a weight, which is used in forward propagation to get the classification output of a given data point. Back propagation in the neural network model will update the weights of each edge through minimizing our loss function [26], which is binary cross entropy.

$$CE = -\sum_{i=1}^{C} t_i log(s_i) = -\sum_{i=1}^{2} t_i log(s_i) \tag{3.6}$$

The binary cross entropy loss function is shown in equation 3.6, where $t_i$ is the ground truth classification of the data point and $s_i$ is the neural network model's output for each class $i$ in $C$. Since we only have two classes, we can simplify the loss equation through setting $C = 2$.

**Parameter Choices**

Each layer is also given an activation function to filter the sum of the inputs into each neuron within the layer. In our models, we used the sigmoid activation function for all layers [25]. The sigmoid function restricts the value of its input to the range of 0 to 1, and pushes values towards one of the two ends of the range. The sigmoid equation is shown in equation 3.7, where $x$ is the input value and $y$ is the output.

$$y = \frac{1}{1 + e^{-x}} \tag{3.7}$$

We also experimented with the relu activation function [23], with limited success. This may have been a product of the increased separation of outputs created by sigmoid activation relative to the relu activation function, which only considers positive outputs. Figure 3-2 shows an example of the difference between sigmoid and relu functions.

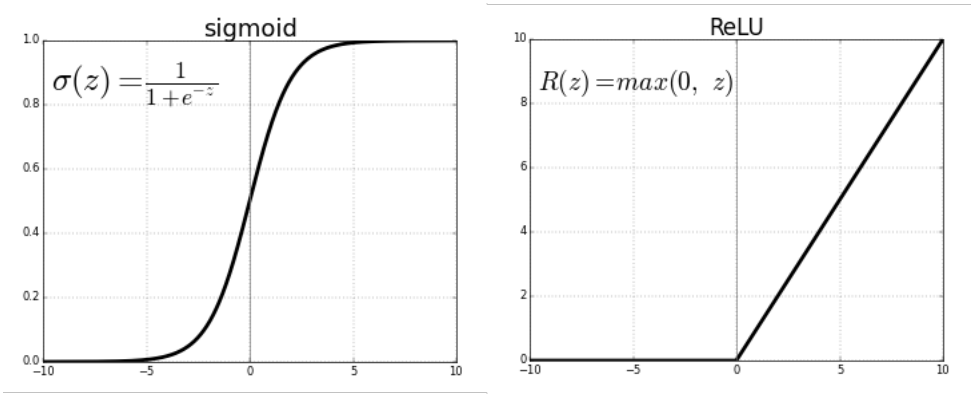



Figure 3-2: Sigmoid vs. Relu Activation functions

### 3.2.2 Advantages: Complexity and Accuracy

Neural networks serves as one of the most popular methods in machine learning for three main reasons. Neural networks effectively evaluate non-linear relationships within data, while other models may require data conditioning or data transformations via kernels. Therefore, neural network models provide an easy out-of-the-box solution for complex data sets. Additionally, since they have strong predictive capa-

bilities and high potential complexity, neural network models are also capable of high classification accuracy. Lastly, neural network models have a high tolerance to noisy data, as well as an ability to classify patterns on which they have not been trained.

### 3.2.3 Disadvantages: Lack of Insights

Despite high accuracy for neural network models in large-scale classification, one of the main drawbacks of neural networks for modeling is the inability to understand the main factors that the neural network model uses for prediction. Specifically, neural network models are "black boxes" in the sense that it is difficult to obtain insights regarding the factors which were the most important in making a given prediction. In the National Football League, since advanced technology and model software cannot be used during games, the factors for prediction are often more important than the prediction accuracy of a given model. An NFL team will most likely care less about the ability to predict a team's run-pass tendencies using a model and more about the key factors that drive run-pass play-calling decisions.

## 3.3 Decision Tree

NFL coaches care most about which factors play the biggest role in play-calling in certain situations. Not only are the features themselves important, but coaches are keen upon learning at what point does a team become more likely to run vs. pass. For example, if the current yards to go for a first down is an important feature, organizations may be looking to understand the upper limit of yards to go until an opposing coach may pass the ball instead of run the ball. To address this, we implemented decision tree models.

### 3.3.1 Model Description

Decision trees split up an input space based upon the features which provide the most accurate split based on the training data. The representation of the decision

tree model is a rooted tree [4]. For example, in a binary decision tree, each node can have zero, one or two child nodes. A node represents a test on an input variable (x), specifically as a split point on that variable. Each branch in the tree represents an outcome of the test. The leaf nodes (also called terminal nodes) of the tree contain an output variable (y) which is used to make a prediction. Once created, a tree can be navigated with a new row of data following each branch with the splits until a final prediction is made.

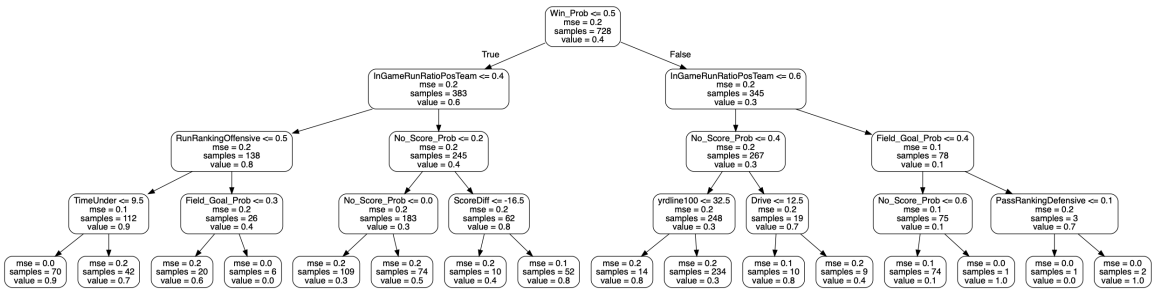



Figure 3-3: Example of a Decision Tree

Our input splitting is greedy and exhaustive; all input spaces and split points are evaluated and selected in a greedy manner, specifically through a cost function. This function is minimized relative to the sum squared error across all training samples. The Gini index is the cost function used to evaluate splits in our dataset. The Gini Index for a binary target variable is calculated through equation 3.8, where $P_t$ is the play-type of point $t$ for a split point of an input feature. The Gini index provides a score regarding the quality of the split for a given feature (e.g. yard line) and a given target variable, which in this case is the play type [30]. A perfect separation results in a Gini Score of 0, while the worst case split is 0.5.

$$1 - \sum_{t=0}^{t+1} P_t^2 \tag{3.8}$$

After calculating the Gini scores for each feature in relation to the target variable, we split the dataset for each given feature and evaluate the cost of the split. The best split (or the one with the smallest Gini score) is used as a node in the decision tree. After finding an optimal split point and setting it as a node in the decision tree, we

split the data accordingly and continue to recurse to find additional nodes until we reach our maximum depth. To avoid over-fitting, we set a maximum depth for the decision tree. This helps reduce undesired granularity in the decision tree based on the training data.

### 3.3.2 Parameter Choices

In decision trees, the key parameter choice is the maximum depth of the tree. The model must find balance between under-fitting and over-fitting through setting a maximum depth of the decision tree. This parameter provides additional insight upon how complex NFL play-calling is as a whole, for if a decision tree can achieve high prediction accuracy with a relatively low maximum depth, then this suggest that a team's play-calling process can be a broken down into a couple of features. Overall, through experimentation, we found that the most effective maximum depth ranged from 3-5 layers.

### 3.3.3 Advantages: Visualization and Real-Game Applications

The value proposition of decision tree models lies in their ease of visualization and potential for real-game application. As it relates to visualization, decision trees are simple to understand. As shown in figure 3-3, the decision tree contains a simplistic tree structure that can be traversed with the human eye given a data point. Additionally, since decision trees can be visualized and understood easily, there is high potential for usage of decision trees in games by coaches. Specifically, decision trees provide a tangible breakdown of play-calling tendencies which can be followed and implemented by NFL coaches during a game.

### 3.3.4 Disadvantages: Accuracy Ceiling and Data Limitations

Two key limitations arise with decision trees. First, decision trees often contain an upper bound for accuracy. Decision trees are often prone to over-fitting based on the depth of the tree. This is due to the amount of specificity necessary to understand

a smaller sample of events as the decision tree moves further down levels of the tree. As the decision tree algorithm works down the levels of the tree, the sample of data being considered becomes so small that dictating decisions for classification based upon small amounts of data leads to over-fitting and decreased classification accuracy. Though limiting decision tree depth could reduce this error, this restricts the strength of the prediction capabilities of decision tree, resulting in a ceiling for overall accuracy of the decision tree [4]. Additionally, decision tree creation becomes time-consuming given a large quantity of data, suggesting that decision trees are best-suited for specific situations on the football field, in which the quantity of training data is smaller. However, this often results in lower classification accuracy as a whole.

## 3.4 Random Forest

Random forests provide the simplicity of decision trees combined with high accuracy. A random forest is a collection of decision trees in which decision tree classifications are aggregated into one final result.

### 3.4.1 Model Description

Random forest is an ensemble machine learning approach. The main principle behind the ensemble approach is the aggregation of multiple weak learners to create a strong learner for classification [18]. In the case of random forests, the various weak learners are decision trees. The random forest model creation begins by splitting up the input data into multiple random subsets. Each random subset of the data often contain around 66% of the total data. For each random subset, a decision tree is created. The process of creating a decision tree is outlined in section 3.3.

However, in the random forest model, when decision trees are created, not all variables are taken into consideration for a specific node. At each level, only a random subset of predictor variables are chosen. Within this subset, the predictor variable with the smallest Gini score of the variables selected will be chosen for the node. This random selection of a subset of predictor variables is done to improve computational

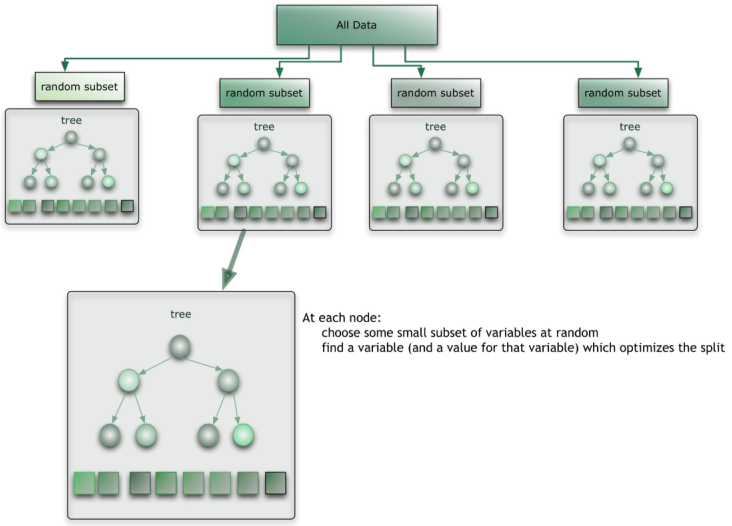



Figure 3-4: Random Forest Architecture

efficiency, to reduce over-fitting, and to increase the variety of decision trees created. Through this, a random forest can create thousands of uncorrelated decision trees quickly.

Once these decision trees are created through the training data set, we iterate through each decision tree created with each sample point of the test data set. For each test data sample, we run the test point through each decision tree and aggregate the results through an average of the classifications of each tree. This provides a probabilistic value of the expected classification of a given play.

The higher the correlation between the decision trees, the higher the risk of over-fitting. Thus, the highest performing models will ensure that the trees are as uncorrelated as possible. Since the play-by-play data set contains a considerable number of features, the expectation is that many of the trees will include different variables based on the eligible predictor variable set for each random subset of data.

### 3.4.2 Parameter Choices

Since random forests are an aggregation of multiple decision trees, the maximum depth of each decision tree must be specified, as mentioned in section 3.3. Addition-

ally, the number of decision trees created must be specified as well. As mentioned above, with higher correlation between decision trees comes a higher error rate. Therefore, if the number of decision trees created within the random forest is too high, this may result in over-fitting for the random forest. On the other hand, without enough decision trees, the ceiling of accuracy for the random forest is reduced, resulting in under-fitting of the model. Thus, we must find the correct balance for the parameter of the number of decision trees within the random forest. We found that the ideal number of decision trees per random forest ranged from 100-150.

# Chapter 4

# Results

When creating the models for our binary classification problem, we wanted to look at two different types of situations. First, we wanted to see if we could create models for specific situations of interest based on yard line. For example, we explored run-pass tendencies of when a team's offense is faced with a 1st down between the 30 yard lines. Secondly, we were interested to see if a model could accurately predict run vs. pass for any given play on the field. If such a model could be created, it would provide key insights into general play-calling tendencies for a team. We evaluated the situations listed in table 4.1 in an attempt to understand the situations in which teams are the most predictable.

| **Situation** | **Down** | **Yards to End Zone** | **Description** |
|---|---|---|---|
| 1-3:100-0 | 1-3 | 100 - 0 | All 1st to 3rd Down Plays |
| 1-3:90-80 | 1-3 | 90 - 80 | 1st - 3rd Down Plays from Team's Own 10 yard line to Own 20 yard line |
| 1:70-30 | 1 | 70 - 30 | 1st Down Plays From Team's Own 30 yard line to Opponent's 30 Yard Line |
| 1-3:70-30 | 1-3 | 70 - 30 | 1st - 3rd Down Plays From Team's Own 30 yard line to Opponent's 30 Yard Line |
| 1-3:20-10 | 1-3 | 20 - 10 | 1st - 3rd Down Plays from Opponent's 20 yard line to 10 yard line |

Table 4.1: Specific Game Situations Analyzed in Testing

### 4.0.1 Baseline

When measuring accuracy of a prediction, our metric for success is test accuracy of the model relative to the run-pass probability of the situation. For example, of all plays run between an offense's own 10 and 20 yard lines from 2009 - 2018, 56.7% were pass plays. Therefore, if we were to guess pass as the play type for each of these plays, we would be 56.7% accurate. Each situation's run to pass play ratio for all plays from 2009-2018 is shown in table 4.2. We will refer to the difference between a model's accuracy and this play ratio as relative accuracy.

| **Situation** | **Play Ratio** | **Play Type Majority** | **Play Count** |
|---|---|---|---|
| 1-3:100-0 | 56.84 | Run | 244,194 |
| 1-3:90-80 | 56.69% | Pass | 29,530 |
| 1:70-30 | 50.60% | Run | 58,040 |
| 1-3:70-30 | 54.37% | Pass | 121,480 |
| 1-3:20-10 | 55.54% | Pass | 17,350 |

Table 4.2: Run vs. Pass Percentage per Game Situation

When choosing situations to evaluate, we ensured that each situation had a reasonably similar run vs. pass play count in the situation to ensure that our models would provide value through accurate prediction.

### 4.0.2 Training and Testing Data Splits

During model creation, we used the date of the play to split between training and testing data. This ensured that our testing data would occur chronologically after our training data and that we were not predicting the play type of a play with future data. For each model, we experimented with two types of training vs. testing time data splits: year-based and month-based. For the year-based data split, we tested our models by using all plays from the last year of our data set. With the year-based data split, we achieved the best results through only keeping the 2018 season in our test data, which is approximately a 90/10 split of the data. We only used the 2018 season because it is the last season available in our data and since it provided us a

larger amount of training data. For the month-based data split, we tested our models by using all plays from the last months of each regular season. We noticed the best results by only using the last month of each regular season (December) as our test data set, which is approximately a 80/20 split of the data. Similar to the year-based data split, we only used the month of December as our test data set to increase the training data used.

Between the two data splits, the month-based data split was on average around 3% more accurate. This may be caused by coaching and personnel changes between seasons, which causes fluctuations in play-calling over time. Over the course of a season, coaching and player personnel of a team remain more consistent relative to over multiple seasons.

## 4.1 Model Accuracy

For each situation, we created general models for the entire NFL and team-specific models. The general models use data that include the one-hot encoded categorical team variables for the offensive and defensive teams, as discussed in section 2.2.1, while the team-specific models' data does not include the offensive one-hot encoded team features.

### 4.1.1 General Models

Table 4.3 lists the test accuracy of each model for each situation, and 4.4 demonstrates the results of the best models relative to the baseline. Each of these results used the month-based data split, with plays in December of each season used as our test data set. The last row of table 4.3 is the weighted average of the accuracy of each model based on the play counts of each situation. Looking at each specific situation, the neural network method was the most effective, for it had the highest accuracy for three of the five situations and the highest weighted accuracy overall. The neural network model predicts the situations with two of the three highest play counts more accurately than the other models. Additionally, the random forest model was slightly

less accurate than the neural network model in general, yet achieved a 5% greater accuracy in one situation (1-3:90-80). The random forest achieved the highest accuracy compared to the other models in the situations with the two lowest play counts. Situations with less training data may not be as diverse, which may result in higher potential accuracy for random forest models compared to neural network models.

Though these results are promising in their high relative accuracy, both neural network and random forest models are considered "black-box" models since the factors that drive their predictions are often difficult to extract. We will explore the factors important to play-calling prediction in section 4.2.1.

| **Situation** | **Log. Regression** | **Neural Net** | **Dec. Tree** | **Random Forest** |
|---|---|---|---|---|
| 1-3:100-0 | 71.1% | 74.9% | 71.2% | 74.3% |
| 1-3:90-80 | 74.6% | 70.5% | 74.5% | 75.8% |
| 1:70-30 | 70.3% | 72.8% | 70.9% | 71.6% |
| 1-3:70-30 | 70.2% | 79.7% | 71.0% | 78.5% |
| 1-3:20-10 | 75.4% | 76.4% | 75.8% | 77.6% |
| **Weighted Acc.** | **71.1**% | **75.7**% | **71.5**% | **75.3**% |

Table 4.3: Test Accuracy of All Models for each Situation

| **Situation** | **Test Accuracy** | **Baseline** | **Difference** | **Model** |
|---|---|---|---|---|
| 1-3:100-0 | 74.9% | 56.84 | 18% | Random Forest |
| 1-3:90-80 | 75.8% | 56.69% | 18% | Random Forest |
| 1:70-30 | 72.8% | 50.60% | 22% | Neural Network |
| 1-3:70-30 | 79.7% | 54.37% | 25% | Neural Network |
| 1-3:20-10 | 77.6% | 55.54% | 20% | Logistic Regression |

Table 4.4: Results of Best Models for each Situation

### 4.1.2 Team-Specific Models

For the team-specific models, model accuracy varied considerably per team. Table 4.5 shows the most predictable teams per situation and table 4.6 shows the least predictable teams per situation. Unlike our general models, we reached the highest

relative accuracy for a team-specific model for plays run between an opponent's 20 and 10 yard line. This may be because of the limited quantity and diversity of data used for team-specific models.

| Situation | Team | Accuracy | Baseline | Difference | Model |
|---|---|---|---|---|---|
| 1-3:100-0 | New York Jets | 75.4% | 51.14 | 24% | Neural Net |
| 1-3:90-80 | Cincinnati Bengals | 78.3% | 51.69% | 26% | Dec. Tree |
| 1:70-30 | Detroit Lions | 83.74% | 62.41% | 21% | Neural Net |
| 1-3:70-30 | Atlanta Falcons | 85.9% | 61.17% | 24% | Neural Net |
| **1-3:20-10** | **Tennessee Titans** | **82.95%** | **56.05%** | **27%** | **Log. Reg** |

Table 4.5: Most Predictable Teams for Each Situation

| Situation | Team | Accuracy | Baseline | Difference |
|---|---|---|---|---|
| **1-3:100-0** | **New Orleans Saints** | **69.4%** | **62.3%** | **7%** |
| 1-3:90-80 | Philadelphia Eagles | 68.5% | 58.69% | 10% |
| 1:70-30 | New England Patriots | 61.74% | 50.1% | 12% |
| 1-3:70-30 | Seattle Seahawks | 65.13% | 53.6% | 11% |
| 1-3:20-10 | Dallas Cowboys | 66.85% | 59.05% | 8% |

Table 4.6: Least Predictable Teams for each Situation

Additionally, we see higher accuracy for the most predictable teams relative to the general models. This is expected, since play-calling is more likely to be consistent when looking at only one team's plays relative to the entire league's plays. Models are much more likely to capture patterns within team-specific data sets given the consistency of play-calling within a single team.

## 4.2 Model Insights

### 4.2.1 Key Factors

Since two of our models (logistic regression and decision trees) are easy to interpret, we can look at the learned weights or tree respectively to provide us with some

intuition upon play-calling tendencies. With general and team-specific data, the logistic regression and decision tree models did best for 1st to 3rd downs between the 20 and 10 yard lines (1-3:20-10). In this situation, the logistic regression model had a max accuracy of 83% and the decision tree had a max accuracy of 80%, both for the Tennessee Titans. The average relative accuracy was 19% for the logistic regression model and 17% for the decision tree.

### Logistic Regression Model

First, let's look at the run and pass factors of the logistic regression model in each situation. After comparing the best logistic regression models for each situation, both general and team-specific, the top factors are shown in table 4.7.

| Situation | Top Run Factors | Top Pass Factors |
|---|---|---|
| 1-3:100-0 | Touchdown Prob, RB Grade | Yards To Go, QB Grade |
| 1-3:90-80 | Game RTP, Touchdown Prob | Yards To Go, Down |
| 1:70-30 | Game RTP, Touchdown Prob | Field Goal Prob, Yards To Go |
| 1-3:70-30 | Touchdown Prob, RB Grade | QB Grade, Yards To Go |
| 1-3:20-10 | Touchdown Prob, Game RTP | Field Goal Prob, Yards To Go |

Table 4.7: Top Positively Weighted Factors for each Situation - Logistic Regression

Key run factors include the offensive team's in game run-to-pass ratio, its probability of scoring a touchdown, and the quality of its running back. These factors intuitively make sense. A higher in-game run-to-pass ratio suggests that a team is having success or simply prefers running the ball in the game so far, making it more inclined to continue to run the ball in future plays. A high touchdown probability suggests that a team is in scoring position, meaning that they are closer to the goal line and more likely to run the ball in. Lastly, a high RB grade means the team has a good running back, which provides incentive for the coach to run the ball.

As it relates to top pass factors, the four key features were yards to go for a 1st down, the current down, the QB grade, and the field goal probability. Down and distance as pass factors are expected. Given a higher down and distance, a team

is more likely to pass the ball, for it becomes more necessary for the team to take risks to be able to get a 1st down. In later downs and longer yardage situations, the offense will be more likely to pass the ball to get the yards necessary for the 1st down. Similarly, when a team has a better quarterback, they will be more inclined to throw the ball on a given play. Lastly, when a team has a higher chance of scoring a field goal, they may be more willing to attempt to get into touchdown-scoring position, since at the least, they should have an opportunity to score through a field goal attempt even if they do not make substantial forward progress. Though an interception or, in certain situations, a sack could move a team out of scoring position, in most cases, a team is willing to take a higher risk for the reward of getting into touchdown-scoring territory.

### Decision Tree

For decision trees, let us take a look at the key factors of the top performing decision trees for general and team-specific data.

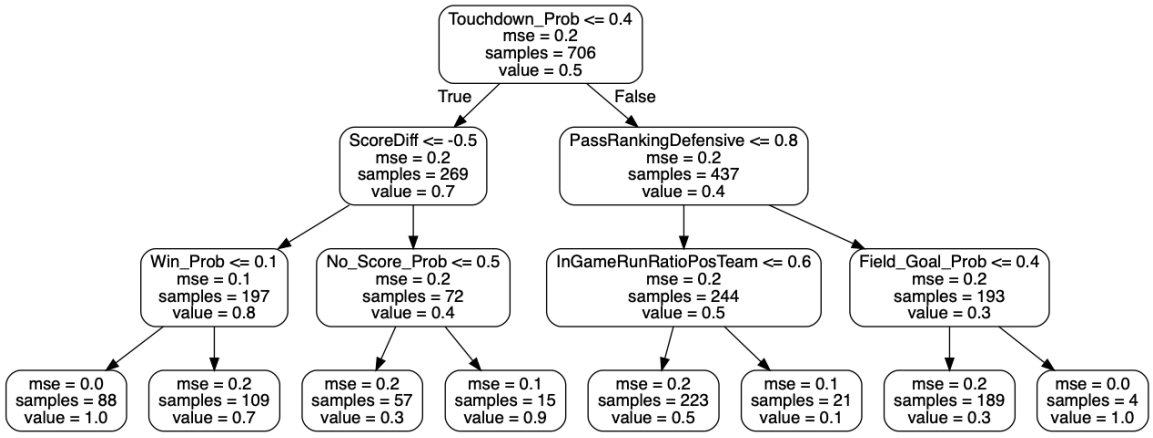



Figure 4-1: Decision Tree for 1st downs between the 30 yard lines

The best decision tree for general data is shown in figure 4-1. This tree is for 1st to 3rd down between the 30 yard lines. The best decision tree for a specific team is shown in figure 4-2, which is for the Tennessee Titans for 1st to 3rd downs between the opponent's 20-10 yard lines. In these decision trees, we see that the

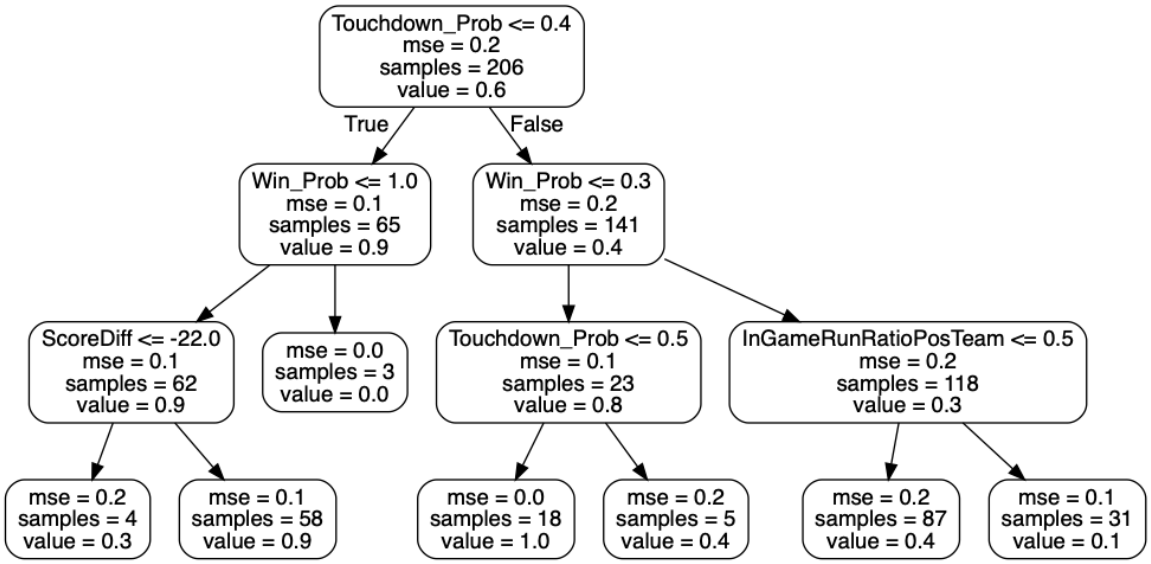



Figure 4-2: Decision Tree for the Tennessee Titans (1-3:20-10)

key factors for prediction are score probabilities (e.g touchdown, field goal, no score), in-game run-to-pass ratio, win probability, and the current score differential. It is interesting to note that, though we'd expect score differential and run-to-pass ratio to be key factors, the paucity of features such as down and distance in the prediction process is surprising. However, if we consider the inputs that go into the win and score probability calculations, as shown in section 2.4, we see that these probabilities amalgamate these features into a specific value. Therefore, despite the absence of features such as down, distance, and time left in the half as top factors of the decision tree and the logistic regression models, these features implicitly play a key role in the classification methods of the decision tree model.

### 4.2.2 Situational Predictability

Next, we looked at how predictable teams are at different times of the game. We looked at the accuracy of our best models on each quarter and each down, as shown in figures 4-3 and 4-4.

In figure 4-3, we see that teams tend to become more predictable during the 2nd or 4th quarter. This may be because the ends of the 2nd and 4th quarters have outsize effects on the game outcome. Specifically, we see that 4th quarter accuracy is

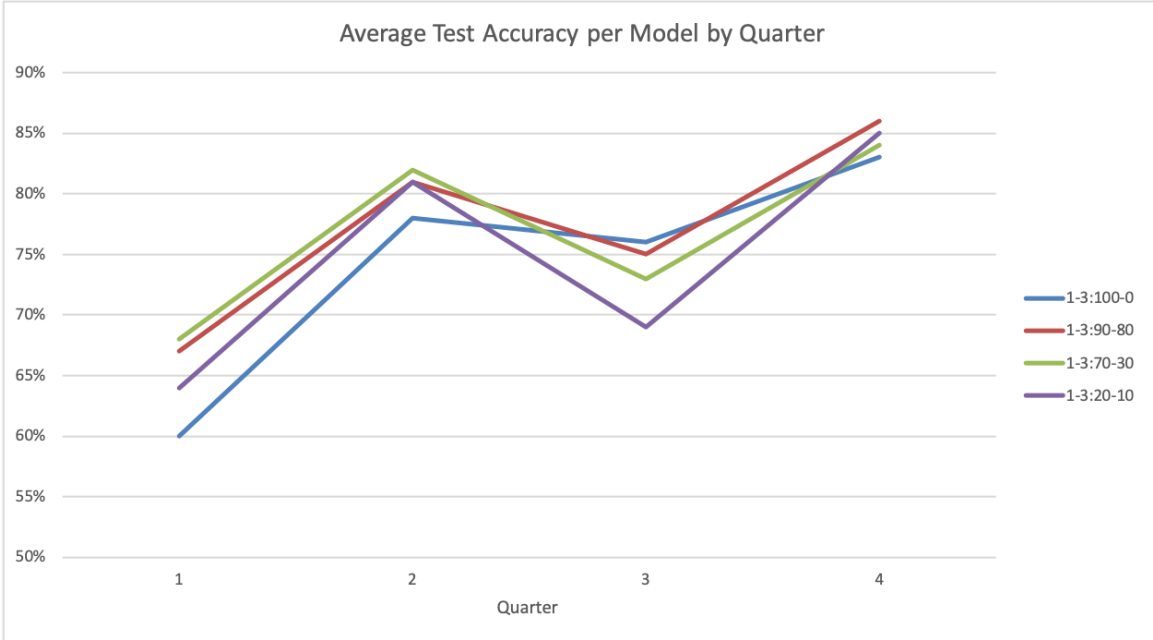



Figure 4-3: Model Accuracy per Quarter

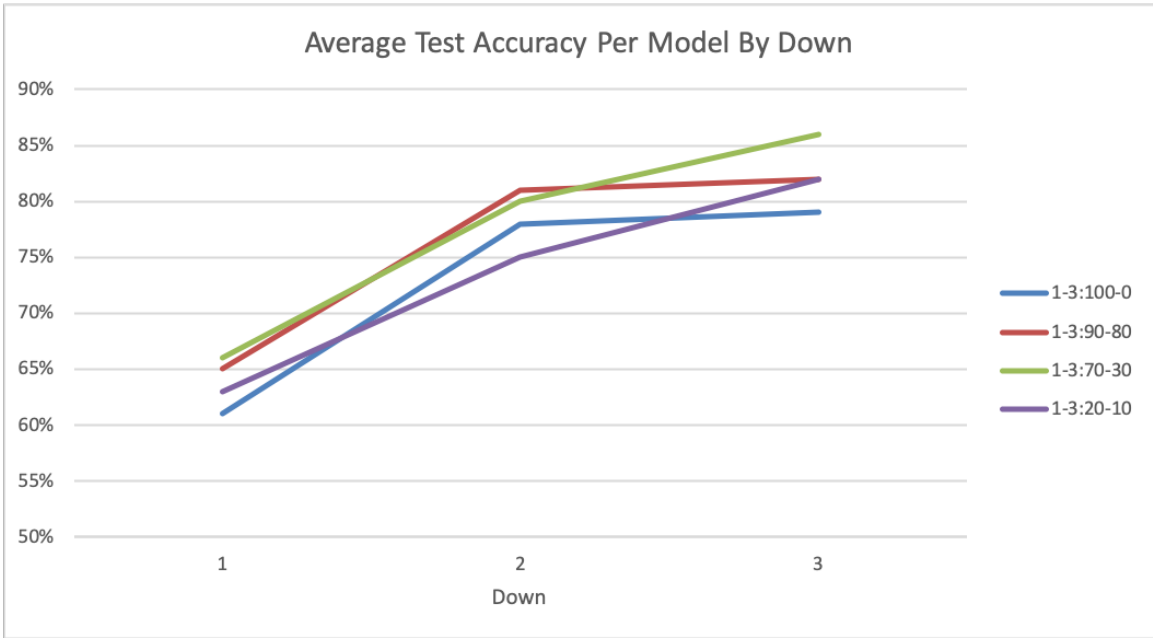



Figure 4-4: Model Accuracy per Down

the highest overall because time remaining and score difference may directly dictate play-calling in end-of-game scenarios. Additionally, the fact that teams are more predictable in the 3rd quarter than the 1st quarter suggests that data on plays from the first half can be used to predict plays for the second half. As a result of this

insight, NFL teams should consider varying their second half play-calling significantly relative to their first-half play-calling to ensure that their predictability is minimized throughout the game.

As we can see in figure 4-4, teams tend to be more predictable on later downs. This may be caused by an offense's need to get a specific number of yards on 3rd down to move the chains. Therefore, their set of plays become restricted, making them more predictable. Additionally, the overall trends of accuracy per model are similar, for 3rd down is the most predictable in all situations and 1st down is the least predictable.

### 4.2.3 Predictability vs. Success

Given the team-specific analysis we conducted, it is interesting to note which teams are the most predictable, and what makes them so predictable. Specifically, in our situational analysis, three teams were the most predictable: the Tennessee Titans, the New York Jets, and the Cincinnati Bengals. Each of these teams has a relatively balanced offense on paper. For each situation, each team's run-to-pass ratio stays $+/-6\%$ of a 50/50 balance between run-pass. However, despite this balance, each offense continues to struggle. In the past 10 years, each team has been ranked in the bottom 30% of teams in offensive efficiency [2].

We explored the idea that a team's run-pass predictability may be correlated to their offensive ranking. Thus, we ranked each team based on their average predictability in the different situations, and compared this ranking to its average offensive ranking over the past 10 years. Figure 4-5 shows how a team's predictability correlates inversely with its offensive success - the less predictable a team is based on the key factors discerned from our models, the better its offense performs. If we look at the least predictable teams from table 4.6, we see that they have seen the most success offensively over the course of the past 10 years. Despite relative consistency of coaching for the top three most successful offensive teams (the New England Patriots, the New Orleans Saints, and the Green Bay Packers), each team has maintained low predictability in their offensive play-calling. This shows the value of not only the

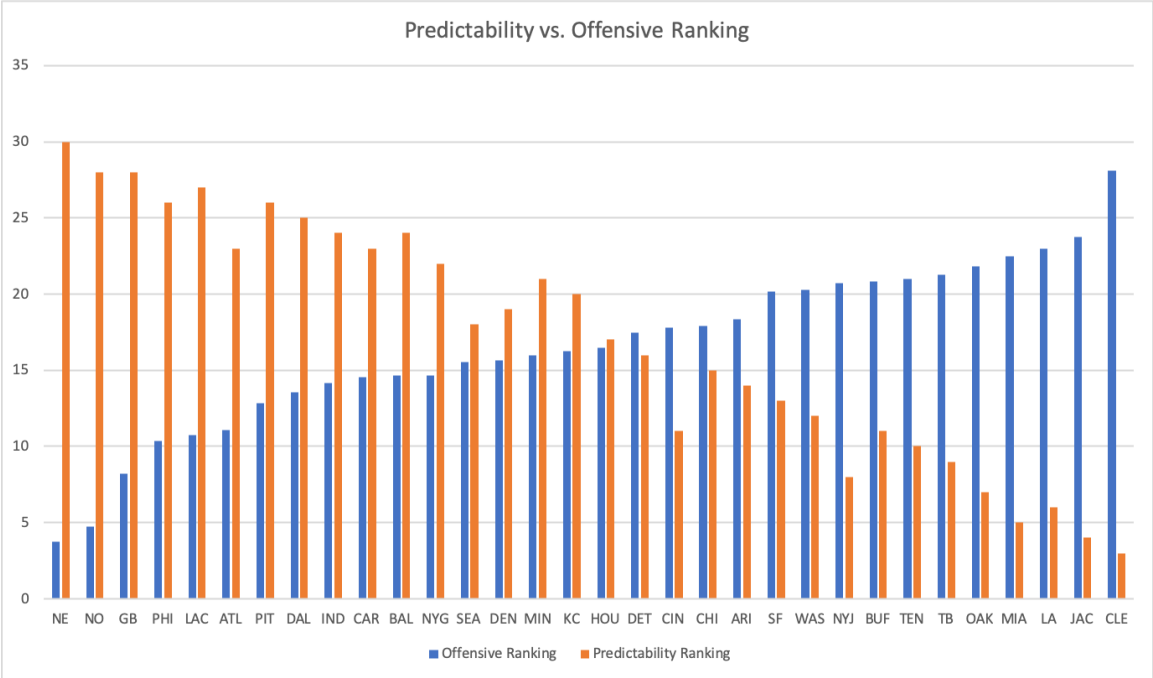



Figure 4-5: Comparing Average Predictability Ranking of Teams to Average Offensive Ranking

insights from our models, but it also showcases that reduced situational predictability can result in increased offensive success.

Moreover, if we compare the average win percentage over the last 10 years of the five most predictable teams (47.34%) to the average win percentage of the five least predictable teams (59.72%), we see a 12% difference. This supplements the value proposition of reducing predictability relative to the key factors extracted from our models, for teams with less predictable offenses tend to score more.

# Chapter 5

# Conclusion

As a whole, our machine learning models worked the best for team-specific data in restricted situations. Our relative accuracy, which is the difference between the accuracy of a model and the run-to-pass ratio of the data, was around 25% for general models and 27% for team-specific models. We were able to glean some key insights for NFL teams to leverage in different aspects of their game based on our results. We found that lower predictability directly correlates to success, which should serve as incentive for NFL teams to utilize the insights from this research to further vary their play-calling from the expected. Given the dependence of play-calling upon factors such as a team's current run-to-pass ratio within a game, teams can become less predictable by potentially varying their run-pass breakdown later in the game. Furthermore, with increased predictability in the second half of a game, teams should seek to vary play-calling in the second half especially.

This project has only scratched the surface of the potential of machine learning within the NFL. Currently, the biggest roadblock to progress in play prediction using machine learning is access to detailed data. With the increased prevalence of equipment sensors, videos, and other data sourcing methods being used in-game, NFL data will become more accessible and diverse, allowing for higher potential for machine learning models. These data sourcing methods, combined with advances in computer vision, will set the stage for a technical revolution in the game of football through machine learning.

# Bibliography


[1] Nfl team stats. http://www.nfl.com/stats/team. Accessed: 2019-03-24.

[2] Official site of the national football league.

[3] Omar Ajmeri and Ali Shah. *Using Computer Vision and Machine Learning to Automatically Classify NFL Game Film and Develop a Player Tracking System*, 2012.

[4] V Ayyadevara. *Decision Tree*, pages 71–103. 07 2018.

[5] Dale Berger. Introduction to binary logistic regression and propensity score analysis. 10 2017.

[6] Pablo Bosch. *Predicting the winner of NFL-games using Machine and Deep Learning*, 2018.

[7] Bryan Denham. *Binary Logistic Regression*, pages 119–152. 03 2017.

[8] Eric Eager. Pro football focus elite subscription. Accessed: 2019-08-12.

[9] Abdalla El-Habil. Multinomial logistic regression model. 8, 03 2012.

[10] Mark Glickman and Hal Stern. *A state-space model for National Football League scores*, 2012.

[11] T.j. Hastie and R.j. Tibshirani. Generalized additive models. *Generalized Additive Models*, page 136âĂŞ173, 2017.

[12] Sushma Jain and Harmandeep Kaur. Machine learning approaches to predict basketball game outcome. pages 1–7, 09 2017.

[13] Josh Klein and Anna Frowein. *Predicting Game Day Outcomes in National Football League Games*, 2018.

[14] Kaan Koseler and Matthew Stephan. Machine learning applications in baseball: A systematic literature review. *Applied Artificial Intelligence*, 31:1–19, 02 2018.

[15] Gunjan Kumar. *Machine Learning for Soccer Analytics*. PhD thesis, 09 2013.

[16] Michael D. Lee and Irina Danileiko. Testing the ability of the surprisingly popular method to predict nfl games. *Judgement and Decision Making*, 13(4), July 2018.

[17] Peter Lee, Ryan Chen, and Vishan Lakshman. *Predicting Offensive Play Types in the National Football League*, 2012.

[18] Tae-Hwy Lee, Aman Ullah, and Ran Wang. *Bootstrap Aggregating and Random Forest*, pages 389–429. 01 2020.

[19] Ḱennis Lock. Statistical methods in sports with a focus on win probability and performance evaluation. Master's project, Iowa State University, 2016.

[20] Sarah Mallepalle and Konstantinos Pelechrinis. *A Naive Bayes Approach for NFL Passing Evaluation using Tracking Data Extracted from Images*, 2019.

[21] Brian Miceli, Eduardo Balreira, and Thomas Tegtmeyer. An oracle method to predict nfl games. *Journal of Quantitative Analysis in Sports*, 10:183–196, 06 2014.

[22] Anuja Nagpal. L1 and l2 regularization methods, Oct 2017.

[23] Vinod Nair and Geoffrey E. Hinton. *Rectified Linear Units Improve Restricted Boltzmann Machines*, 2010.

[24] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[25] Jeffrey Pennington, Samuel Schoenholz, and Surya Ganguli. Resurrecting the sigmoid in deep learning through dynamical isometry: theory and practice. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 4785–4795. Curran Associates, Inc., 2017.

[26] Himanshu Singh and Yunis Lone. *Artificial Neural Networks*, pages 157–198. 01 2020.

[27] Brendan Teich, Roman Lutz, and Valentin Kassarnig. *NFL Play Prediction*, 2016.

[28] James Warner. *Predicting Margin of Victory in NFL Games: Machine Learning vs. the Las Vegas Line*, 12 2010.

[29] Ronald Yurko, Samuel Ventura, and Maksim Horowitz. *nflWAR: A Reproducible Method for Offensive Player Evaluation in Football*, 2018.

[30] Xinhua Zhang, Novi Quadrianto, Kristian Kersting, Zhao Xu, Yaakov Engel, Claude Sammut, Mark Reid, Bin Liu, Geoffrey Webb, Moshe Sipper, Lorenza Saitta, Michele Sebag, Charu Aggarwal, Thomas GÃďrtner, TamÃąs HorvÃąth, Stefan Wrobel, Deepayan Chakrabarti, Julian McAuley, TibÃľrio Caetano, and Lise Getoor. *Gini Coefficient*. 01 2010.